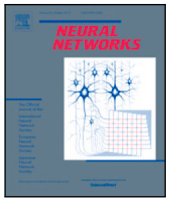
Neural Networks 173 (2024) 106171

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



Full Length Article

Methodology based on spiking neural networks for univariate time-series forecasting

Sergio Lucas *, Eva Portillo

Department of Automatic Control and Systems Engineering, Faculty of Engineering of Bilbao, University of the Basque Country (UPV/EHU), Plaza Ingeniero Torres Quevedo, 1, Bilbao, 48013, Basque Country, Spain

ARTICLE INFO

Keywords:
Spiking Neural Network
Forecasting
Supervised learning
PWM based encoding–decoding algorithm
Surrogate gradient

ABSTRACT

Spiking Neural Networks (SNN) are recognised as well-suited for processing spatiotemporal information with ultra-low energy consumption. However, proposals based on SNN for classification tasks are more common than for forecasting problems. In this sense, this paper presents a new general training methodology for univariate time-series forecasting based on SNN. The methodology is focused on one-step ahead forecasting problems and combines a PulseWidth Modulation based encoding–decoding algorithm with a Surrogate Gradient method as supervised training algorithm. In order to validate the generality of the presented methodology sine-wave, 3 UCI and 1 available real-world datasets are used. The results show very satisfactory forecasting results ($MAE \in [0.0094, 0.2891]$) regardless of the characteristics of the dataset or the application field. In addition, weights can be initialised just once to achieve robust results, boosting the advantages of computational and energy cost of SNN.

1. Introduction

Model accuracy has been the main factor to be enhanced by Machine Learning (ML) and Artificial Intelligence (AI) researchers in recent years with model efficiency being considered as a non-important criterion (García-Martín, Rodrigues, Riley, & Grahn, 2019). Nonetheless, the existing problem between IA techniques and the global energy consumption is increasingly being recognised (de Vries, 2023; Suetake, ichi Ikegawa, Saiin, & Sawada, 2023). Actually, hardware efficiency improvement and innovation in model architectures and algorithms could help to mitigate or even reduce AI-related electricity consumption in the long term (de Vries, 2023).

In this sense, Spiking Neural Networks (SNN), the so-called third generation of neural networks, have gained popularity in the last years. One of the main advantages of SNN is that they can be implemented in neuromorphic hardware with ultra-low power consumption (Sboev, Litvinova, Vlasov, Serenko, & Moloshnikov, 2016). In fact, recent works (Bu, Ding, Yu, & Huang, 2022; Deng et al., 2020; Fang et al., 2023) have claimed that SNN are more power efficient than Artificial Neural Networks (ANN), which are one of the most widely used IA algorithms (Mesanza et al., 2020; Nakai & Nishimoto, 2023; Shi et al., 2022; Zamri et al., 2022). Among other reasons, SNN gain their energy efficiency due to event-driven calculation, sparse activation, and multiplication-free characteristics (Bu et al., 2022).

The potential of SNN is that they mimic biological neurons more closely than ANN using more biologically plausible and energy efficient neuronal models as computational units. In SNN the information is encoded in temporal spike sequences or spike trains trying to imitate the nervous system. This means that the shape of the spikes does not carry any information itself, but the importance is in the number and the timing of the aforementioned spikes. On the contrary, in ANN the information is encoded in real type values, which can be understood as a rate encoding scheme (Wang, Lin, & Dang, 2020).

Given the temporal encoding of the spikes SNNs possess intrinsic features to manage temporal data. In spite of this important quality, the vast majority of works in the literature are focused on applying SNNs to classification problems such as fault prediction (de Abreu, Silva, Nunes, Moioli, & Guedes, 2023), image recognition (Qasim Gilani, Syed, Umair, & Marques, 2023), pattern recognition (Aghabarar, Kiani, & Keshavarzi, 2023), object detection (Lien & Chang, 2022), event recognition (Yao et al., 2023), and electrocardiogram classification (Feng, Geng, Chu, Fu, & Hong, 2022). Actually, little effort has been made in applying SNNs to regression problems (Rançon, Cuadrado-Anibarro, Cottereau, & Masquelier, 2021) and even much less in time-series forecasting problems due to two main reasons.

The first reason is the traditional lack of algorithms capable of encoding accurately real value data into spikes and decoding or reconstructing the SNN output into real values precisely (Arriandiaga,

* Corresponding author.
E-mail addresses: sergio.lucas@ehu.eus (S. Lucas), eva.portillo@ehu.eus (E. Portillo).

https://doi.org/10.1016/j.neunet.2024.106171
Received 17 May 2023; Received in revised form 17 November 2023; Accepted 8 February 2024
Available online 16 February 2024

Portillo, Espinosa-Ramos, & Kasabov, 2020). Among different existing encoding methods, rate encoding and temporal encoding approaches stand out. Rate encoding uses the number of spikes emitted in a time window. Popular algorithms behind this idea are Poisson distribution, HoughSpiker Algorithm (HSA), Ben's Spike Algorithm (BSA), Address-Event Representation (AER) and Adaptive Threshold-Based (ATB). On the contrary, temporal encoding is based on the distance between spikes. Among temporal encoding methods, Time-to-First-Spike and Rank Code can be highlighted. Although rate encoding is widely used, temporal encoding preserves more relevant information from the original signal than rate coding (Lopes-dos Santos, Panzeri, Kayser, Diamond, & Quian Quiroga, 2015). Nonetheless, many temporal encoding algorithms also have significant drawbacks such as the impossibility to decode the output from the SNN, which is essential in regression and forecasting problems, or the decoding process introduces significant errors in the reconstruction of the signal (Arriandiaga et al., 2020; Wang, Guo, & Adjouadi, 2016).

The second reason is the supervised training process. In general, supervised learning is considered more accurate and reliable than unsupervised learning (Kim, 2020). Given that in forecasting tasks past values are available supervised training is commonly applied (Bojer, 2022). However, the difficulties for applying Back Propagation (BP) and Back Propagation Through Time (BPTT) to SNN are due to the non-differentiable nature of the spiking units. Over the past few years a significant effort has been made to solve this limitation to implement supervised learning in SNN. The supervised learning approaches for SNN can be categorised into indirect and direct training. Indirect training lies in firstly training a conventional ANN and, secondly, the learned parameters are transferred to a functionally equivalent SNN (O'Connor, Neil, Liu, Delbruck, & Pfeiffer, 2013). Nonetheless, in these methods the signals used for training cannot accurately represent the statistical behaviour of the spike trains, which results in insufficient accuracy for many practical applications (Lee, Delbruck, & Pfeiffer, 2016). Thus, this proposal advocates for direct training in SNN.

Section 1.1 proposes a systematic literature review about the forecasting proposals based on SNN. The literature works are grouped into categories and analysed attending to relevant aspects that define both forecasting and SNN scopes.

1.1. Related work

A forecasting problem can be understood as the prediction of future values of a series based solely on its past observations (Semenoglou, Spiliotis, & Assimakopoulos, 2023). In this sense, this work proposes two main basis to classify the existing forecasting proposals based on SNN. The first one is the application scope of each proposal, which can be either a general methodology (G) valid for any forecasting task or a solution for a specific problem or application field (S). The second basis is the difficulty in use of each proposal, which is categorised depending on three criteria: (a) if other AI and ML techniques are applied alongside the SNN the difficulty in use increases; (b) regarding the learning approach and algorithm not using a standard, well-known, general and available algorithm increases the difficulty; (c) if a more complex SNN topology than the traditional one (basic feedforward) is used the difficulty in use increases. In this context three categories are defined depending on how many criteria are met in each proposal:

- High Difficulty (HD): two or more of the criteria described above are met.
- Medium Difficulty (MD): one of the criteria described above is met.
- Low Difficulty (LD): none of the criteria described above is met.

According to these basis in Table 1 a categorisation of the literature review is proposed. For each literature reference classified as (1) G or S, and as (2) HD, MD or LD, the following information is provided:

- Information about forecasting
  - i Number of previous values required (Np) to make a prediction.
  - ii Prediction horizon (H).
- Information about SNN
  - i Encoding method (TE: temporal encoding, RE: rate encoding).
  - ii Type of learning approach (U: unsupervised, SD: supervised direct, SI: supervised indirect) and learning algorithm applied to SNN.
  - iii Neuron model.
  - iv Structure (number of layers and neurons) of the network.
- Information about the datasets
  - i Number of datasets (N).
  - ii Application field.
  - iii Availability (DNA (a): Dataset Not Available because the source is not provided; DNA (b): Dataset Not Available because the link of the source has expired; DNA (c): Dataset Not Available because the information is partially provided; DA: Dataset Available).

Notice that in Table 1 "–" indicates that the according information is not provided in the literature reference impeding comparison, reproducibility and replication of the corresponding proposal.

The first general conclusions that can be drawn from Table 1 are: (1) The application field of SNN for forecasting is relatively reduced with most of the works within the field of energy (Brusca, Capizzi, Lo Sciuto, & Susi, 2019; Capizzi, Sciuto, Napoli, Woźniak, & Susi, 2020; Chen et al., 2016; Dudek et al., 2022; Han, Li, & Qian, 2018; Kulkarni, Simon, & Sundareswaran, 2013; Madhiarasan & Deepa., 2016; Sharma & Srinivasan, 2010; Sun et al., 2016; Wang, Xue, Liu, Peng & and Jiang, 2020; Wei, Wang, Niu, & Li, 2021), followed by air pollution (Liu, Lu, Wang, & Kasabov, 2021; Maciąg, Kasabov, Kryszkiewicz, & Bembenik, 2019; MacIag, Kryszkiewicz, & Bembenik, 2020) and financial (Matenczuk et al., 2021; Reid, Hussain, & Tawfik, 2013, 2014) fields; (2) Regarding the application scope the vast majority of the works are specific proposals (S).

As stated above, works classified as HD meet at the same time two or more of the aforementioned criteria. HD works in Table 1 are Brusca et al. (2019), Capizzi et al. (2020), Dudek et al. (2022), Laña, Capecci, Del Ser, Lobo, and Kasabov (2018), Liu et al. (2021), Maciąg et al. (2019), MacIag et al. (2020), Reid et al. (2013, 2014), Saeedinia, Jahed-Motlagh, Tafakhori, and Kasabov (2021), Wei et al. (2021).

In Saeedinia et al. (2021) the criteria of using a complex SNN topology and a non-standard learning approach are met. In this work a 3-dimensional (3D) SNN is proposed. The SNN is structured according to Magnetic Resonance Imaging (MRI) personal data consisting of interconnected observed (input/output) and hidden neurons. Hence, the proposed 3D SNN structure is limited to MRI problems. In addition, 300 neurons (15 observed and 285 hidden) based on Izhikevich model are used. This model is commonly applied in neuroscience research due to its high biological plausibility. However, Izhikevich neuron model is also considered a complex model since it requires two differential equations to describe the functioning of a neuron. Thus, it is unusual to apply it in ML research, where the Leaky Integrate-and-Fire (LIF) model is the most widely used model due to its simplicity (Schuman et al., 2017). Regarding the learning approach, in Saeedinia et al. (2021) a combination between supervised (GDM: Gradient Descent Method) and unsupervised (STDP: Spike Time Dependent Plasticity) strategies is carried out. The combination of two learning approaches during the training process increases considerably the computational and energy cost, countering the advantages offered by SNN.

Table 1
Literature review of the forecasting with SNN.

Applic. Scope	Difficulty in use	Applied techniques	Ref.	Dataset			SNN information				Forecasting information	
				N.	Applic. Field	Availability	Encoding Method	Learning approach and algorithm	Structure	Neuron Model	Np	H
S	HD	SNN + Output Module	Saeedinia et al. (2021)	2	Health	DNA (a) DNA (b)	–	U (STDP) and S (GDM)	15 observed neurons and 285 hidden neurons	Izhikevich	–	–
		Polychronous SNN	Reid et al. (2013)	3	Financial	3xDNA (a)	TE. Absolute Spike Timing	U (STDP).	–	Izhikevich	–	1, 5
			Reid et al. (2014)	3	Financial	3xDNA (b)						1, 5, 10, 15
		Online evolving SNN (OeSNN)	MacIag et al. (2020)	1	Air Pollution	DNA (c)	InputLayerEncoding Procedure	OeSNN-IP	–	–	–	1, 12
		NeuCube + dynamic evolving Spiking Neural Network (deSNN)	Capizzi et al. (2020)	1	Energy (Biogas Production)	DNA (a)	RE. BSA.	U (STDP) and S (SDSP + Rank Order (RO)).	10 × 10 × 10 8 × 8 × 8 6 × 6 × 6 4 × 4 × 4	LIF	10	100
			Brusca et al. (2019)	1	Energy (Power Generation)	DNA (a)	RE. BSA.	U (STDP) and S (SDSP + RO).	10 × 10 × 10	LIF	5	1
			Laña et al. (2018)	1	Traffic	DNA (c)	RE. ATB.	U (STDP) and S (STDP + RO).	90 × 60	LIF	20	1
		Clustering + NeuCube + evolving Spiking Neural Network (eSNN)	Maciąg et al. (2019)	1	Air Pollution	DA	RE. ATB and Spike Rate.	U (STDP) and S (eSNN training).	3 dimensional template network with 1000 neurons	LIF	12	1, 3, 6
			Liu et al. (2021)	2	Air Pollution	DNA (b) DA	Step-Forward Encoding.	U (STDP) + S.	–	LIF	–	1, 3, 6, 12, 14
		VMD + ANN – SNN + Loihi hardware	Dudek et al. (2022)	1	Energy (Wind Power)	DNA (a)	–	SI.	–	–	–	1
		Empirical Wavelet Transform + RNN + Convolutional SNN	Wei et al. (2021)	4	Energy (Wind Speed)	4xDNA (a)	–	SD. SG.	–	LIF	10	–
	MD	Grey Correlation Analysis (preselection samples) + SNN	Chen et al. (2016)	1	Energy (Power Generation)	DNA (a)	TE. Time To First Spike	SD. SpikeProp.	[16 - (empirical formula) - 16]	SRM	–	1
		VMD + SNN	Sun et al. (2016)	1	Energy (Carbon Price)	DNA (c)	–	SD. SpikeProp.	3 layers. Neurons depend on subseries	SRM	Depending on subseries	–
		SNN + Continuous Ranking Probability Score (CRPS) + Group Search Optimiser (GSO)	Wang, Xue, et al. (2020)	2	Energy (Wind Power)	DA DNA (a)	–	SD. SpikeProp.	[12-11-17-(number of intervals defined)]	IF	–	1
		SNN + SNN	Kulkarni et al. (2013)	2	Energy (Electricity load)	DNA (b)	–	SD. GDM.	[29 - (adjustable hyperparameter) - 6] [68 - (adjustable hyperparameter) - 48]	SRM	Depending on variables	1
	LD	SNN	Matenczuk et al. (2021)	1	Financial	DNA (a)	–	–	–	–	14	1
			Han et al. (2018)	1	Energy (Wind Speed)	DA	–	SD. SpikeProp.	[28 - (empirical formula) - 24]	SRM	24 + other variables	24
			Madhiarasan and Deepa. (2016)	1	Energy (Wind Speed)	DNA (a)	–	SD. SpikeProp.	[7 - 9 - 1]	SRM	–	–
			Yang and Zhongjian (2011)	1	Grain Yield	DNA (a)	–	SD. SpikeProp.	[10 - 5 - 1]	–	–	–
G	MD	SNN	Sharma and Srinivasan (2010)	2	Energy (Electricity Price)	DA DNA (b)	TE. Time Interval.	SD. Evolutionary Strategies	[7 - 12 - 1]	IF	–	1

In Reid et al. (2013, 2014) successful forecasting in financial time-series datasets are performed. Nonetheless: (i) the STDP algorithm is applied, which is an unsupervised strategy considered less accurate than the supervised one for forecasting purposes (Kim, 2020); (ii) the aforementioned Izhikevich complex model is used; (iii) Polychronous SNN based on polychronization (Izhikevich, 2006) is used, which lies in organising the SNN into clusters of neurons with similar spike firing characteristics. Hence, the training process of a Polychronous SNN require higher memory and computational cost compared to the basic feedforward SNN topology.

The literature work (MacIag et al., 2020) is classified as HD work since the criteria of using a complex SNN topology and a non-standard learning approach are met. In particular, a model based on Online evolving SNN (OeSNN) is presented. This type of SNN were originally designed for classification problems but authors of MacIag et al. (2020) adopt them for air pollution forecasting. However, additional information about air pollution such as temperature or wind parameters are introduced as input data which may distort the forecasting concept and it is out of scope of this proposal. In addition, authors propose a new encoding algorithm (InputLayerEncoding Procedure) and a new learning approach (OeSNN-IP). Nonetheless, they are only suitable for OeSNN and cannot be reproducible for other SNN topologies.

HD works are also characterised by the use of other techniques alongside SNN, which is the case of Neucube (Brusca et al., 2019; Capizzi et al., 2020; Laña et al., 2018; Liu et al., 2021; Maciąg et al., 2019). NeuCube is a well-known computational framework for the development of machine learning models inspired in the brain functioning. This framework consists of three different blocks: data encoding block, SNN cube block and prediction block. In the case of forecasting related works that use NeuCube: (i) in the encoding block rate encoding algorithm is applied, which is less accurate than temporal encoding (Lopes-dos Santos et al., 2015); (ii) in SNN cube block topologies such as dynamic evolving Spiking Neural Network (deSNN) (Brusca et al., 2019; Capizzi et al., 2020; Laña et al., 2018) and evolving Spiking Neural Network (eSNN) (Liu et al., 2021; Maciąg et al., 2019) are used, which were designed for classification problems; (iii) the structures commonly applied in SNN cube block consist of 3D SNN, which increase the computational cost of the training; (iv) apart from the 3D SNN, other LIF spiking neurons are required in the prediction block to make the forecast; (v) an unsupervised learning approach to

train the SNN cube block and a supervised learning approach to train the spiking neurons in the prediction block are applied, increasing considerably the computational and energy cost.

Other HD works such as Dudek et al. (2022) and Wei et al. (2021) combine SNN with techniques that pre-process the time-series. In the case of Dudek et al. (2022) Variational Mode Decomposition (VMD) technique is applied, while in Wei et al. (2021) Empirical Wavelet Transform algorithm is used. This idea of pre-processing the data is applied to decompose the original series into subseries in order to simplify the process of modelling and ease of inference (Suradhaniwar, Kar, Durbha, & Jagarlapudi, 2021). Then, the forecasting is made by applying different SNN to each subseries. However, these actions lead to specialised models for each subseries focusing each SNN on specific characteristics of the data. Notice that building and training one SNN per subseries increases considerably the computational cost of the methodology. Furthermore, these works also present some other limitations: (i) in Dudek et al. (2022) an indirect supervised training is performed, which has been demonstrated to be less accurate and more energy consumption than direct supervised training algorithms (O'Connor et al., 2013); (ii) in Wei et al. (2021) a Gated Recurrent Unit (GRU) is used as principal forecasting unit instead of SNN. The SNN are used to predict the forecasting errors made by each GRU. Hence, this is not a solution based only on SNN and the computational and energy cost advantages of the SNN are not fully exploited.

Works classified as MD meet only one of the criteria stated above. MD works in Table 1 are Chen et al. (2016), Kulkarni et al. (2013), Sun et al. (2016), Wang, Xue, et al. (2020).

The criteria of using other techniques alongside SNN is met in Chen et al. (2016) and Sun et al. (2016) since both pre-process the time-series. In Chen et al. (2016) a forecasting of photovoltaic system power generation is carried out. A grey correlation analysis (GRA) is applied in order to classify the data from the dataset in four different groups with similar characteristics. Then, a SNN is used for each group. In the case of Sun et al. (2016), the same strategy is performed but VMD technique is used to decompose the original time-series. The idea of applying different SNN to groups with similar characteristics reduces the generality of the forecasting methodology and increases considerably the computational cost since it is necessary to build and train a different SNN for each group. In both works Spike Response Model (SRM) is applied as neuron model, which is commonly applied in SNN.

In Wang, Xue, et al. (2020) the criteria of using more techniques alongside SNN is also met. In this case a traditional SNN is combined with other statistical techniques to directly calculate the varying intervals of future wind power with associated confidence levels. The methodology proposed is based on using the group search optimiser (GSO) algorithm as optimisation algorithm, where the objective function to reduce is the probability ranges calculated by the continuous ranking probability score (CRPS). Hence, the methodology is focused on probabilistic forecasting, being a specific field within forecasting and not pertinent for other forecasting problems. Furthermore, no information is provided about the encoding method and the previous values required to make the forecasting.

In Kulkarni et al. (2013) two consecutive SNNs are used for a short-term load forecasting. The first SNN is used to forecast the temperature profile. However, as input data other variables such as the humidity or the maximum solar radiation of the previous day are introduced. Similarly, in the second SNN many other inputs are required such as the maximum and minimum demand in the last day in order to forecast the electrical load. Again, introducing as input more variables than the one to be forecasted may distort the forecasting concept and it is out scope of this proposal.

In LD works none of the criteria stated above are met. LD works in Table 1 are Han et al. (2018), Madhiarasan and Deepa. (2016), Matenczuk et al. (2021) and Yang and Zhongjian (2011).

In Matenczuk et al. (2021) a comparison between the performance of traditional NN and SNN for financial time-series forecasting is performed. However, regarding SNN there is a lack of information concerning the encoding method, the learning approach, the structure of the network and the neuron model, impeding comparison, reproducibility and replication.

In Han et al. (2018), Madhiarasan and Deepa. (2016) and Yang and Zhongjian (2011) traditional SNN topologies with SpikeProp learning algorithm are performed. However, some drawbacks are the following: (i) no information is supplied about the encoding method in any of the works; (ii) in Han et al. (2018) other variables such as temperature, humidity and pressure are introduced as input in the network, which is out of scope of this proposal; (iii) in Madhiarasan and Deepa. (2016) and Yang and Zhongjian (2011) there is no information provided about the number of previous values required to make the forecast and the prediction horizon, impeding comparison, reproducibility and replication.

To the best of the author's knowledge, only one Sharma and Srinivasan (2010) in the literature defines a general methodology. However: (a) authors claim its generalisation capability only within the energy application field; (b) information about the encoding method is not complete enough; (c) evolutionary strategies are applied as supervised learning algorithm, thus increasing the difficulty in use; (d) no information is provided about the number of previous values introduced as input data into the network; (e) although one dataset is available, authors mention a special normalisation and data processing without providing detailed information, which unables the results comparison with other methodologies.

Thus, from the literature review it can be concluded that:

1. There is no proposal of supervised training general methodology based on SNN applicable to any forecasting problem independently of the characteristics of the application field.
2. A wide variety of proposals applies different ML and AI techniques alongside SNN. This entails an increase in model complexity and, consequently, an increase in computational and energy cost which leads to counter the advantages offered by SNN.
3. The difficulty and thus the implementation time for the practical application of the existing proposals is considerable, requiring a high level of expertise in SNN.
4. No proposal provides all the information necessary to reproduce their work, impeding the comparison of different approaches.

In this sense, the aim of this work is to provide a new methodology for univariate time-series forecasting with SNN, in particular for one-step ahead forecasting. The methodology is based on the combination of a temporal encoding algorithm and a direct supervised training strategy which, as explained above, are both encouraged in the literature review for this type of problem. Regarding the encoding algorithm our temporal encoding method based on PulseWidth Modulation (PWM) is extended and used (Arriandiaga et al., 2020). This algorithm provides substantial improvements in terms of precision in the encoding and decoding phases with respect to predecessor algorithms. Concerning the supervised training strategy a proposal based on Surrogate Gradient (SG) method (Neftci, Mostafa, & Zenke, 2019) is presented.

Thus, the main contribution of this work is a new methodology that can be easily applied to any time-series regardless of its characteristics and application field, boosting the use of SNN in forecasting problems. In addition, this work makes specific contributions such as:

- A systematic review of the state-of-the-art for SNN based forecasting, as shown in Table 1.
- A general methodology validated on five different application fields.
- Ultra low-latency one time-step solution.

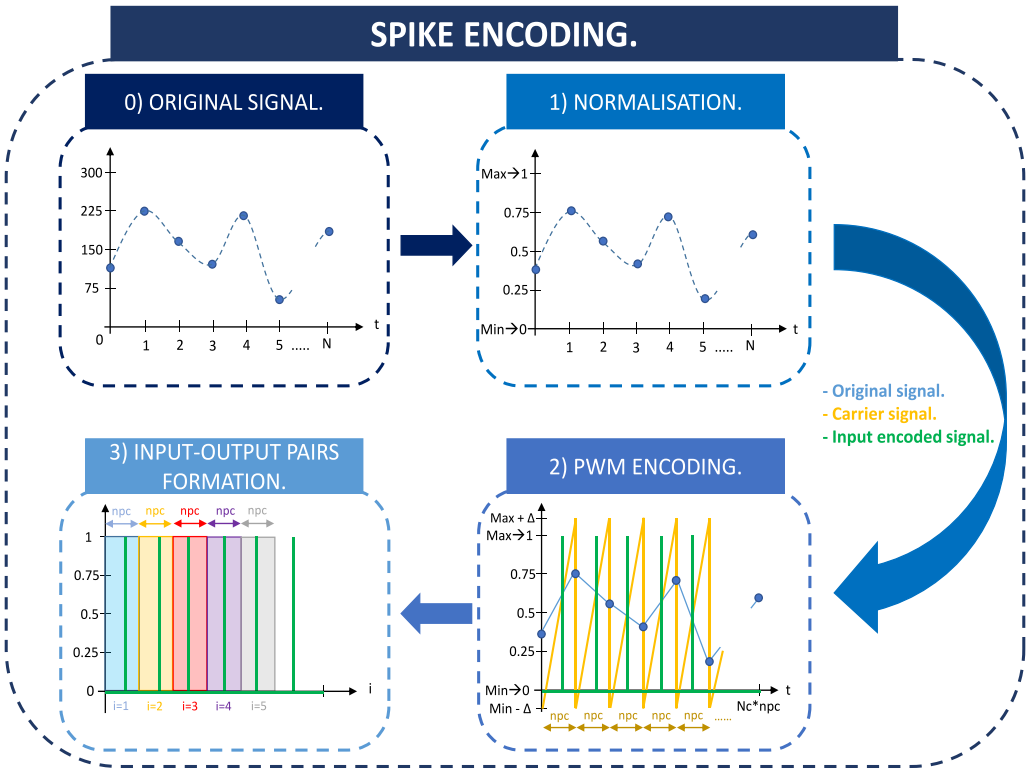


Fig. 1. Spike encoding of the data.

- A robust methodology whose results do not vary depending on the initialisation of the SNN weights.
- An extension of the PWM based encoding–decoding algorithm.
- A loss function suitable for the PWM based algorithm.

The rest of this paper is organised as follows: Section 2 exposes the methodological approach proposed for a supervised training for univariate time-series forecasting with SNN. Section 3 describes the experimental datasets used to assess the performance of the proposed approach. Section 4 presents the results and the discussion is included in Section 5. Finally, Section 6 concludes the paper.

2. Methodological approach

As stated in the first section, Surrogate Gradient (SG) method (Neftci et al., 2019) is applied as direct supervised training algorithm. SG methods are based on using differentiable surrogate functions in place of discrete activation functions during the training phase and applying gradient-based optimisation techniques with the surrogate function (Han & Lee, 2021), which enables the application of BP algorithms to SNN.

2.1. Spike encoding of the data and input–output pairs for training

Before the training based on SG method it is necessary to encode all the real value information into spikes. Although it is claimed that the encoding process has a great influence on the performance of supervised learning algorithms for SNN (Wang, Lin, & Dang, 2020), few works in the literature (see Table 1) clearly point out how the data is encoded and decoded and how input–output pairs are formed.

For the proposed methodology a recent temporal encoding algorithm based on PWM is used (Arriandiaga et al., 2020). It should be noticed that the ultra-low power consumption advantages of SNN may be offset by using rate coding algorithms since they introduce long latency during which many spikes are processed for ensuring decision accuracy (Xu, Zhang, Liu, & Li, 2020).

The PWM based encoding algorithm is based on the PWM principles to encode the data emitting spikes when there is any intersection between the original signal (time-series signal) and the carrier signal (PWM signal) commonly represented by a saw-tooth. Fig. 1 illustrates the method for spike encoding data.

2.1.1. Normalisation

As explained in Arriandiaga et al. (2020), for the task of forecasting it is necessary to have one spike at each time-step. In order to ensure one intersection at every time-step the original signal must be between the limits of the carrier signal, i.e. it must be normalised.

In principle it could be thought that both signals (carrier and original) should be within $[min, max]$ range. However, as shown in the first case of Fig. 2, when two consecutive values of the original signal are at the extremes of the saw-tooth no intersection occurs and, thus, no spike is emitted. In order to solve these cases the normalisation of the original signal is within $[min, max]$ range and the normalisation of the carrier signal is within $[min - \Delta, max + \Delta]$ range. The second case of Fig. 2 shows that this normalisation always ensures the intersections solving the spike emission for two consecutive extreme values. Notice that this implies that a spike will never be emitted at the beginning/end of the carrier. As is common in NN in general $[min, max]$ range can be defined as [0,1].

2.1.2. PWM encoding–decoding algorithm

Once the data is normalised, the encoding algorithm itself can be applied. The PWM algorithm works based on two hyperparameters. The first hyperparameter is the number of carriers (nc) straightly related to the carrier frequency. The value of nc determines the number of intersections to be produced, in other words, it determines the total number of spikes (as shown in Fig. 3a). As explained above, for the task of forecasting it is necessary to have one value at each time-step. Hence, nc is equal to the number of values of the time-series minus one (Arriandiaga et al., 2020).

The second hyperparameter is the number of points per carrier (npc) wave, which is related with the resolution. The more the npc, the

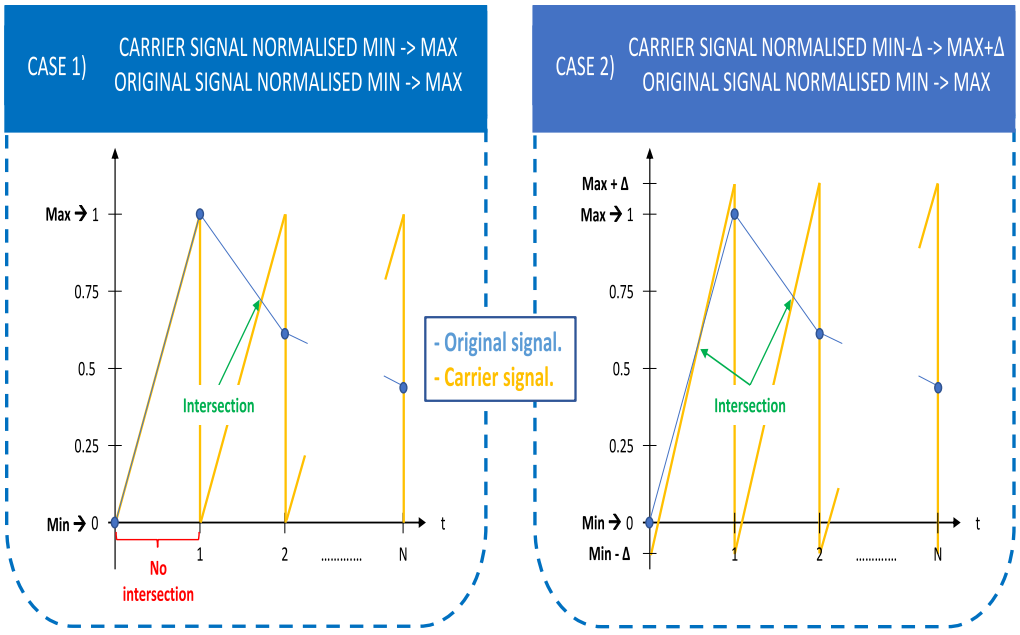


Fig. 2. Normalisation of the signals.

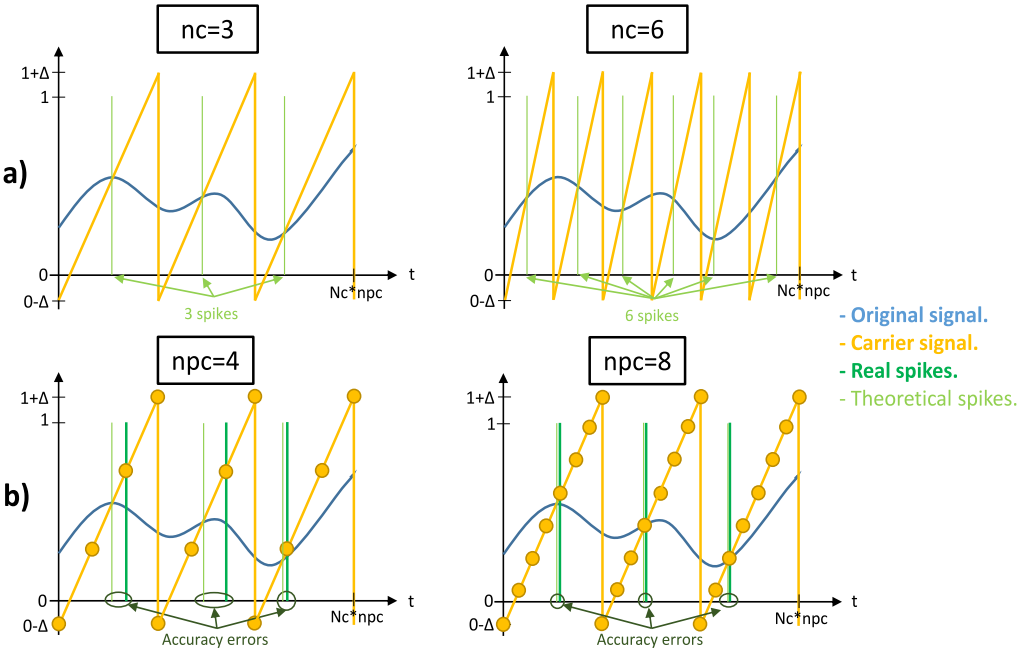


Fig. 3. Influence of the hyperparameters of the PWM algorithm.

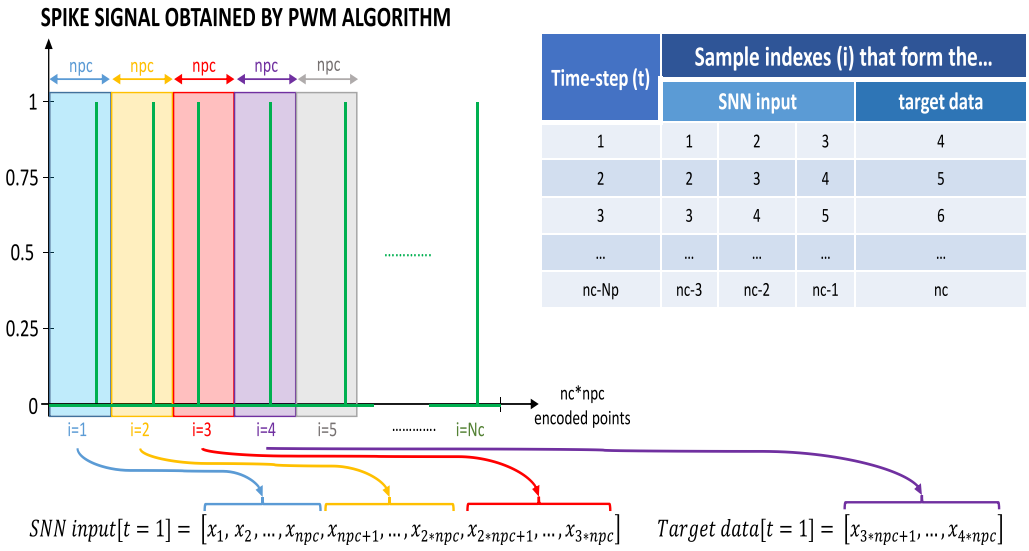
better the accuracy of the reconstruction. However, it also increases the computational and memory cost. One of the greatest benefits of the PWM encoding–decoding algorithm is that the value of npc enables any user to establish a trade-off between the accuracy of the encoding–decoding and the computational and memory cost. Thus, the value of this hyperparameter depends on the application requirements.

The encoding process with PWM algorithm is based on two steps. Firstly, an interpolation of the normalised time-series signal is performed in order to ensure its dimension is the same as the carrier ($nc * npc\ points$). Then, the spikes are obtained by the PWM encoding algorithm.

2.1.3. Input–output pairs

The spike signal after applying the PWM encoding algorithm consists of $nc * npc$ values. If this signal is divided according to nc, nc samples formed by npc values each are obtained. In order to illustrate how the input–output pairs are formed Fig. 4 shows different samples with different coloured rectangles. Variable i represents the chronological index of each sample, being $i \in [1, \ldots, nc]$. It should be noticed that in Section 2.3 the spike signal will be split into training, validation and test datasets.

In forecasting it is necessary to have the real information about the previous values. Thus, if Np samples are used for one-step ahead



Time-step (t)	Sample indexes (i) that form the...			
	SNN input			target data
1	1	2	3	4
2	2	3	4	5
3	3	4	5	6
...	...	...	...	...
nc-Np	nc-3	nc-2	nc-1	nc

Fig. 4. Input–output pairs formation for the first sample to be forecasted for $Np = 3$.

forecast for the first Np samples of the time-series it is not possible to make the forecast. In this sense, as shown in Fig. 4, for the case of $Np = 3$ the first sample to be forecasted is i = 4. At the bottom of Fig. 4 it is illustrated for $Np = 3$ and $t = 1$ the SNN input (i = 1, i = 2, i = 3) together with its corresponding target ($i = 4$). Also notice that the input–output pairs are created in chronological order as shown in the table in Fig. 4.

2.2. SNN architecture

The aim of this methodology is to achieve satisfactory forecasting results with the simplest SNN. Thus, the SNN architecture proposed is formed exclusively by the input and output layers. This kind of SNN are known as single-layer SNN and are widely used due to its simple architecture (Wang, Lin, & Dang, 2020).

At each time-step all the necessary samples have to be fed into the SNN. This means that the number of neurons in the input layer for the proposed architecture is equal to $Np * npc$.

In the case of output layer the number of neurons is equal to npc since the scope is one-step ahead forecasting. Fig. 5 shows the outline of the proposed architecture using the SNN coloured input and target data of Fig. 4.

Finally, among the well-known spiking neuron models (Han & Lee, 2021) in this methodology the LIF model (Gerstner, Kistler, Naud, & Paninski, 2014) is used due to its simplicity and low computational cost. The LIF model and its variants are one of the widely used instances of the spiking neuron model (Yamazaki, Vo-Ho, Bulsara, & Le, 2022). This model is usually abstracted into a resistor-capacitance circuit in which the subthreshold dynamic is described by the following differential equation:

$$\tau_m \cdot \frac{dV_m}{dt} = -(V_m(t) - V_{rest}) + R_m \cdot I(t) \tag{1}$$

where V_m is the membrane potential, V_{rest} is the resting potential, R_m is the leakage resistance, $\tau_m = C_m \cdot R_m$ is the time constant or leaky integrator which determines how quickly the membrane potential changes, C_m is the membrane capacitance, and $I(t)$ is the sum of the current supplied from the jth neuron to the ith neuron which can be expressed (Wang, Xue, et al., 2020) as:

$$I(t) = \sum_j w_{ij} \sum_f \delta(t - t_j^{(f)}) \tag{2}$$

where w_{ij} is the weight between the jth neuron and ith neuron, δ is a Dirac function and $t_j^{(f)}$ is the firing time of the neuron j.

2.3. Training

The proposed training methodology is based on BP with a SG method. The training is implemented with SpikingJelly framework (Fang et al., 2023) with the sigmoid function as SG.

Fig. 6 illustrates the training methodology. Firstly, the spike signal is divided chronologically into training, validation and test sets (SPLITTING). Secondly, the supervised training methodology is carried out, which is based on an incremental learning strategy under the following procedure:

1. A SNN input from the training dataset is introduced and propagated throughout the network. In Section 2.3.1 the analytical equations that govern the forward propagation are explained.
2. An output is emitted at each time-step. In Section 2.3.2 an extension of the PWM based algorithm is proposed for (a) the case that the output contains more than one spike, and (b) the case that no spike is emitted in the output.
3. A loss function is computed between the SNN output and the target data. In Section 2.3.3 a specific loss function for this methodology is proposed.
4. The loss function is backpropagated and the SNN parameters are updated as explained in Section 2.3.4.
5. Steps 1, 2, 3 and 4 are repeated until the last SNN input from the training dataset is introduced, completing one epoch.
6. At the end of each epoch the SNN with the last updated parameters is simulated with the training and validation datasets. The aim is to estimate the training and validation errors for that epoch based on an Early Stopping fashion as explained in Section 2.3.5. As a result, the SNN that yields best results is selected and tested with the test dataset.

2.3.1. Samples propagation

For numerical simulation the dynamics of any spiking neuron model in discrete time can be described with the following three equations (Fang et al., 2021):

$$H^t = f(V^{t-1}, X^t) \tag{3}$$

$$S^t = \Theta(H^t - V_{th}) \tag{4}$$

$$V^t = H^t(1 - S^t) + V_{rest} \cdot S^t \tag{5}$$

Eq. (3) is related with the charging phase of a neuron. H^t is the value of the membrane potential of a neuron after receiving an external

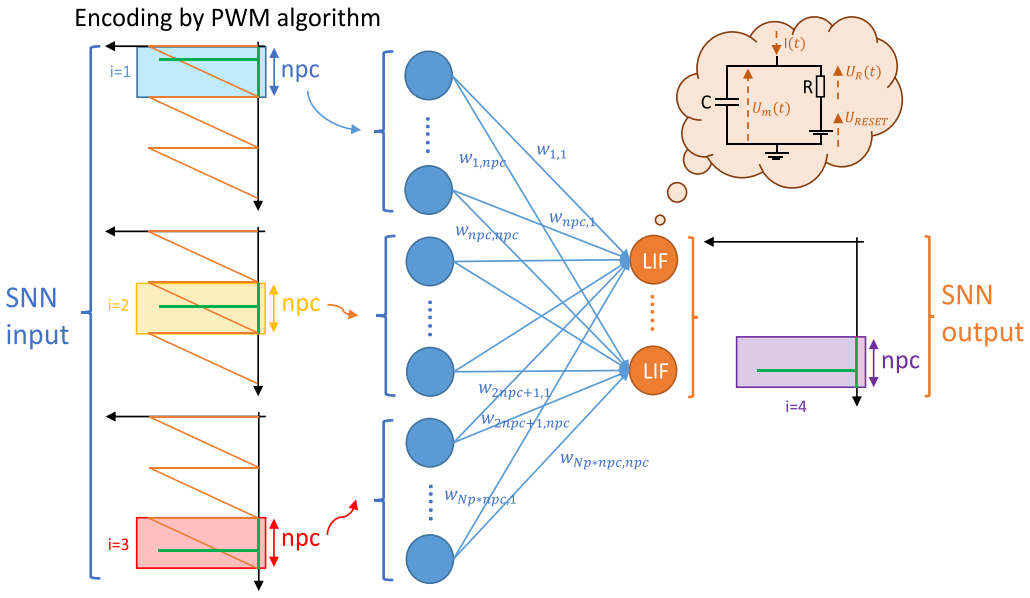


Fig. 5. SNN architecture with 0 hidden units.

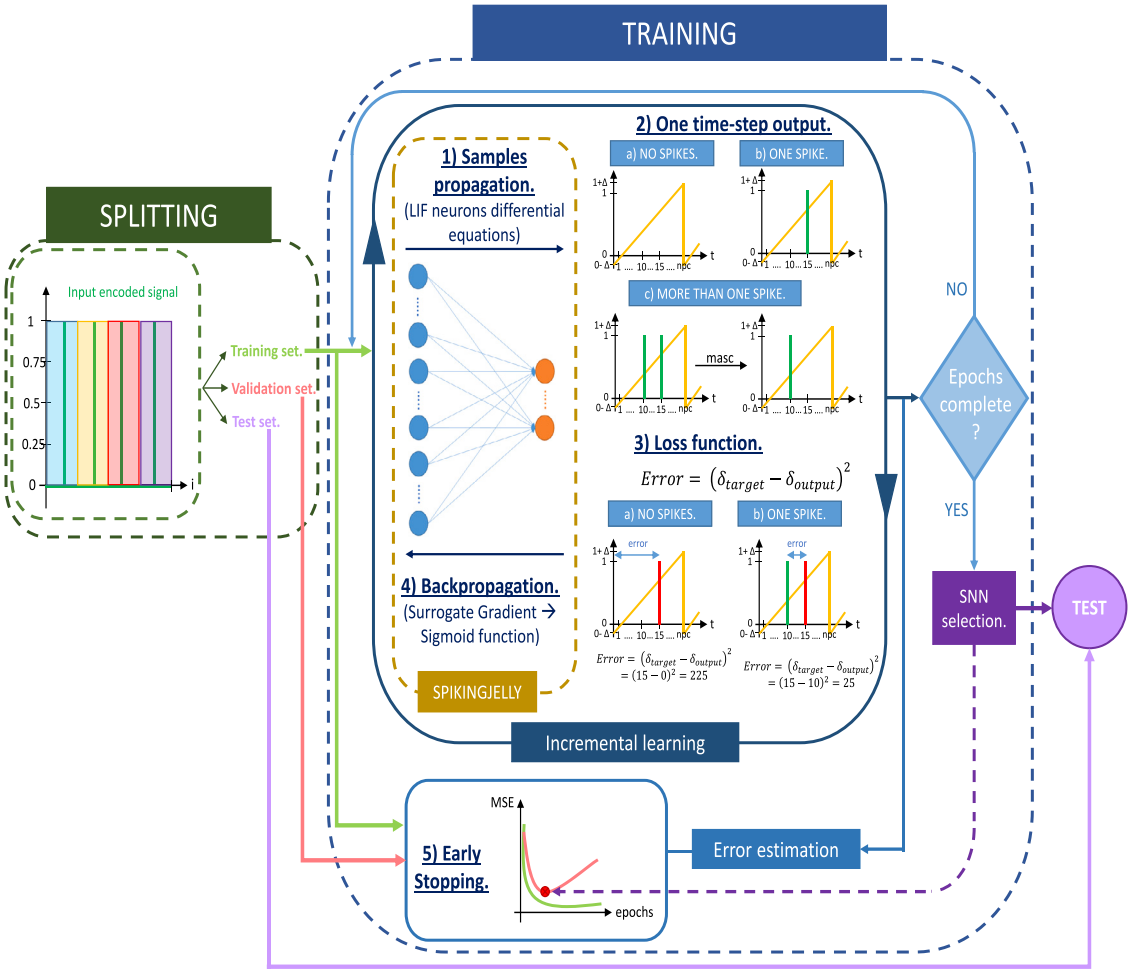


Fig. 6. Training methodology.

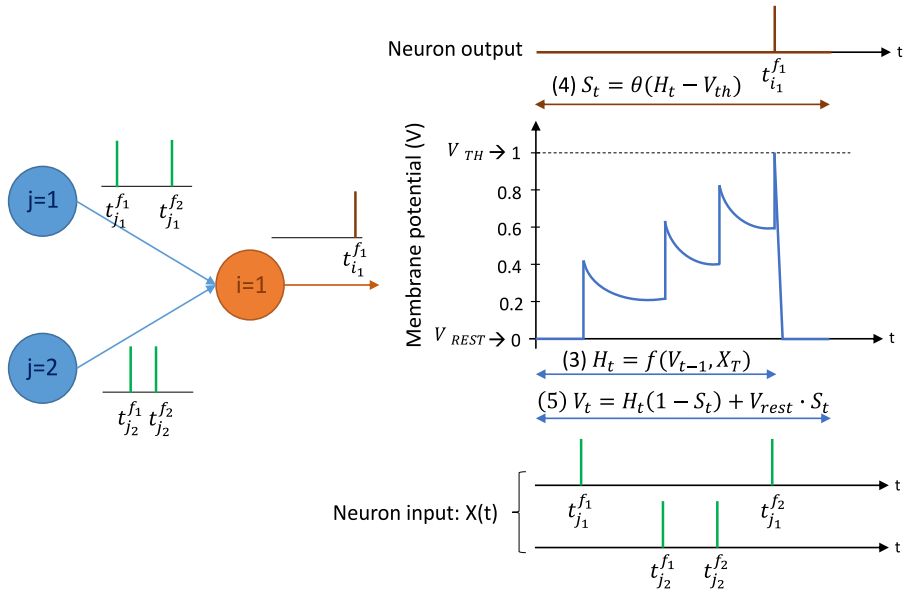


Fig. 7. Conceptual scheme of the LIF neuron dynamics.

input (X^t) and before firing a spike. The value of H^t does not only depend on the external input (X^t) at the current time-step t, but also on the residual value of the membrane potential V^{t-1} at the previous time-step $t-1$. The function $f(\cdot)$ is different for each spiking neuron model. In the case of the LIF model this function takes the form of Eq. (1) so as Eq. (3) can be rewritten as:

$$H^t = V^{t-1} + \frac{1}{\tau}[-(V^{t-1} - V_{rest}) + X^t] \tag{6}$$

Eq. (4) is related with the firing phase of a neuron. In this equation the value of the membrane potential H^t is compared with a firing threshold V_{th}. S^t represents the output spike at time-step t: if $H^t \geq V_{th}$ the neuron will fire a spike $S^t = 1$, otherwise $S^t = 0$. For this purpose the Heaviside step function ($\Theta(x)$) is applied:

$$\Theta(x) = \begin{cases} 1, \ if \ x \geq 0 \\ 0, \ if \ x < 0 \end{cases} \tag{7}$$

Eq. (5) is related with the residual value remaining of the membrane potential from one time-step to the next one. The value of V^t depends on whether the neuron has fired or not at one time-step. In the case that the neuron does not fire, V^t is equal to the load value reached (H^t) at time-step t. On the contrary, if a spike is emitted, the value of the membrane potential returns to a resting potential (V_{rest}). Eq. (5) is also called hard reset (Fang et al., 2021).

Fig. 7 illustrates the dynamic of a spiking neuron with the aforementioned equations. Notice that, in terms of Systems Engineering, the charging phase is the impulse response of a first order system.

2.3.2. *One time-step output: PWM based encoding–decoding algorithm extension for loss function and simulation*

Notice that in many application fields such as bioengineering or industrial sector the final user may need to monitor the real physical values of the SNN output, i.e. in system units.

In this sense, the PWM based algorithm can also be used to decode the SNN output. For decoding the same carrier signal used for encoding is applied so as the intersections between this carrier signal and the spike emitted at the SNN output signal are detected. This method yields the SNN output in system units as seen in Fig. 8.

At this point it should be noticed that this algorithm originally was intended to encode or decode one only spike per saw-tooth (Arriandiaga et al., 2020) as shown in Fig. 4, where the input data at each time-step is formed by Np spikes. However, the number of spikes at the output depends on how many neurons are activated at each time-step. In this context, three cases can occur according to the number of spikes present at one time-step output. The three cases and the decoding process proposed for each of them are the following:

- No spike at the output carrier. If the SNN emits no spike at a time step, no original discrete point will be reconstructed in that carrier and the original value reconstructed by the PWM algorithm will come from the interpolation between the two closest carriers with spikes as shown in Fig. 9.
- Only one spike at the output carrier. The original PWM based algorithm is applied (Arriandiaga et al., 2020).
- More than one spike at the output carrier. As explained above the PWM based encoding algorithm was not intended to decode more than one spike per carrier. Hence, for this case an extension of the PWM based algorithm is proposed and two alternatives with regards to the loss function are considered:

    1. Decoding only the first spike emitted at the output, and then directly applying the PWM based algorithm. In this case the loss function is the difference between the time instant of the target spike and the time instant of the first spike of the output.
    2. Decoding all the spikes of the output. In this case the loss function is the difference between the average of the time instants of the output spikes and the time instant of the target spike.

In order to analyse these alternatives the Mackey–Glass Time Series Dataset (MGTSD) is used. This dataset is available on UCI repository (Waheeb, 2016) and defined in Kshirsagar, Balakrishnan, and Yadav (2020). MGTSD is commonly used in prediction problems as benchmark dataset. Two SNN are trained following the criteria of both proposals with the hyperparameters described in Table 2a. For this study 4 previous values are selected to forecast one-step ahead and the time-series dataset is splitted into training (70%), validation (20%) and test (10%) sets.

As shown in Fig. 10, decoding with all the spikes introduces noise to the decodification, which leads to an increase in the Mean Square Error (MSE) and Mean Absolute Error (MAE) measures (see Table 2b). To explain this phenomenon in more detail Fig. 11a is shown: when the output yields 3 spikes the PWM algorithm provides 3 discrete decoded points showing that the more number of interpolations, the more noise

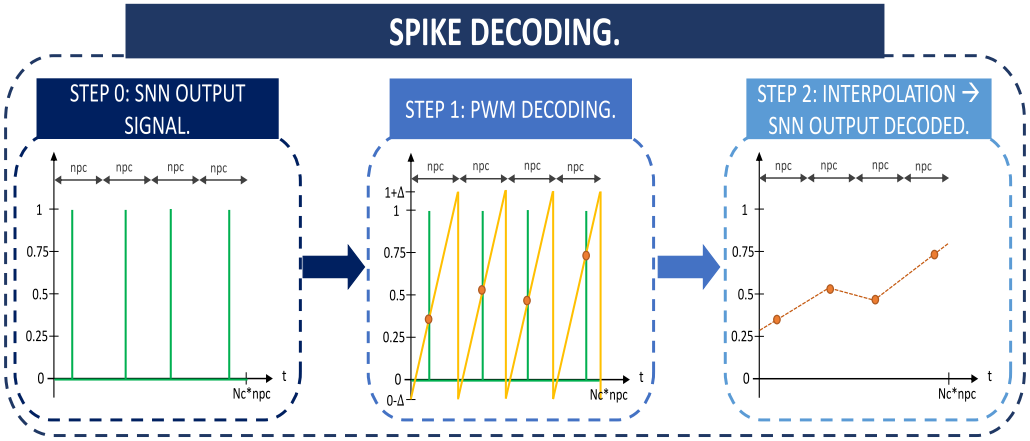


Fig. 8. Spike decoding of the data.

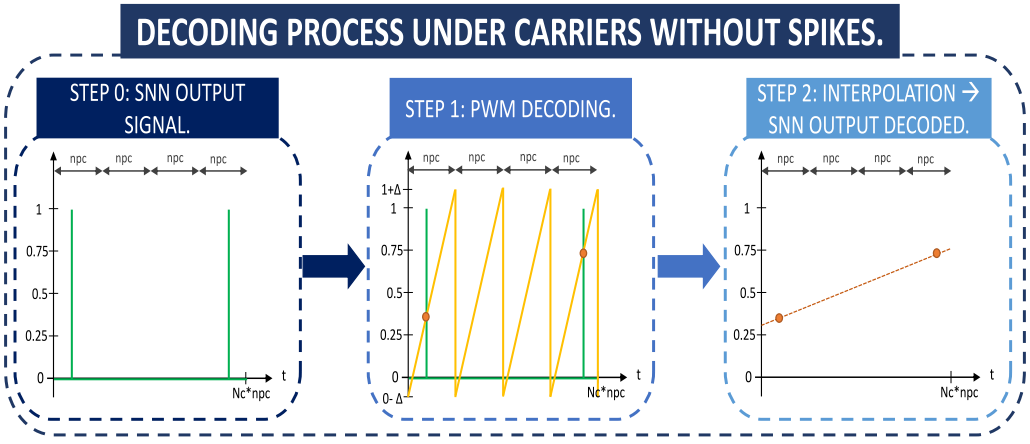


Fig. 9. Decoding process under carriers without spikes.

Table 2
Training hyperparameters in the study to decode with all spike or only one spike.

(a) Training hyperparameters				(b) Forecasting measures		
Neuron model	LIF	PWM algorithm		Decoded with...	MSE	MAE
Tau	100	nc	N-1	... the first spike	0.004	0.047
V_threshold	1	npc	64	... all the spikes	0.005	0.051
V_rest	0					
Learning rate	0.001					

in the decoded signal. This phenomenon disappears if only the first spike is considered as shown in Figs. 10 (left graph) and 11b, yielding lower MSE and MAE (Table 2b). Both MSE and MAE are computed by considering the normalised original signal and the decoded output SNN signal.

Since better results are achieved decoding with the first spike of each time-step carrier an extension of the PWM based algorithm is proposed in order to handle every time-step SNN output. This extension is based on applying a masc ($masc$ = [0 0 1 ... 0 0]) to every carrier with more than one spike detecting the instant in which the first spike is emitted and removing the rest of the spikes from the carrier, as shown in Fig. 11b.

2.3.3. Loss function

In time-series prediction the squared difference between the predicted and the true values is widely used as loss function (Neftci et al., 2019). Hence, the loss function proposed is based on extracting the time instants of both the first spike of the SNN output and the target and computing their difference (see Eq. (8)).

$$E = (\delta_{output_{first\ spike}} - \delta_{target})^2 \tag{8}$$

where δ is the time instant where the spikes are emitted at each carrier. The essential aspect of this method is that the time instant of the spikes is relative to the value of npc. In other words, for each time-step and signal $\delta \in [0, \dots, npc - 1]$.

Given the extension of the PWM algorithm, which removes the case of more than one spike in a time-step, at one time-step SNN output there can only be either no spike or one only spike emitted. Eq. (8) can be directly applied to the case of one spike. However, in the case of no spike emitted there is not $\delta_{output_{first\ spike}}$. In this case the premise considered is that the error should be equal to the total distance from the beginning of the carrier since this is the highest possible error. This can be interpreted as if one artificial spike is introduced at the first position of the carrier and then Eq. (8) is applied. Notice that conceptually speaking this solution should only be possible as long as in the first position of the carrier a real spike is never emitted. To this end, and as explained in Section 2.1.1, the PWM algorithm applies a

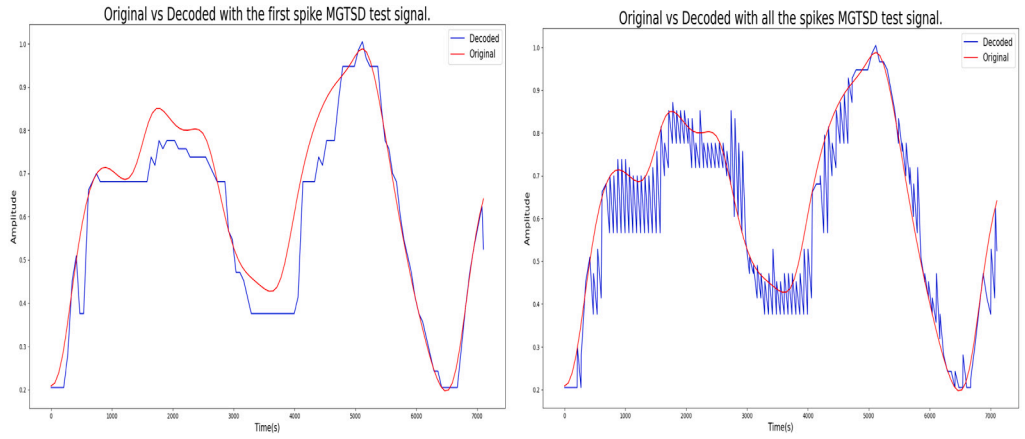


Fig. 10. Difference between decoding with the first spike and all the spikes in each carrier.

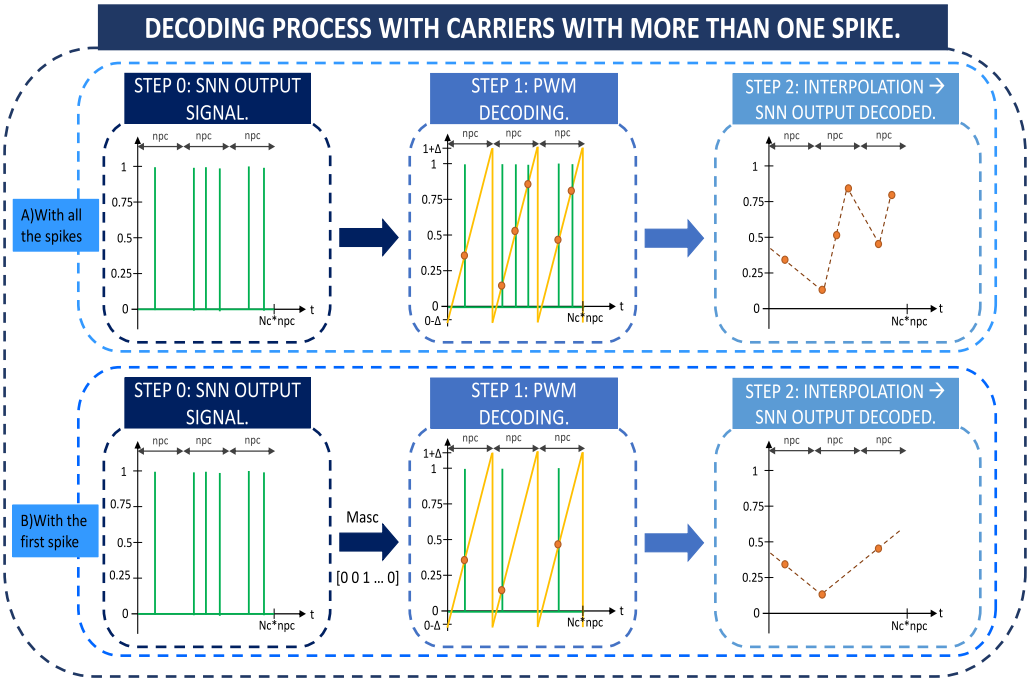


Fig. 11. Decoding process for carriers with more than one spike.

carrier signal whose lower limit is $min - \Delta$. This makes it not possible to emit one real output spike at the first position of the carrier. Thus, with this solution the error can be estimated by Eq. (8) as shown in Fig. 12.

2.3.4. Backpropagation

Once the loss function is calculated the backpropagation phase can be performed in order to update the parameters of the SNN. The weights are updated following the next rule:

$$w_{ij,l} = w_{ij,l} - \eta \Delta w_{ij,l} \tag{9}$$

$$\Delta w_{ij,l} = \frac{\partial E}{\partial w^t_{ij,l}} = \frac{\partial E}{\partial S^t_{i,l}} \cdot \frac{\partial S^t_{i,l}}{\partial H^t_{i,l}} \cdot \frac{\partial H^t_{i,l}}{\partial w^t_{ij,l}} \tag{10}$$

where $w_{ij,l}$ represents the weight between the ith neuron at the layer l and jth neuron at the layer l-1 and η is the learning rate.

In Eq. (10) $S^t_{i,l}$ represents the spike generation function for the ith neuron at the layer. Notice that Eq. (4) ($S^t = \Theta(H^t - V_{th})$) is the general expression for the spike generation process. If the partial derivative of this function is developed:

$$\frac{\partial S^t_{i,l}}{\partial H^t_{i,l}} = \Theta'(H^t_{i,l} - V_{th,i}) \tag{11}$$

$\Theta(x)$ represents the Heaviside step function, whose derivative ($\Theta'(x)$) at the time instant where the spike is emitted tends to infinity making it not possible to use during backpropagation. A BP training with SG is based on using the function $\Theta(x)$ during the phase of samples propagation and a surrogate function $\theta(x)$ during the phase of backpropagation. This surrogate function has a shape similar to $\Theta(x)$ but it is continuous in time, removing the problem of the non-differentiability and making it possible to apply BP. In the proposed methodology the sigmoid function $\theta(x) = \frac{1}{1+e^{-x}}$ is applied as surrogate function.

2.3.5. Early stopping

In order to assess and validate this proposal the training stage has been completed with the addition of an Early Stopping phase in order to

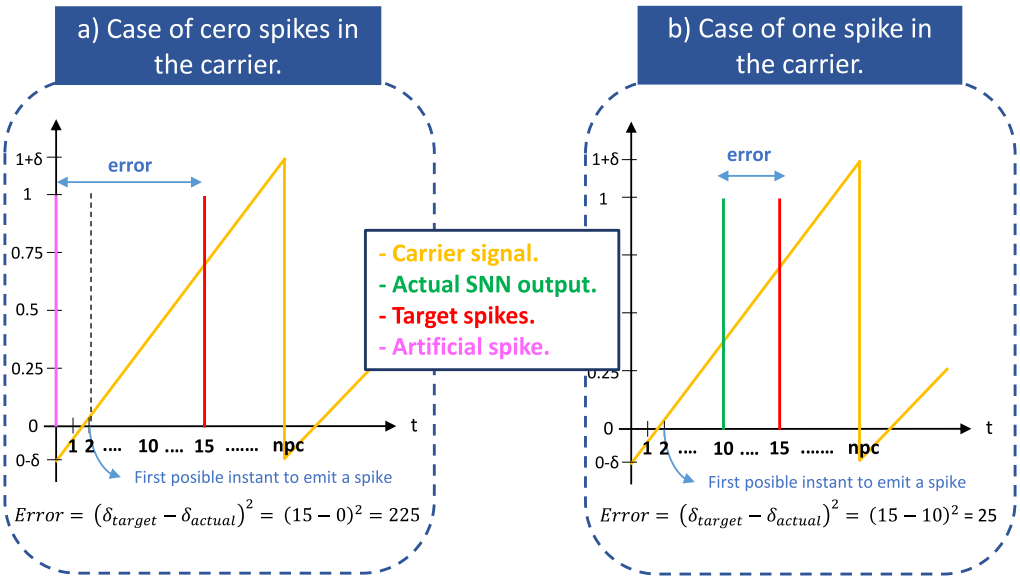


Fig. 12. Loss estimation.

select the best available SNN parameters. At the end of each epoch the validation set is introduced to the SNN so as to calculate the validation error of the SNN with the last updated parameters. This error is based on estimating the MSE between the time instant from the first spike ($\delta_{output_{first\ spike}}$) and the time instant from the target (δ_{target}) at each time-step. The proposed methodology selects the SNN with the lowest validation error.

$$Validation\ error = \sum_{i=1}^{N_{val}} (\delta_{target} - \delta_{output_{first\ spike}})^2 \tag{12}$$

being N_{val} the number of validation samples minus Np.

3. Experimental setup

In this section the validation of the proposed training methodology for SNN is presented. The validation is based on testing the proposed methodology with datasets from different application fields. In this sense the following aspects should be taken into account: simulation design, datasets, hyperparameters assignment and performance metrics formulation.

3.1. Simulation design

The validation is performed on a 3.8GHZ Intel Core i7 processor with a 64-bit Windows 10 Operating System and a 35 GB RAM memory.

3.2. Datasets

In order to assess the forecasting performance of the proposed training methodology for SNN five univariate time-series datasets are considered. As shown in Table 1 the vast majority of the datasets used in the literature review are not available due to (a) the lack of information provided by the authors, (b) the links provided have expired or (c) although the dataset is available, the authors have subjected the datasets to specific changes not explained in detail. In this sense all the datasets applied in this work are public, available and kept unchanged. In addition, the five datasets are from different application fields with different dynamics and characteristics in order to demonstrate that the methodology is independent of the particularities of any specific field.

The first dataset is a sine-wave signal with the following parameters: frequency $f_o = 100$ Hz, amplitude $A = 3$, sampling frequency $f_s = 60 \cdot f_o$ and the number of cycles equal to 20. This dataset is formed by 1200 discrete points. The sine-wave signal is chosen in order to validate the proposed methodology on a periodic dataset without noise whose characteristics are completely known.

The second dataset is the Mackey–Glass Time Series Dataset (MGTSD), which is based on differential equations. The dataset is available in Waheeb (2016) and is formed by 1201 discrete points. MGTSD is commonly used in prediction problems as benchmark dataset.

Since most of the SNN forecasting works presented in Table 1 are focused on the energy application field the third dataset is related to the power consumption of three different distribution networks of Tetouan City. This dataset is available in Salam and Hibaoui (2021) and the power consumption of the zone 3 is selected to validate the methodology. In addition, given the large number of points presented in the dataset data from 00:00 01/01/2017 to 23:50 31/01/2017 is used, which is formed by 4464 discrete points.

Another application field where SNN are also widely used for forecasting is air pollution. The fourth dataset is the PM_{10} hourly monitoring dataset for the Greater London Area, which is available in Department for Environment Food & Rural Affairs (2023) and it is collected by the UK network for air pollution. The selected station is London Bloomsbury, which is part from the Automatic Urban and Rural Monitoring Network (AURN). The selected period is from 2016 to 2017. This dataset is also applied in Maciąg et al. (2019). However, the samples employed in that work are selected randomly from the dataset, which makes it not possible to compare the results. For the validation of the proposed methodology the period from 10:00 21/02/2017 to 20:00 03/06/2017 is selected since it is the largest period in the dataset without loss of data and it is formed by 2459 discrete points.

The fifth dataset is a benchmark dataset based on human voice records from Carnegie Mellon University ARCTIC speech databases, which are available in Black (2003). The ARCTIC datasets are explained in Kominek and Black (2003) and each voice is recorded with a sample rate of 32 kHz. The voice dataset applied in this paper is in the file *"arctic_a0001.wav"* from the subfolder *"US English bdl (made)"*. Due to the large number of discrete points presented in this dataset, the set of samples 11600–14750 is selected.

3.3. Hyperparameters assignment

In this section the hyperparameters for the validation of the proposed methodology are presented. Regarding the LIF model $\tau = 100$, $V_{threshold} = 1$, $V_{rest} = 0$ and $learning\ rate = 0.01$ are used, which are

Table 3
Training hyperparameters for the validation of the methodology.

Training hyperparameters							
Neuron model	LIF	PWM algorithm		Forecasting		Splitting	
Tau	100	nc	N-1	Np	1–10	Training	70%
$V_{threshold}$	1	npc	32, 64, 128	H	1	Validation	20%
V_{rest}	0					Test	10%
Learning rate	0.001						

common values in the literature (Wei et al., 2021). In the PWM based algorithm the values studied in Arriandiaga et al. (2020) are applied: nc is equal to the number values (N) of the time-series minus one, while npc is varied among 32, 64 and 128. In terms of the forecasting problem, in view of the state-of-the-art (Section 1.1) and in order not to increase the computational cost it has been chosen to vary the number of previous values (Np) from 1 to 10. Every time-series is divided chronogically into training (70%), validation (20%) and test (10%) sets. Finally, since the memory of the SNN is controlled from the inputs by introducing the previous samples in the correct order chronogically, all the neurons are resetted to V_{rest} in every time-step (one time-step solution). In Table 3 all the hyperparameters are summarised.

3.4. Performance metrics formulation

Two forecasting measures are selected to assess the forecasting results provided by the methodology. Notice that the precision of the target used for training will depend on the npc hyperparameter of the PWM based algorithm. This means that if a low value of npc is selected to encode the target, the SNN will be trained to learn a signal with less resolution. Thus, in order to analyse the results of the methodology MAE is computed: (i) between the original signal and the decoded SNN output signal (MAE_{o-d}); (ii) between the decoded target signal and the decoded SNN output signal (MAE_{t-d}). These measures can be expressed as:

$$MAE_{o-d} = \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{13}$$

$$MAE_{t-d} = \sum_{i=1}^{N} |\ddot{y}_i - \hat{y}_i| \tag{14}$$

where y_i is the value of the original signal, $\hat{y}_i$ is the value of the decoded SNN output signal, $\ddot{y}_i$ is the value of the decoded target signal (influenced by npc), and N is the number of total points.

4. Results

In this section the results for each dataset are presented. For each dataset are shown: (a) the original time-series and its division into training, validation and test sets; (b) MAE_{o-d} and MAE_{t-d} for npc 32, 64 and 128, and Np ranging from 1 to 10. For selected SNNs, the following results are presented: (c) the weight values distribution; (d) the test original signal, the decoded target and the decoded SNN output; (e) input neurons activation for test samples; (f) output neurons activation for test samples.

4.1. Sine-wave dataset

In Fig. 13 the results for sine-wave dataset are presented. Fig. 13a shows the training (839 samples up to blue line), validation (240 samples from blue to green line) and test (120 samples from green to red line) sets. Notice that this layout is the same for the rest of datasets.

As expected, Fig. 13b shows that MAE_{t-d} is lower than MAE_{o-d}. The values of both errors decrease when npc value is increased. Notice that the sine-wave dataset is the only perfectly periodic time-series without noise. Hence, employing a higher npc involves that SNN is trained with a higher resolution, i.e. more precise values. Another important conclusion from Fig. 13b is that the results improve for higher Np, so as the best result is yielded by Np equal to 5 and npc equal to 128 with MAE_{t-d} equal to 0.

For these hyperparameters 100% of the carriers contain only one spike. Moreover, Fig. 13c shows that 99.64% of its weights are practically zero [−0.04, 0.04] and, thus, they can be considered as inactive weights. The fact that the network has only 0.36% of active weights suggests that the computational cost of using this SNN is very low. In addition, the selected SNN offers a very precise forecasting as shown in Fig. 13d, where the decoded SNN output signal is overlapped with the decoded target being 0 the maximum error between these two signals and 0.0094 between the original and the output signal. The fact that the output signal is overlapped with the decoded target means that the time instant of every spike emitted by the SNN matches the time instant of every target. Hence, this SNN achieves a perfect forecasting so as the only difference between the original and the SNN output is due to the PWM based algorithm.

Finally, Figs. 13e and 13f show the input and output neurons activation patterns, respectively.

4.2. MGTSD

Fig. 14 shows the results for MGTSD. As shown in Fig. 14a training, validation and test sets for MGTSD are formed by 840, 240 and 120 samples, respectively.

In contrast with the sine-wave time-series, Fig. 14b shows that the best forecasting measures are achieved with Np equal to 1. It can also be observed that for $Np = 1$ independently of npc value both MAE_{o-d} and MAE_{t-d} achieve the lowest similar values. Regarding MAE_{t-d}, similar MAE_{t-d} values for different targets (different npc values) suggests that the SNN is capable of learning targets with different resolutions. Actually, in the case of this time-series the target with the lowest resolution is enough to achieve satisfactory forecasting results. This can also be observed by comparing MAE_{o-d} where the original signal is the same one for the three targets.

For Np values higher than 1 the highest MAE_{o-d} and MAE_{t-d} correspond to npc equal to 128. Actually, it can be observed that MAE_{o-d} and MAE_{t-d} are directly related to (a) the increase in the number of carriers that emits more than one spike at each time-step and (b) the distribution of these spikes within the carrier. Fig. 15a shows that for $Np = 1$ and for every value of npc the number of carriers that emits more than one spike is less than 10%. For this value of Np the number of carriers that emits only one spike prevails: 92.44% for npc equal to 32, 98.32% for npc equal to 64, and 99.16% for npc equal to 128. However, for $Np > 1$ and with independence of the value of npc the number of carriers that emits more than one spike rises, which results in an increase in the values of MAE_{o-d} and MAE_{t-d}. In addition, in this dataset the higher the Np, the longer the maximum distance between extreme spikes of every carrier (see Fig. 15b), being the case of npc equal to 128 the worst case. This means that the spikes are less concentrated in each carrier around the target time instant, which has an important impact on MAE_{o-d} and MAE_{t-d} since the less concentrated the spikes are in each carrier, the greater the likelihood for the first spike to be further from the target time instant.

Given the results for this dataset the selected SNN is the one with npc equal to 32 and Np equal to 1. For these hyperparameters 92.43%

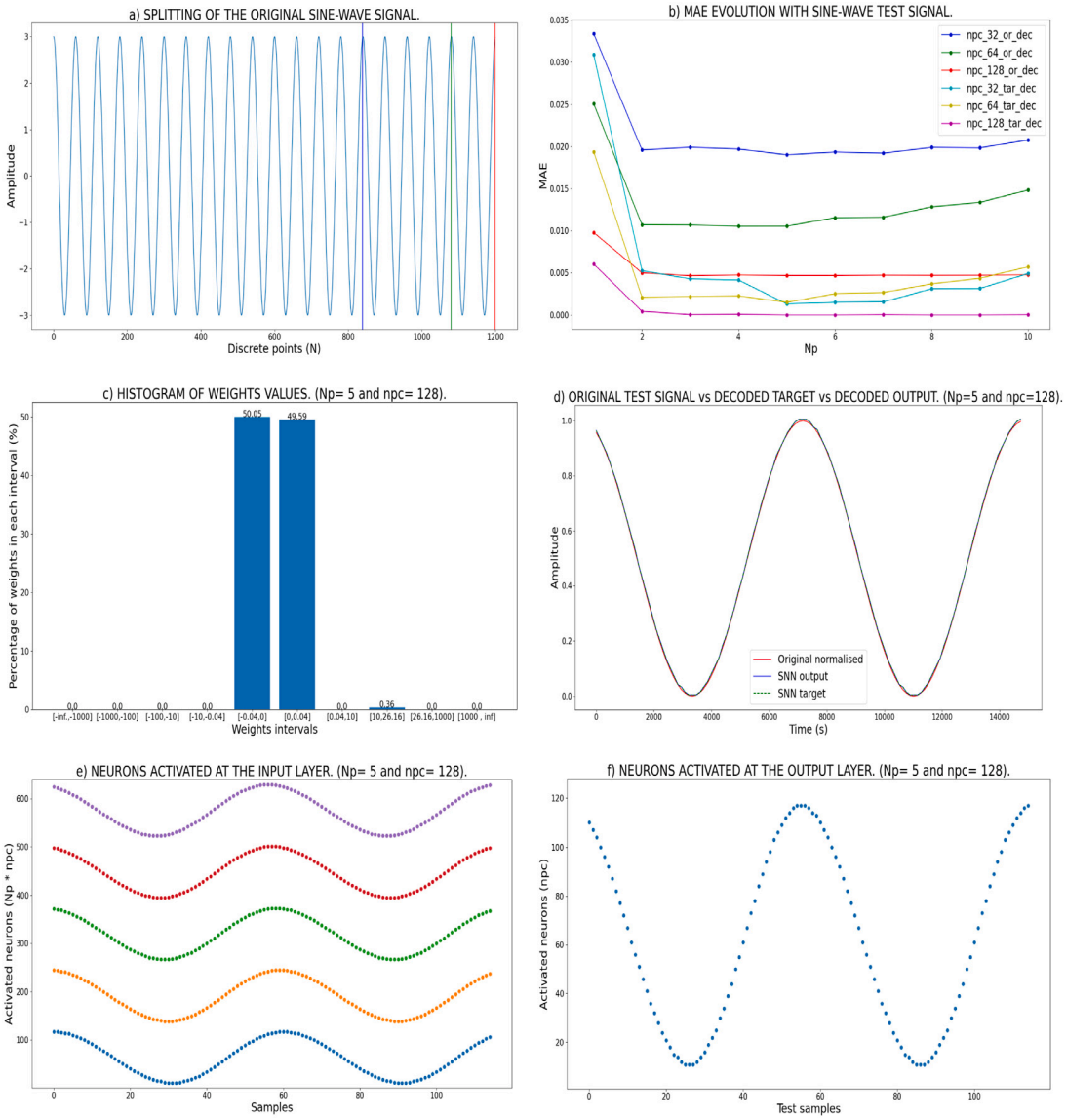


Fig. 13. Results for sine-wave dataset: (a) splitting of the time-series; (b) MAE according to Np and npc; (c) weight values distribution; (d) original test signal, decoded target and decoded SNN output; (e) input neurons activation; (f) output neurons activation.

of the carriers emit only one spike while the rest of the carriers emit two spikes, being 2 time instants the maximum distance between them. Thus, although there are carriers with more than one spike they are highly concentrated around a given time instant and the selection of the first spike should not influence significantly on the results. In addition, Fig. 14c shows that only 9.47% of the weights are active, being 2.34% of the weight values higher than 100. Likewise, in Fig. 14d can be observed that the SNN output in general follows satisfactorily both the target and the original signals, yielding a maximum error of 0.1015 between the original and the decoded SNN output signals and 0.1093 between the decoded target and the decoded SNN output signals. Both maximum errors are achieved in upstream sections of the signal as seen in Fig. 14d.

Finally, both Figs. 14e and 14f reflect that for the selected SNN there is always one neuron activated per sample in the input and output layers.

4.3. Tetouan power consumption dataset

Fig. 16 shows the results for Tetouan power consumption dataset. In this dataset the training, validation and test sets are formed by 3124, 893 and 446 samples, respectively.

Fig. 16b shows that Np equal to 1 provides the best forecasting measures. Regarding MAE_{o-d} for this Np value, npc equal to 64 and 128 offer better results than npc equal to 32. MAE_{t-d} achieves similar values with independence of the npc value. This result suggests that the SNN is capable of learning different targets, however, the resolution of the target for npc equal to 32 is not as accurate as those for npc equal to 64 and 128.

For higher values of Np the lowest values of MAE_{o-d} and MAE_{t-d} correspond to npc equal to 32, followed by npc equal to 64 and 128. As in the previous dataset, this phenomenon is influenced by the number of carriers that emits more than one spike and the distribution of the spikes within the carrier. As seen in Fig. 17a, for Np equal to 2 and 3 the SNN trained with npc equal to 32 emits the lowest percentage of carriers with more than one spike, which results in the lowest values of MAE_{o-d} and MAE_{t-d}. For $Np = 4$ and for every value of npc the percentages of carriers with more than one spike are similar, which leads to similar values of MAE_{o-d} and MAE_{t-d}. For higher values of Np the spikes within the carriers are less concentrated (see Fig. 17b), specially in the case of npc equal to 128. This fact, together with the high number of carriers with more than one spike, influences on the worsening of the SNN results.

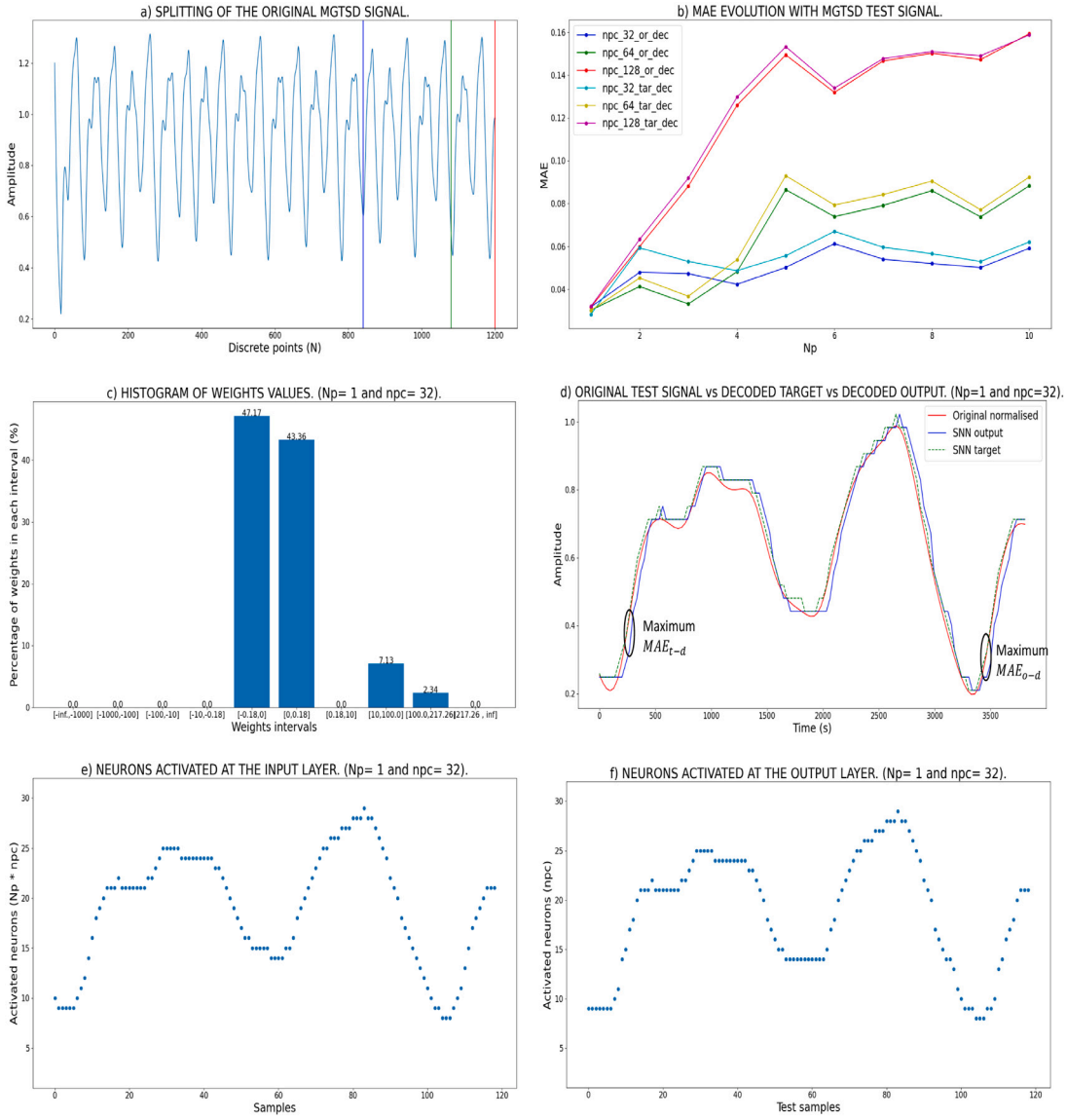


Fig. 14. Results for MGTSD dataset: (a) splitting of the time-series; (b) MAE according to Np and npc; (c) weight values distribution; (d) original test signal, decoded target and decoded SNN output; (e) input neurons activation; (f) output neurons activation.

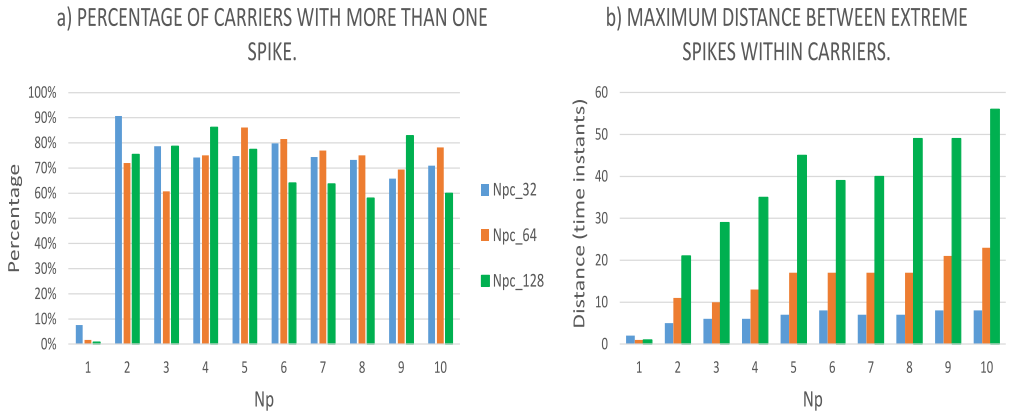


Fig. 15. Number of carriers emitting more than one spike and maximum distance between extreme spikes with MGTSD dataset.

Given the forecasting measures of this dataset the selected SNN is the one with npc equal to 128 and Np equal to 1. For these hyperparameters 92.81% of the carriers emit only one spike, while 7.19% emit two spikes, being 5 time instants the maximum distance between them. Since npc is equal to 128 this distance only entails 0.039 in relative terms, showing a very high concentration of the output spikes around

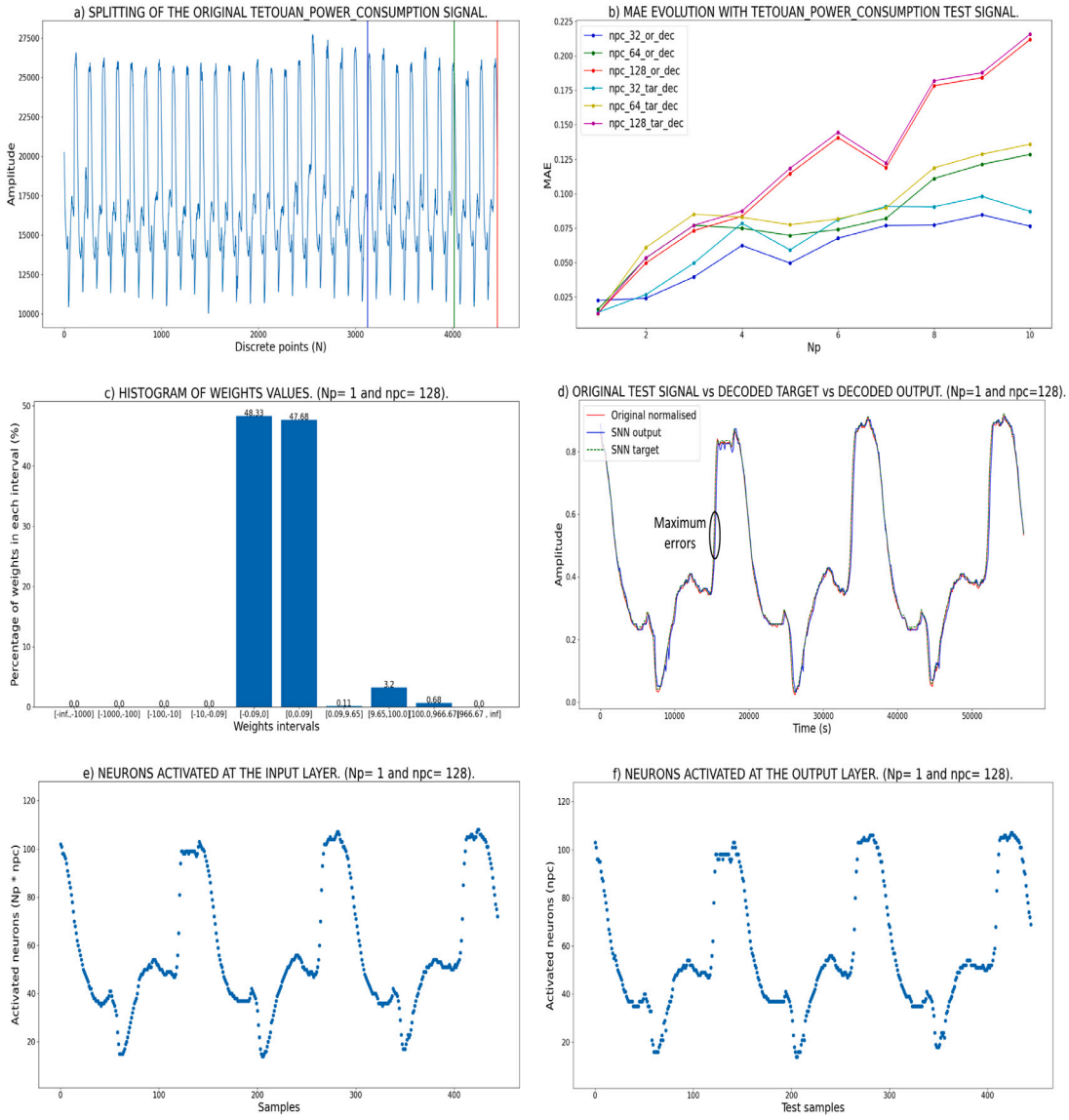


Fig. 16. Results for Tetouan power consumption dataset: (a) splitting of the time-series; (b) MAE according to Np and npc; (c) weight values distribution; (d) original test signal, decoded target and decoded SNN output; (e) input neurons activation; (f) output neurons activation.

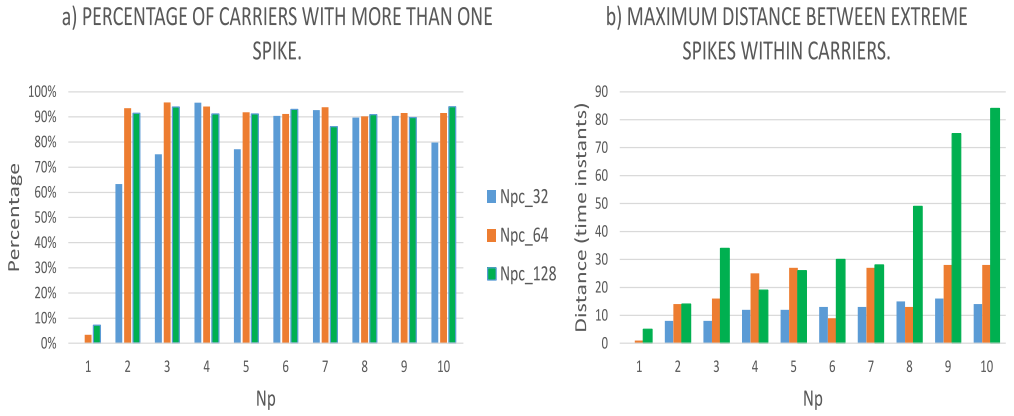


Fig. 17. Number of carriers emitting more than one spike and maximum distance between extreme spikes with Tetouan power consumption dataset.

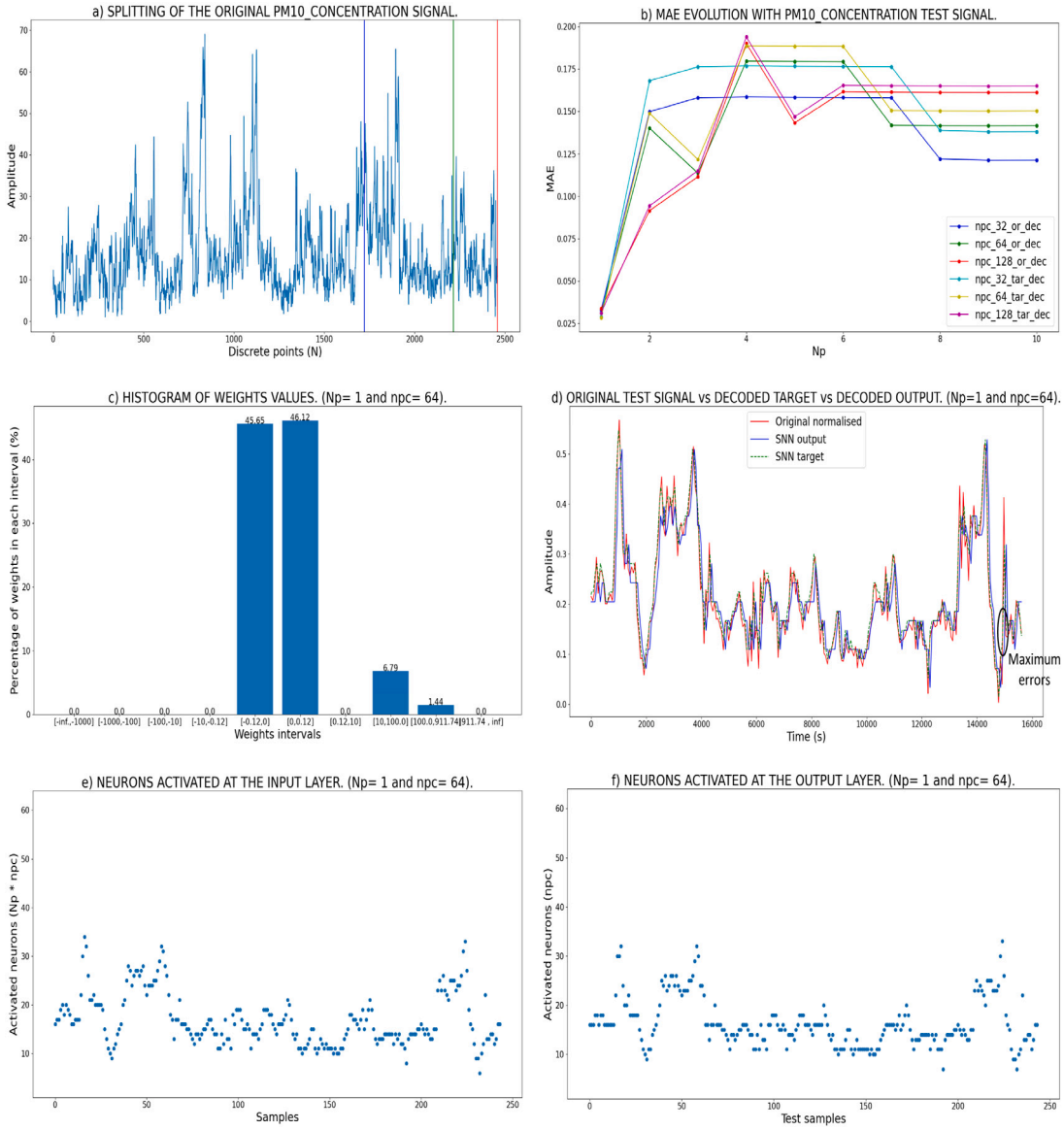


Fig. 18. Results for PM_{10} concentration dataset: (a) splitting of the time-series; (b) MAE according to Np and npc; (c) weight values distribution; (d) original test signal, decoded target and decoded SNN output; (e) input neurons activation; (f) output neurons activation.

the target spike. In addition, Fig. 16c shows that only 3.99% of the weights are active, which is the smallest value among all the considered UCI datasets. Fig. 16d shows the forecasting evolution of this SNN, so as 0.16 is the maximum error between the original and decoded SNN output signals and 0.1596 between the decoded target and decoded SNN output signals. Both errors are practically equal and match in an upstream section of the signal as seen in Fig. 16d.

Finally, both Figs. 16e and 16f show that the SNN activates only one neuron at each time-step.

4.4. PM_{10} Concentration dataset

Fig. 18 shows the results for PM_{10} concentration dataset. In this dataset the training, validation and test sets are formed by 1721, 492 and 245 samples, respectively.

Fig. 18b shows that the best forecasting measures are achieved with Np equal to 1. Both MAE_{o-d} and MAE_{t-d} are slightly lower for the case of npc equal to 64 such that npc equal to 32 causes information loss, while npc equal to 128 includes noise in the learning process.

Fig. 18b also shows that by increasing Np by only one, both MAE_{o-d} and MAE_{t-d} increase substantially. This result can be explained by observing Fig. 19a, where for $Np = 2$ the number of carriers with more than one spike suffers also an important increase for every value of npc. In addition, for Np equal to 2 and 3 the highest percentage of carriers with more than one spike corresponds to npc equal to 32, which leads to higher MAE_{o-d} and MAE_{t-d} compared with npc equal to 64 and 128. For $Np = 4$ the percentages are equal. However, as shown in Fig. 19b for npc equal to 128 there is a peak in the maximum difference between extreme spikes that leads to the highest values of MAE_{o-d} and MAE_{t-d}. For Np values higher than 4 MAE_{o-d} and MAE_{t-d} values are coherent with the number of carriers with more than one spike.

For this dataset npc equal to 64 and Np equal to 1 are selected to show the rest of the results. For these hyperparameters 97.54% of the carriers emit only one spike while the rest of the carriers emit two spikes, being one time instant the maximum distance between them. Fig. 18c shows that 8.23% of the weights are active. In addition, Fig. 18d shows that the SNN is able to follow satisfactorily both the target and the original signals, being 0.3069 the maximum error between

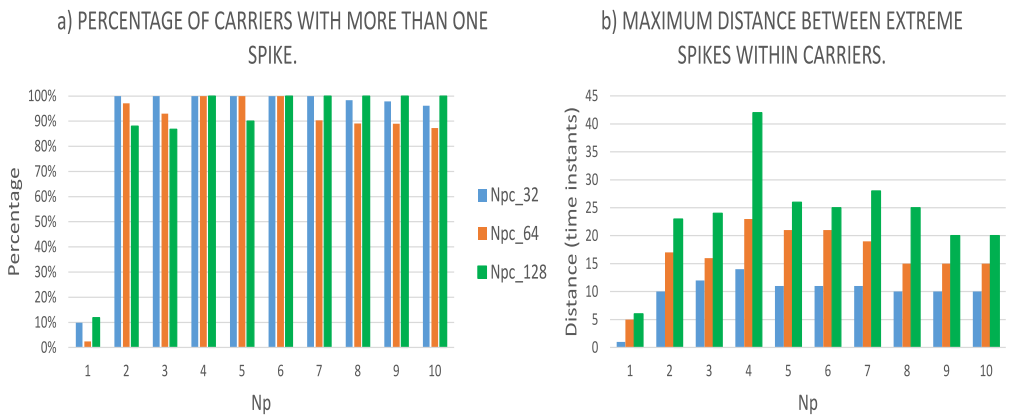


Fig. 19. Number of carriers emitting more than one spike and maximum distance between extreme spikes with PM_{10} concentration dataset.

the original and decoded SNN output signals and 0.1812 between the decoded target and decoded SNN output signals. Both errors match in an upstream section of the signal as seen in Fig. 18d.

Finally, both Figs. 18e and 18f show that for the selected SNN always yields one spike per time-step.

4.5. ARCTIC dataset

Fig. 20a shows the ARCTIC dataset. The training, validation and test sets are formed by 2204, 630 and 315 samples, respectively.

Fig. 20b shows that the best forecasting measures are achieved with Np equal to 1 so as npc equal to 64 yields the lowest MAE_{o-d} followed by npc equal to 32 and 128.

For Np values higher than 1 the lowest MAE_{o-d} is achieved for npc equal to 32 except for Np equal to 5 and 6. In the latter cases the lowest MAE_{o-d} is achieved for npc equal to 64. This circumstance can be explained by Fig. 21a, where for Np equal to 5 and 6 there is a lower number of carriers that emits more than one spike for the case of npc equal to 64 than for npc equal to 32. In addition, on the whole the highest values of MAE_{o-d} and MAE_{t-d} are achieved for npc equal to 128. As seen in Fig. 21b the spikes are less concentrated for npc equal to 128, worsening the forecasting performance.

In the case of this dataset $Np = 1$ and $npc = 64$ are selected as the hyperparameters for the rest of the results. For those values 100% of the carriers emit only one spike. In addition, 11.86% of the weights are active, which is a higher percentage than in previous datasets. The more randomness of this time-series may have influence on this phenomenon. In spite of this, Fig. 20d shows that this configuration of SNN provides satisfactory forecasting results, yielding a maximum error between the original and decoded SNN output signals of 0.0634 and between the target decoded and decoded SNN output signals of 0.0739. Both errors match around a local minimum as seen in Fig. 20d.

Finally, both Figs. 20e and 20f show the neurons activation of the input and output layer, respectively. Notice that there is always one spike per carrier at the output with these hyperparameters.

5. Discussion

This methodology has demonstrated its generality yielding satisfactory results to 5 datasets from different application fields with different dynamics and characteristics.

Regarding the best achieved results, for periodic time-series without noise whose characteristics are completely known (sine-wave dataset) the methodology is sufficiently robust to achieve perfect forecastings. For the 3 UCI and the available real-world datasets it can be concluded that the SNN performance is directly related to the number of output carriers that emits only one spike. An increase in the number of the carriers that emits more than one spike entails a decrease in the SNN learning performance.

The main hyperparameter that enhances the increase in the number of carriers that emits more than one spike is Np. Concerning the sine-wave time-series, it is the only dataset whose best results have been achieved by increasing Np. For the rest of datasets when Np is equal to 1 the number of carriers with more than one spike are less than 10% for every value of npc, while for $Np > 1$ these percentages are increased above 80% as seen in Fig. 22a. Thus, the optimal value of Np for these datasets is 1. This phenomenon may be related with the dynamics of the time-series. Given the results of the sine-wave time-series, which presents a periodic comparatively slow dynamic ($f_o = 100$ Hz) with absent of noise, and the results of the ARCTIC time-series, which is characterised by a much faster dynamic ($f_o = 32$ kHz), it has been observed that time-series with slower dynamics require more Np since more historical data influence the next value to be forecasted. On the contrary, for faster dynamics only 1 Np is enough to adjust to the changes in the time-series.

Another aspect that influences on to the SNN learning performance is the dispersion of the spikes emitted within the carrier. Except for sine-wave, not only the number of carriers with more than one spike becomes higher when Np is increased, but also the maximum difference between the first and the last spike emitted within the carrier is commonly increased. Fig. 22b shows the mean of the maximum difference between extreme spikes within the carrier for these datasets. It can be observed that the mean increases more than double when Np increases from 1 to 2. In addition, for higher values of Np the mean is in general increasingly higher especially for npc equal to 128. This circumstance, together with the fact that the methodology uses the first spike emitted in each carrier, influences on the forecasting measures (MAE_{o-d} and MAE_{t-d}).

Shannon entropy (H) is a statistical measure of the degree of randomness in time-series (Gan & Learmonth, 2015) and it is defined as follows:

$$H = -\sum_{i=1}^{n}(p_i \cdot log(p_i)) \tag{15}$$

where p_i is the discrete distribution of each of the values of the time-series. Table 4 shows the Shannon entropy of the time-series ranked from lower to higher degree of randomness such that sine-wave is the lowest while ARCTIC is the highest. Regarding the PWM based encoding–decoding algorithm, it can be concluded that the value of the hyperparameter npc depends on the particular characteristics of the time-series, especially the degree of randomness. For sine-wave dataset, which possesses the lowest degree of randomness, the best results are achieved with higher npc since the values used for training are more precise. Nonetheless, for time-series with more noise and degree of randomness, increasing npc does not imply necessarily an improvement

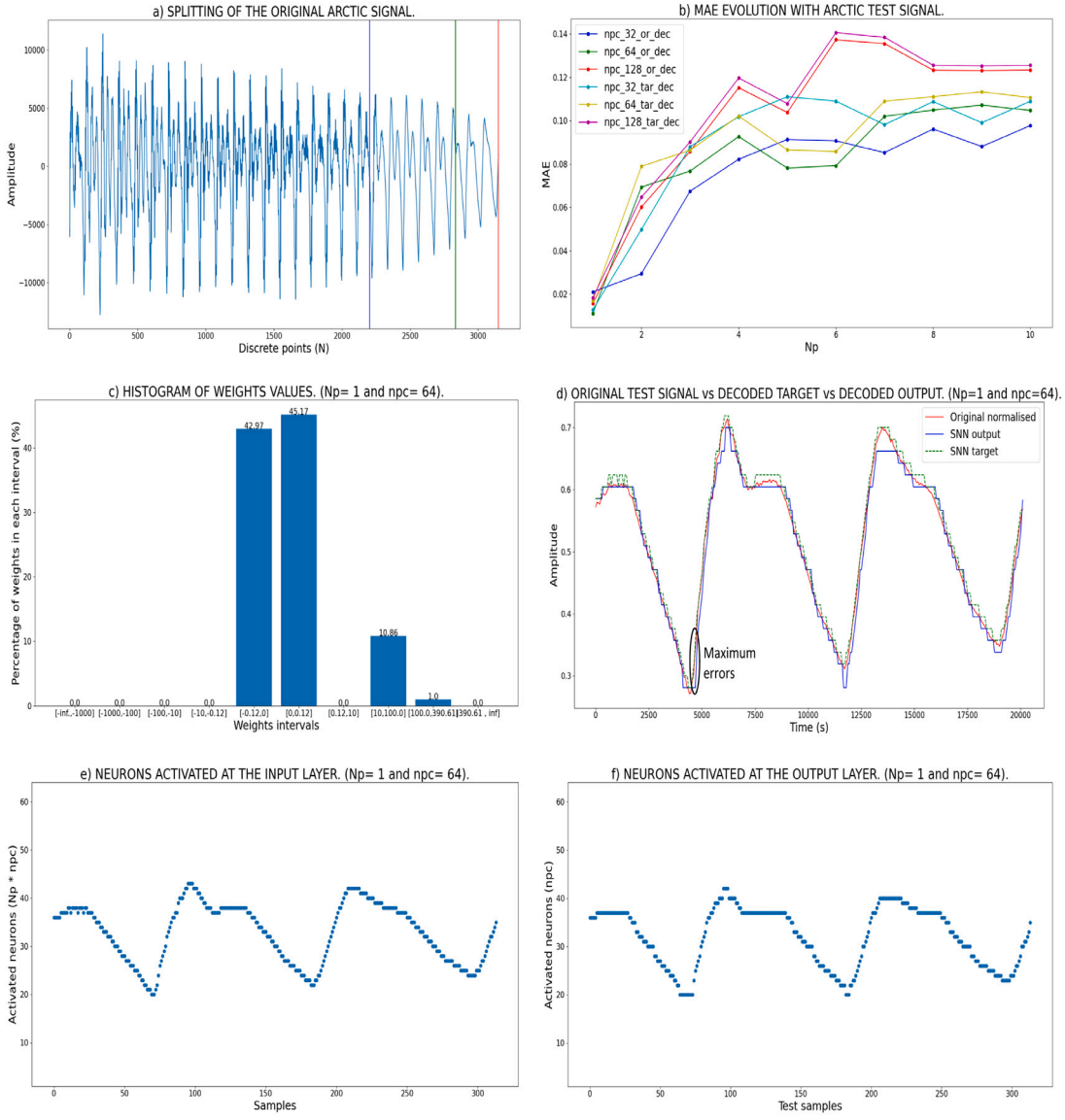


Fig. 20. Results for ARCTIC dataset: (a) splitting of the time-series; (b) MAE according to Np and npc; (c) weight values distribution; (d) original test signal, decoded target and decoded SNN output; (e) input neurons activation; (f) output neurons activation.

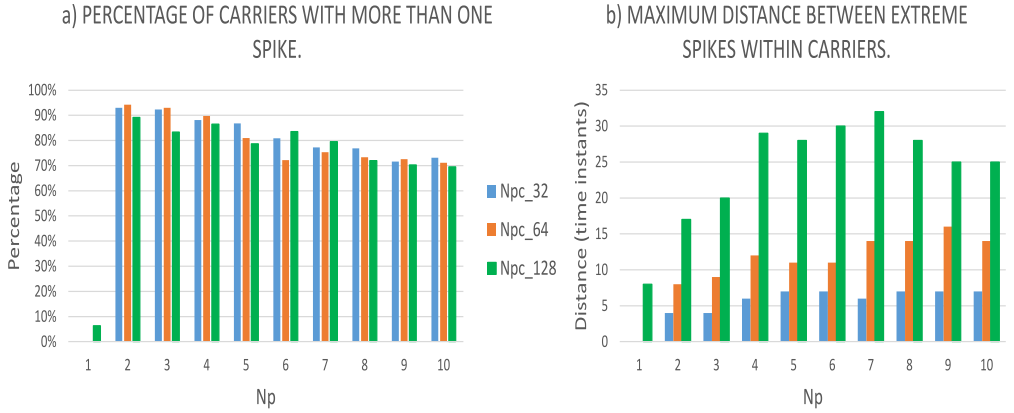


Fig. 21. Number of carriers emitting more than one spike and maximum distance between extreme spikes with ARCTIC dataset.

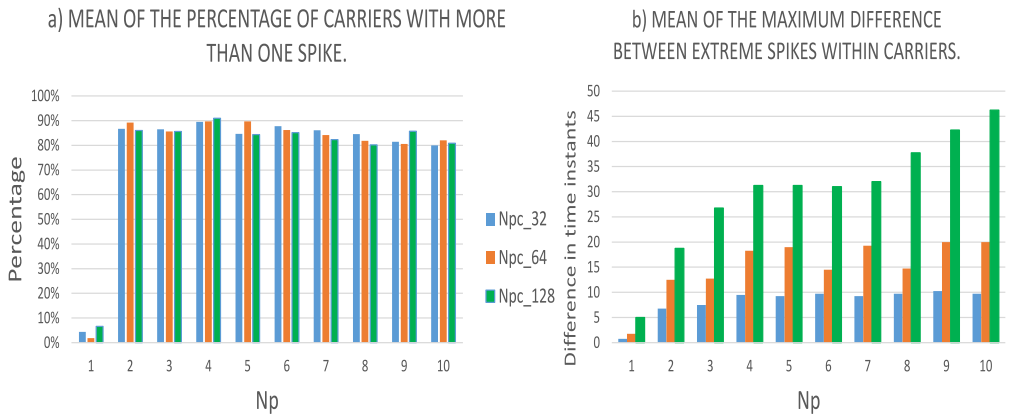


Fig. 22. Mean of the number of carriers emitting more than one spike and maximum difference between extreme spikes with MGTSD, Tetouan power consumption, PM_{10} concentration and ARCTIC datasets.

Table 4
Shannon entropy.

Datasets	Shannon entropy
Sine-wave	6.59
MGTSD	7.09
Tetouan energy	7.29
PM_{10} concentration	7.69
ARCTIC	7.82

Table 5
Shannon entropy and percentage of active neurons for every dataset with $Np = 1$.

Datasets	Shannon entropy	Percentage of active weights		
		npc=32	npc=64	npc=128
Sine-wave	6.59	5.27%	1.37%	0.36%
MGTSD	7.09	9.47%	5.57%	2.80%
Tetouan energy	7.29	11.03%	6.69%	3.99%
PM_{10} concentration	7.69	15.34%	8.23%	4.31%
ARCTIC	7.82	19.34%	11.86%	6.08%

in the learning performance of the SNN as it may result in learning noise and, thus, increasing overfitting. In this sense, in these cases the best solution is to choose a npc that provides a trade-off between the computational cost and the precision required in the problem at hand or in the specific application field.

One of the greatest advantages of the proposed SNN training methodology is the low model complexity. The model complexity in neural networks is usually measured in terms of the topology and architecture of the neural network, the neuron model and the parameters of the learning algorithm itself. Notice that one of the main premises of this proposal is using the simplest model: (i) a feedforward topology; (ii) no hidden layer is used; (iii) LIF model is applied which is a first order system characterised by its simplicity. In addition, all the selected SNN yield a low percentage of active weights, being the worst case the ARCTIC dataset with 11.86% of active weights. A low percentage of active weights further reduces the complexity of the model and boosts the computational cost advantages of the SNN.

It has been observed that the difference in percentage of active weights is directly related to characteristics of the time-series. Table 5 shows the percentage of active weights of every SNN trained with Np equal to 1. Notice that the sine-wave, which is the time-series with the lowest degree of randomness, possesses the lowest percentage of active weights independently of npc value. On the contrary, for time-series with faster dynamic and degree of randomness such as PM_{10} concentration and ARCTIC datasets the SNN need 10% more active weights in order to achieve satisfactory forecasting results.

Moreover, it should be highlighted that in this proposal inference is performed in a single forward pass, i.e. one time-step latency, bringing the advantage of (a) avoiding the memory access cost of accumulated membrane potentials (Chowdhury, Rathi, & Roy, 2021); (b) low-power efficiency since long latency during which many spikes are processed increases the energy costs, i.e. the accumulation of membrane potential over a large number of time-steps results in higher number of operations (Chowdhury, Rathi, & Roy, 2022; Xu et al., 2020); (c) real-time applications friendliness (Chowdhury et al., 2022). In addition, this methodology apply a temporal spike sparsity of $1/npc$. Temporal sparsity is a desirable feature in a spiking model since a lower number of spikes means a lower consumption of energy (as a non-activated neuron will consume no energy) (Dudek et al., 2022).

Another advantage of the proposed methodology is its robustness. As well-known, ANN training is repeated multiple times using different sets of initial conditions in order to expand the search space and achieve the best possible solution. Table 6 shows the MAE_{o-d} measure repeating the training for all the selected SNN 10 times with different randomly initialised weights. It can be observed that for the time-series with less degree of randomness, i.e. sine-wave, MGTSD and Tetouan energy consumption datasets, the proposed methodology is robust enough to converge on the same results, being zero the standard deviation. For PM_{10} concentration and ARCTIC datasets, despite its higher degree of randomness the standard deviations of the results are practically zero, being 0.00013 for PM_{10} concentration time-series and 0.00017 for ARCTIC. Hence, the proposed methodology improves SNN advantages related with computational and energy costs since only one weights initialisation is enough to achieve satisfactory results. Also, for all the selected SNNs the number of carriers that emits no spikes is 0%. This means that in every time-step the methodology provides always one predicted value, which is an essential aspect in forecasting problems.

6. Conclusions

In this paper the design of a new supervised training methodology for univariate time-series forecasting with SNN is presented. This methodology is based on the combination of PWM based encoding–decoding algorithm, which surpasses its predecessor algorithms in terms of precision, and a SG based method, which allows the implementation of the supervised training in SNN. In order to validate the generality of the presented methodology sine-wave, 3 UCI and 1 available real-world datasets have been employed.

Different from the state-of-the-art, the proposed methodology is addressed to any application field with independence of its characteristics, achieving very satisfactory forecasting results in all the considered

Table 6
MAE_{o-d} measure training the selected SNN 10 times with different weight initialisations.

Initialisation	MAE_{o-d}				
	Sine-wave	MGTSD	Tetouan consumption	PM10 concentration	ARCTIC
1	0.0047	0.0318	0.0134	0.0312	0.0112
2	0.0047	0.0318	0.0134	0.0312	0.0112
3	0.0047	0.0318	0.0134	0.0316	0.0112
4	0.0047	0.0318	0.0134	0.0312	0.0112
5	0.0047	0.0318	0.0134	0.0314	0.0112
6	0.0047	0.0318	0.0134	0.0312	0.0112
7	0.0047	0.0318	0.0134	0.0312	0.0116
8	0.0047	0.0318	0.0134	0.0312	0.0116
9	0.0047	0.0318	0.0134	0.0312	0.0112
10	0.0047	0.0318	0.0134	0.0312	0.0112
Mean	0.0047	0.0318	0.0134	0.03126	0.01128
Standard deviation	9.1428E−19	0	1.82856E−18	0.00013499	0.00016865

datasets. Despite the simplicity and generality of this methodology, an important advantage is that depending on the application requirements the user can easily adjust a trade-off between the accuracy of the results and the computation cost by the selection of the hyperparameter npc of the PWM based encoding–decoding algorithm. In addition, the proposed methodology is characterised by ultra-low latency and high robustness in the train phase, enhancing the SNN advantages in terms of computational and energy costs.

Given that the vast majority of works in the literature are focused on applying SNNs to classification problems, the presented methodology is proposed with the aim of being a step-forward in the design and development of SNN for forecasting.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the links to all the datasets employed in the "References" section of the manuscript.

Acknowledgements

This work has been supported by grant IT1726- 22 funded by the Basque Government, grant PID2020-112667RB- I00 funded by MCIN/AEI/10.13039/501100011033, NEUROTIP project funded by Programme Euskampus Missions Euskampus Foundation, and grant PIBA_2020_1_0008 funded by Department of Education of the Basque Government. Open Access funding provided by University of Basque Country.

References

Aghabarar, H., Kiani, K., & Keshavarzi, P. (2023). Improvement of pattern recognition in spiking neural networks by modifying threshold parameter and using image inversion. *Multimedia Tools and Applications*, http://dx.doi.org/10.1007/S11042-023-16344-3.

Arriandiaga, A., Portillo, E., Espinosa-Ramos, J. I., & Kasabov, N. K. (2020). Pulsewidth modulation-based algorithm for spike phase encoding and decoding of time-dependent analog data. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(10), 3920–3931. http://dx.doi.org/10.1109/TNNLS.2019.2947380.

Black, A. W. (2003). CMU_ARCTIC speech synthesis databases. http://festvox.org/cmu_arctic/. (Accessed 07 February 2023).

Bojer, C. S. (2022). Understanding machine learning-based forecasting methods: A decomposition framework and research opportunities. *International Journal of Forecasting*, *38*(4), 1555–1561. http://dx.doi.org/10.1016/J.IJFORECAST.2021.11.003.

Brusca, S., Capizzi, G., Lo Sciuto, G., & Susi, G. (2019). A new design methodology to predict wind farm energy production by means of a spiking neural network–based system. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, *32*(4), http://dx.doi.org/10.1002/JNM.2267.

Bu, T., Ding, J., Yu, Z., & Huang, T. (2022). Optimized potential initialization for low-latency spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence, vol. 36, no. 1* (pp. 11–20). Association for the Advancement of Artificial Intelligence, http://dx.doi.org/10.1609/AAAI.V36I1.19874, arXiv:2202.01440.

Capizzi, G., Sciuto, G. L., Napoli, C., Woźniak, M., & Susi, G. (2020). A spiking neural network-based long-term prediction system for biogas production. *Neural Networks*, *129*, 271–279. http://dx.doi.org/10.1016/J.NEUNET.2020.06.001.

Chen, T., Sun, G., Wei, Z., Li, H., Cheung, K. W., & Sun, Y. (2016). Photovoltaic system power generation forecasting based on spiking neural network. In *Proceedings of the 2015 Chinese intelligent systems conference, vol. 359* (pp. 573–581). Springer Verlag, http://dx.doi.org/10.1007/978-3-662-48386-2_59/TABLES/3.

Chowdhury, S. S., Rathi, N., & Roy, K. (2021). One timestep is all you need: Training spiking neural networks with ultra low latency. arXiv:2110.05929.

Chowdhury, S. S., Rathi, N., & Roy, K. (2022). Towards ultra low latency spiking neural networks for vision and sequential tasks using temporal pruning. In *Computer vision - ECCV 2022. Lecture notes in computer science: vol. 13671*, (pp. 709–726). Springer Science and Business Media Deutschland GmbH, http://dx.doi.org/10.1007/978-3-031-20083-0_42/COVER.

de Abreu, R. S., Silva, I., Nunes, Y. T., Moioli, R. C., & Guedes, L. A. (2023). Advancing fault prediction: A comparative study between LSTM and spiking neural networks. *Processes*, *11*(9), 2772. http://dx.doi.org/10.3390/PR11092772.

de Vries, A. (2023). The growing energy footprint of artificial intelligence. *Joule*, *7*(10), 2191–2194. http://dx.doi.org/10.1016/J.JOULE.2023.09.004.

Deng, L., Wu, Y., Hu, X., Liang, L., Ding, Y., Li, G., et al. (2020). Rethinking the performance comparison between SNNS and ANNS. *Neural Networks*, *121*, 294–307. http://dx.doi.org/10.1016/J.NEUNET.2019.09.005.

Department for Environment Food & Rural Affairs (2023). UK air information resource. https://uk-air.defra.gov.uk/data/. (Accessed 15 March 2023).

Dudek, G., Baczy'nski, D., Baczy'nski, B., Manuel González Sopeña, J., Pakrashi, V., & Ghosh, B. (2022). A spiking neural network based wind power forecasting model for neuromorphic devices. *Energies*, *15*(19), 7256. http://dx.doi.org/10.3390/EN15197256.

Fang, W., Chen, Y., Ding, J., Yu, Z., Masquelier, T., Chen, D., et al. (2023). SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, *9*(40), http://dx.doi.org/10.1126/SCIADV.ADI1480.

Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., & Tian, Y. (2021). Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 2661–2671).

Feng, Y., Geng, S., Chu, J., Fu, Z., & Hong, S. (2022). Building and training a deep spiking neural network for ecg classification. *Biomedical Signal Processing and Control*, *77*, Article 103749. http://dx.doi.org/10.1016/J.BSPC.2022.103749.

Gan, C. C., & Learmonth, G. (2015). Comparing entropy with tests for randomness as a measure of complexity in time series. arXiv:1512.00725.

García-Martín, E., Rodrigues, C. F., Riley, G., & Grahn, H. (2019). Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, *134*, 75–88. http://dx.doi.org/10.1016/J.JPDC.2019.07.007.

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). Neuronal dynamics: From single neurons to networks and models of cognition. *Neuronal dynamics: From single neurons to networks and models of cognition* (pp. 1–577). Cambridge University Press, http://dx.doi.org/10.1017/CBO9781107447615.

Han, C. S., & Lee, K. M. (2021). A survey on spiking neural networks. *International Journal of Fuzzy Logic and Intelligent Systems*, *21*(4), 317–337. http://dx.doi.org/10.5391/IJFIS.2021.21.4.317.

Han, F., Li, R., & Qian, D. (2018). Short-term wind speed forecasting model based on spiking neural network. In *2018 International conference on advanced mechatronic systems* (pp. 359–363). IEEE Computer Society, http://dx.doi.org/10.1109/ICAMECHS.2018.8507102.

Izhikevich, E. M. (2006). Polychronization: Computation with spikes. *Neural Computation*, *18*(2), 245–282. http://dx.doi.org/10.1162/089976606775093882.

Kim, H. (2020). *Design and optimization for 5G wireless communications | IEEE eBooks | IEEE xplore*.

Kominek, J., & Black, A. W. (2003). CMU ARCTIC databases for speech synthesis.

Kshirsagar, P., Balakrishnan, N., & Yadav, A. D. (2020). Modelling of optimised neural network for classification and prediction of benchmark datasets. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, *8*(4), 426–435. http://dx.doi.org/10.1080/21681163.2019.1711457.

Kulkarni, S., Simon, S. P., & Sundareswaran, K. (2013). A spiking neural network (SNN) forecast engine for short-term electrical load forecasting. *Applied Soft Computing*, *13*(8), 3628–3635. http://dx.doi.org/10.1016/J.ASOC.2013.04.007.

Laña, I., Capecci, E., Del Ser, J., Lobo, J. L., & Kasabov, N. (2018). Road traffic forecasting using neucube and dynamic evolving spiking neural networks. In *International symposium on intelligent and distributed computing, vol. 798* (pp. 192–203). Springer Verlag, http://dx.doi.org/10.1007/978-3-319-99626-4_17.

Lee, J. H., Delbruck, T., & Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, *10*(NOV), Article 228000. http://dx.doi.org/10.3389/FNINS.2016.00508/BIBTEX, arXiv:1608.08782.

Lien, H. H., & Chang, T. S. (2022). Sparse compressed spiking neural network accelerator for object detection. *IEEE Transactions on Circuits and Systems. I. Regular Papers*, *69*(5), 2060–2069. http://dx.doi.org/10.1109/TCSI.2022.3149006, arXiv:2205.00778.

Liu, H., Lu, G., Wang, Y., & Kasabov, N. (2021). Evolving spiking neural network model for PM2.5 hourly concentration prediction based on seasonal differences: A case study on data from Beijing and Shanghai. *Aerosol and Air Quality Research*, *21*(2), Article 200247. http://dx.doi.org/10.4209/AAQR.2020.05.0247.

Maciąg, P. S., Kasabov, N., Kryszkiewicz, M., & Bembenik, R. (2019). Air pollution prediction with clustering-based ensemble of evolving spiking neural networks and a case study for London area. *Environmental Modelling & Software*, *118*, 262–280. http://dx.doi.org/10.1016/J.ENVSOFT.2019.04.012.

MacIag, P. S., Kryszkiewicz, M., & Bembenik, R. (2020). Online evolving spiking neural networks for incremental air pollution prediction. In *Proceedings of the international joint conference on neural networks*. Institute of Electrical and Electronics Engineers Inc., http://dx.doi.org/10.1109/IJCNN48605.2020.9206775.

Madhiarasan, M., & Deepa., S. (2016). Long-term wind speed forecasting using spiking neural network optimized by improved modified grey wolf optimization algorithm. *International Journal of Advanced Research*, *4*(7), 356–368. http://dx.doi.org/10.21474/IJAR01/1132.

Matenczuk, K., Kozina, A., Markowska, A., Czerniachowska, K., Kaczmarczyk, K., Golec, P., et al. (2021). Financial time series forecasting: Comparison of traditional and spiking neural networks. *Procedia Computer Science*, *192*, 5023–5029. http://dx.doi.org/10.1016/J.PROCS.2021.09.280.

Mesanza, A. B., Lucas, S., Zubizarreta, A., Cabanes, I., Portillo, E., & Rodriguez-Larrad, A. (2020). A machine learning approach to perform physical activity classification using a sensorized crutch tip. *IEEE Access*, *8*, 210023–210034. http://dx.doi.org/10.1109/ACCESS.2020.3039885.

Nakai, T., & Nishimoto, S. (2023). Artificial neural network modelling of the neural population code underlying mathematical operations. *NeuroImage*, *270*, Article 119980. http://dx.doi.org/10.1016/J.NEUROIMAGE.2023.119980.

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

O'Connor, P., Neil, D., Liu, S. C., Delbruck, T., & Pfeiffer, M. (2013). Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in Neuroscience*, *7*(7), 178. http://dx.doi.org/10.3389/FNINS.2013.00178/ABSTRACT.

Qasim Gilani, S., Syed, T., Umair, M., & Marques, O. (2023). Skin cancer classification using deep spiking neural network. *Journal of Digital Imaging*, *36*(3), 1137–1147. http://dx.doi.org/10.1007/S10278-023-00776-2/METRICS.

Rançon, U., Cuadrado-Anibarro, J., Cottereau, B. R., & Masquelier, T. (2021). StereoSpike: Depth learning with a spiking neural network. http://dx.doi.org/10.48550/arxiv.2109.13751, arXiv:2109.13751.

Reid, D., Hussain, A. J., & Tawfik, H. (2013). Spiking neural networks for financial data prediction. In *Proceedings of the international joint conference on neural networks*. http://dx.doi.org/10.1109/IJCNN.2013.6707140.

Reid, D., Hussain, A. J., & Tawfik, H. (2014). Financial time series prediction using spiking neural networks. *PLoS One*, *9*(8), Article e103656. http://dx.doi.org/10.1371/JOURNAL.PONE.0103656.

Saeedinia, S. A., Jahed-Motlagh, M. R., Tafakhori, A., & Kasabov, N. (2021). Design of MRI structured spiking neural networks and learning algorithms for personalized modelling, analysis, and prediction of EEG signals. *Scientific Reports*, *11*(1), 1–14. http://dx.doi.org/10.1038/s41598-021-90029-5.

Salam, A., & Hibaoui, A. E. (2021). Power consumption of Tetouan city Data Set. https://archive.ics.uci.edu/ml/datasets/Power+consumption+of+Tetouan+city. (Accessed 27 February 2023).

Lopes-dos Santos, V., Panzeri, S., Kayser, C., Diamond, M. E., & Quian Quiroga, R. (2015). Extracting information in spike time patterns with wavelets and information theory. *Journal of Neurophysiology*, *113*(3), 1015–1033. http://dx.doi.org/10.1152/JN.00380.2014.

Sboev, A., Litvinova, T., Vlasov, D., Serenko, A., & Moloshnikov, I. (2016). On the applicability of spiking neural network models to solve the task of recognizing gender hidden in texts. *Procedia Computer Science*, *101*, 187–196. http://dx.doi.org/10.1016/J.PROCS.2016.11.023.

Schuman, C. D., Potok, T. E., Patton, R. M., Birdwell, J. D., Dean, M. E., Rose, G. S., et al. (2017). A survey of neuromorphic computing and neural networks in hardware. arXiv:1705.06963v1.

Semenoglou, A.-A., Spiliotis, E., & Assimakopoulos, V. (2023). Image-based time series forecasting: A deep convolutional neural network approach. *Neural Networks*, *157*, 39–53. http://dx.doi.org/10.1016/J.NEUNET.2022.10.006.

Sharma, V., & Srinivasan, D. (2010). A spiking neural network based on temporal encoding for electricity price time series forecasting in deregulated markets. In *Proceedings of the international joint conference on neural networks*. http://dx.doi.org/10.1109/IJCNN.2010.5596676.

Shi, L., Wank, M., Chen, Y., Wang, Y., Liu, Y., Hector, E. C., et al. (2022). Sleep classification with artificial synthetic imaging data using convolutional neural networks. *IEEE Journal of Biomedical and Health Informatics*, http://dx.doi.org/10.1109/JBHI.2022.3210485.

Suetake, K., ichi Ikegawa, S., Saiin, R., & Sawada, Y. (2023). S3NN: Time step reduction of spiking surrogate gradients for training energy efficient single-step spiking neural networks. *Neural Networks*, *159*, 208–219. http://dx.doi.org/10.1016/J.NEUNET.2022.12.008, arXiv:2201.10879.

Sun, G., Chen, T., Wei, Z., Sun, Y., Zang, H., & Chen, S. (2016). A carbon price forecasting model based on variational mode decomposition and spiking neural networks. *Energies*, *9*(1), 54. http://dx.doi.org/10.3390/EN9010054.

Suradhaniwar, S., Kar, S., Durbha, S. S., & Jagarlapudi, A. (2021). Time series forecasting of univariate agrometeorological data: A comparative performance evaluation via one-step and multi-step ahead forecasting strategies. *Sensors*, *21*(7), 2430. http://dx.doi.org/10.3390/S21072430.

Waheeb, W. (2016). Mackey-glass time series dataset. https://figshare.com/articles/dataset/Mackey-Glass_time_series/4233584. (Accessed 10 January 2023).

Wang, Z., Guo, L., & Adjouadi, M. (2016). Wavelet decomposition and phase encoding of temporal signals using spiking neurons. *Neurocomputing*, *173*, 1203–1210. http://dx.doi.org/10.1016/J.NEUCOM.2015.08.078.

Wang, X., Lin, X., & Dang, X. (2020). Supervised learning in spiking neural networks: A review of algorithms and evaluations. *Neural Networks*, *125*, 258–280. http://dx.doi.org/10.1016/J.NEUNET.2020.02.011.

Wang, H., Xue, W., Liu, Y., Peng, J., & Jiang, H. (2020). Probabilistic wind power forecasting based on spiking neural network. *Energy*, *196*, Article 117072. http://dx.doi.org/10.1016/J.ENERGY.2020.117072.

Wei, D., Wang, J., Niu, X., & Li, Z. (2021). Wind speed forecasting system based on gated recurrent units and convolutional spiking neural networks. *Applied Energy*, *292*, http://dx.doi.org/10.1016/J.APENERGY.2021.116842.

Xu, C., Zhang, W., Liu, Y., & Li, P. (2020). Boosting throughput and efficiency of hardware spiking neural accelerators using time compression supporting multiple spike codes. *Frontiers in Neuroscience*, *14*, Article 498784. http://dx.doi.org/10.3389/FNINS.2020.00104/BIBTEX, arXiv:1909.04757.

Yamazaki, K., Vo-Ho, V. K., Bulsara, D., & Le, N. (2022). Spiking neural networks and their applications: A review. *Brain Sciences*, *12*(7), 863. http://dx.doi.org/10.3390/BRAINSCI12070863.

Yang, L., & Zhongjian, T. (2011). Prediction of grain yield based on spiking neural networks model. In *2011 IEEE 3rd international conference on communication software and networks* (pp. 171–174). http://dx.doi.org/10.1109/ICCSN.2011.6014244.

Yao, M., Zhang, H., Zhao, G., Zhang, X., Wang, D., Cao, G., et al. (2023). Sparser spiking activity can be better: Feature refine-and-mask spiking neural network for event-based visual recognition. *Neural Networks*, *166*, 410–423. http://dx.doi.org/10.1016/J.NEUNET.2023.07.008.

Zamri, N. E., Azhar, S. A., Sidik, S. S. M., Mansor, M. A., Kasihmuddin, M. S. M., Pakruddin, S. P. A., et al. (2022). Multi-discrete genetic algorithm in hopfield neural network with weighted random k satisfiability. *Neural Computing and Applications*, *34*(21), 19283–19311. http://dx.doi.org/10.1007/S00521-022-07541-6/TABLES/21.