Voltage prognosis of PEMFC estimated using Multi-Reservoir Bidirectional Echo State Network

Damien Chanal
FEMTO-ST Institute, FCLAB
Univ. Bourgogne Franche-Comté, CNRS
Belfort, France
damien.chanal@femto-st.fr

Nadia Yousfi Steiner
FEMTO-ST Institute, FCLAB
Univ. Bourgogne Franche-Comté, CNRS
Belfort, France
nadia.steiner@univ-fcomte.fr

Didier Chamagne
FEMTO-ST Institute, FCLAB
Univ. Bourgogne Franche-Comté, CNRS
Belfort, France
didier.chamagne@univ-fcomte.fr

Marie-Cécile Pera
FEMTO-ST Institute, FCLAB
Univ. Bourgogne Franche-Comté, CNRS
Belfort, France
marie-cecile.pera@univ-fcomte.fr

Abstract— **One of the key points in the development of fuel cell technologies is to increase their lifetime. For this, it is necessary to be able to detect a degraded state in order to correct it, but also to be able to estimate the future state, which can facilitate predictive tasks such as control and maintenance. Echo State Networks are good candidates to estimate the future performance of proton exchange membrane fuel cells. They are part of the reservoir computing family and process time series using a neural reservoir. This paper proposes a method to predict the future voltage of a low-temperature proton exchange membrane operating in a steady state. For this purpose, several bidirectional reservoirs are used in parallel which improves the captured dynamics but also reduces the reservoir optimization effort which is currently one of the limitations in the development of Echo State Networks. The forecasting performance is quantified in three different ways: First, to simulate the temporal evolution of a system, the network is trained by varying the size of the training base. This allowed observing that the network can correctly capture the degradation dynamics while having an adaptation capacity when new data are added. Secondly, a comparison with a unidirectional ESN shows that bidirectional reservoirs will more easily capture non-periodic recovery and noise phenomena. Lastly, a comparison with 3 stacked bidirectional long-term memories shows that the proposed approach provided similar results while facilitating the optimization due to its low number of parameters to be set.**

Keywords — Bidirectional Reservoir, Echo State Network, Deep learning, Multi-Reservoirs, PEM Fuel Cell, Prognosis, Reservoir computing

I. Introduction

Fuel cells are good candidates for electricity generation as part of future clean energy. They produce electricity, heat, and water using hydrogen and oxygen with an electrical efficiency of about 50%. They are very useful in several areas such as transportation and stationary sectors. One of the most advanced fuel cell technologies is the Low-Temperature Proton Exchange Membrane (LT-PEM), which can start at temperatures around 0 degrees and operate between 60 and 80 degrees. Currently, one of the obstacles to the development of fuel cells is their limited life span. According to the Department of Energy, USA, the ultimate goal is to increase the lifetime of fuel cells for stationary and transportation applications to 80,000 and 8,000 hours, respectively, under realistic operating conditions [1] (the 2020 goal was 40,000 and 5,000 hours). To achieve and improve these lifetime goals, control, prognostic and diagnostic tools tailored to fuel cell systems should be used to anticipate and enable the correction of any abnormal conditions that may arise. Fuel cell diagnosis can detect a degrading condition, while prognostics aims to estimate the future performance of a system in order to implement preventive measures and evaluate their effectiveness in the context of predictive maintenance. There are two main approaches to prognostics: one is based on the use of a physical model while the other is based on the use of data. Each method has its advantages and disadvantages and in some applications, they can be combined [2]–[6]. Methods based on physical laws are very interesting since they need little to no experimental data (which can be costly in time and/or money). However, for some multi-physics systems, it can be very complicated to set up these models in order to obtain a behavior close to reality, thus favoring the implementation of data-based models. This is particularly the case for fuel cell systems that involve various domains such as electrical, thermal, and fluid mechanics.

This paper presents a data-driven prognosis approach as well as the use of Multi-Reservoirs Bidirectional Echo State Networks (MR-BiESN). The objective is to predict the voltage degradation of a fuel cell operating in a quasi-static state. The main contribution of this paper is to propose an Echo State Networks architecture that reduces the need for user expertise and empirical testing while achieving good performance. The data used are from the 2014 IEEE PHM challenge [7].

This paper is structured as follows: the first part of this paper will be devoted to the presentation and study of Echo State Networks; the second part will detail the approach that was chosen to estimate the voltage degradation of the fuel cell and the results will be presented and discussed in the third part.

II. Echo State Networks

The future state of a variable can be obtained using two approaches. One consists in using a mix between physical model and optimization while the other one is based on neural networks and requires past operating data. One of the best-known neural networks dedicated to prediction tasks is the Long Short-Term Memory (LSTM) network [8]. However, despite good performances, the training time of LSTMs is particularly long which makes re-training tasks complicated. To alleviate this constraint without degrading performances, the Gated Recurrent Unit (GRU), a simpler architecture of LSTM, was developed in 2014 [9]. Despite improvements in RNNs architectures, training of LSTM and GRU use backpropagation through time which requires high computational resources.

A significantly different approach has been developed in the 2000s named Reservoir Computing (RC). RC principle consists of mapping one or more input signals into a high-dimensional computational space (reservoir) containing abundant dynamic transient states. The specificity of RC methods is that reservoirs weights are fixed and only the readout layer is trained. The principle of reservoir computing was developed independently by Jaeger [10] and Maass [11], in the form of Echo State Networks (ESN) and Liquid State Machines (LSM) respectively.

The approach presented in this paper uses a method based on a type of neural network called "Echo State Network" (ESN), therefore the other existing methods will not be described.

A. *Theoretical principles*

According to Jaeger [10], the principle of ESNs is to use an input layer where each neuron receives information. Then, a reservoir of neurons propagates the information in a high dimensional space which improves the separation of the data. The last step is the transmission of the information from the reservoir to an output layer to read the results. The particularity of ESN is its simplicity as well as its low computation time. Indeed, the weights of the recurrent connections are randomly fixed during the generation of the reservoir, and thus only the output weights are trained through linear regression. Fig. 1 shows the architecture of an ESN.

From a mathematical point of view, the equations behind ESN are relatively simple:

Consider that we have an ESN with K input units, N reservoir-internal units, and L output units.

The input weights $\mathbf{W^{in}}$ are collected in a matrix of size $N \times K$. The activation of the input neurons at time "*n*" is represented by: *u(n) = (u1(n), ... (uK(n)).*

The internal weights **Wx** them are collected in a matrix of size N $\times$ N. The activation of the internal neurons at time "*n*" is represented by: *x(n) = (x1(n), ... (xÑ(n)).*

The output weights $\mathbf{W^{out}}$ them in a matrix of size $L \times (K + N + L)$. The activation of the output neurons at time "*n*" is represented by: *y(n) = (y1(n), ... (yL(n))*

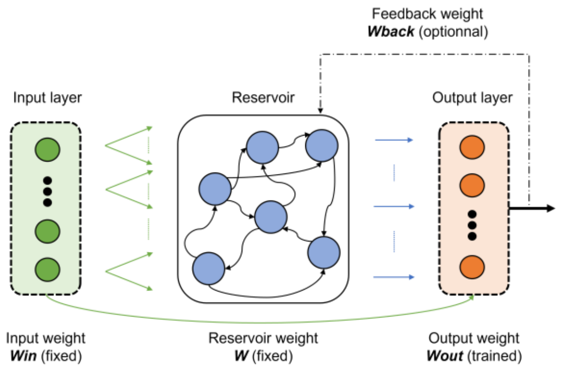

Fig. 1. Schematic representation of an Echo State Network

It is also possible to use a feedback weight matrix $\mathbf{W^{back}}$ of size $N \times L$ that lies between the output weight matrix and the reservoir (optional).

We can calculate the activation of the neurons internal to the reservoir using the equation below:

$$x(n+1) = f\,(\boldsymbol{W^{in}} \times u(n+1) + \boldsymbol{Wx}(n) + \boldsymbol{W^{back}} \times \mathbf{y(n)}) \quad (1)$$

"f" represents the activation function of the neurons and "n" is the time step. With the result of (1), it is possible to calculate the output with (2) described below:

$$y(n+1) = f^{out}(W^{out}(u(n+1),\ x(n+1),\ y(n))) \quad (2)$$

The computation of the output weights is therefore equivalent to solving the linear equation system (3):

$$\boldsymbol{W^{out}} \times X = Y_{target} \quad (3)$$

When training the network, the objective is to minimize the Mean Squared Error (MSE) in (3) which is represented by (4) below:

$$MSE = \textstyle\sum_{n=1}^{r} (\,y(n) - \hat{y}(n)\,)^2 \quad (4)$$

With "r" corresponding to the length of the input/output vectors used and "$\hat{y}(n)$" the true output.

B. *Parameters of ESN*

Still according to Jaeger [10], ESN reservoirs can be defined by 6 main parameters which are: spectral radius, connectivity, leakage rate, number of neurons, scaling factor of input, and feedback weights. These different parameters are detailed below:

From a mathematical point of view, the spectral radius (ρ) corresponds to the maximum eigenvalue of the reservoir matrix. During the initialization of the reservoir, this matrix is randomly generated, and therefore the value of the first spectral radius of the reservoir matrix is also random. In order to let the user, control this parameter, a first solution is to divide the matrix by its spectral radius value (range scaling). This gives a matrix with a spectral radius of 1 and it is sufficient to multiply it by the value desired by the user. Another possibility is to use the Euclidean norm as an upper bound for weights (norm scaling). Matrix weights are divided by their Euclidean norm and the result is then multiplied by the desired spectral radius. It is recommended to use a spectral radius value less than 1 to respect the echo state property which implies that the state of the reservoir should only depend on the input signal and not on the initial conditions that existed before this input (the initial conditions should progressively disappear with time). From a more practical point of view, the spectral radius allows determining the speed at which the influence of the previous inputs and states is reflected in the current states: the higher value of spectral radius, the higher interactions with the past states.

The second parameter to define an ESN reservoir is the leakage rate (α). The leakage rate is in the interval [0, 1] and allows adjusting the dynamics of the neurons (i.e., observing the importance of the previous outputs of the reservoir when calculating the current activations). The neurons using a leaky rate are called "integrators" (Leaky integrator neurons). The higher the leakage rate, the less the previous outputs have an impact on the current outputs). Equation (5) below shows how the integration of integrating neurons changes the calculation of neuron activations presented in Equation (1).

$$x(n+1) = (1 - \alpha).\, x(n) +$$
$$\alpha\, .\, f\,(\boldsymbol{W^{in}} \times u(n+1) + \boldsymbol{Wx}(n) + \boldsymbol{W^{back}} \times \boldsymbol{y(n)}) \qquad (5)$$

The third parameter is connectivity (*c*). It represents the percentage of non-zero weights in the reservoir matrix. According to Jaeger, the idea is to obtain a reservoir rich in dynamics (i.e. not homogeneous). For this, one solution is to build a reservoir with random and weakly connected weights. This decoupling into subnetworks allows the development of individual dynamics. According to Lukoševičius [12], the impact of connectivity on the results is relatively small. However, a sparsely connected reservoir improves computation times (as the reservoirs are updated faster).

The last parameter of the ESN reservoirs is the number of neurons, which can range from a dozen to a few thousand because the calculation is a linear regression and therefore a relatively low computation time compared to other methods (for example, Long Short-Term Memory).

The two last parameters are the scaling factors of the input and feedback weights. Also according to Lukoševičius [12], in the case of uniform input weight initialization, the scaling factor "b" is defined as the range of the interval [-b, b] (or the standard deviation for a normal distribution). As specified by authors Lun and Jaeger in references [13], [14], the scaling factors determine the level of nonlinearity of the tank's response. In his work, Jaeger [15], explains that in the case of a hyperbolic tangent activation function, the closer the factor is to 0 the more linear the response will be, and conversely, the larger the factor the more binary (-1 or 1) the neurons' outputs tend to be.

C. Specific cases

The bidirectional reservoir: A possible alternative to the classical ESN reservoirs is to double the number of reservoirs to make them learn the sequences both in the chronological direction (from left to right) but also in the opposite direction (right to left). The idea behind this modification is that the bidirectional ESN (BiESN) can learn the future but also the past of each data composing the learning sequences. Indeed, with a unidirectional ESN, we only learn from one or the other depending on the desired reading direction. A bidirectional reservoir thus allows increasing the dynamics captured within the learned sequences and improving their interpretation by the algorithm. Bianchi [16] uses a BiESN combined with a neural network for time series classification. Fig. 2 below shows the architecture of a BiESN reservoir:

Multi-reservoir ESN: Another alternative to improve ESNs is to use not only one tank but several in parallel and/or in series. This alternative allows using tanks of different configurations at the same time, instead of one well-optimized tank. This allows alleviating the ESN tank optimization step which can be long and complex. Sun [17] summarizes the main existing architectures of ESNs. Fig. 3 shows a representation of a multi-reservoir ESN where reservoirs are used in a parallel configuration (also called grouped ESN).

III. PROGNOSIS APPROACH DESIGNED

As previously explained, the objective of the developed algorithm is to predict the voltage of a fuel cell based only on a database. For this purpose, the network will train itself (i.e. optimize its weights) using training data and once the training is done, it will predict the future voltage using the predicted voltage at time "*t*" as input to determine the voltage at time "*t+1*", in an iterative way. The prediction is usually done over a defined time horizon or until an end-of-life threshold is reached (Remaining Useful Life - RUL). One of the advantages of neural network-based approaches is the possibility to re-train the network by adding new data. In the context of an embedded application, this would correspond to predicting the evolution of a system after a few hours of operation and re-training the model periodically so that it updates itself using the information acquired since the last prediction and adapts its degradation dynamics to the new data.

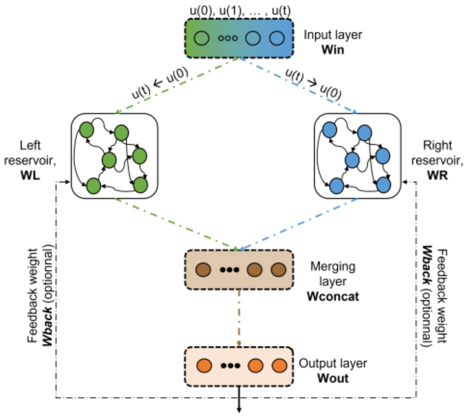

Fig. 2. Architecture of a bidirectional ESN

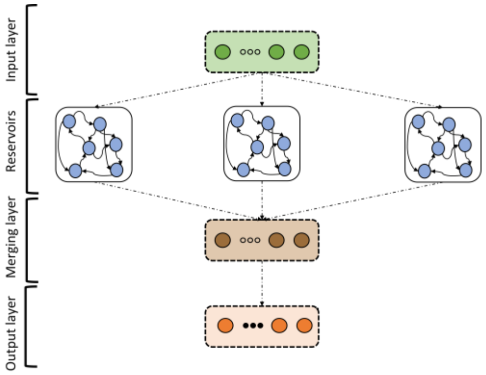

Fig. 3. Scheme of a multi-reservoir ESN with parallel architecture

A. *The network*

Since the speed of execution is one of the criteria for an industrial application, ESNs appear to be a suitable choice for this task. Moreover, using several tanks facilitates the implementation of the network since the number of hyper-parameters to be determined empirically or by optimization is greatly reduced. In the developed approach, the generated network is composed of three bidirectional tanks used in parallel. The parallel arrangement makes it easier to capture the dynamics of the data (low, medium, or fast) because each reservoir is specialized for a single dynamic, but also to dispense with an optimization algorithm to determine the ideal parameter set. As explained in section 2, the fact of using the reservoirs in a bidirectional way improves the learning of the sequences; the network can learn in both temporal directions because there is no causal relation in a neural network between the input and the output thus no time arrow. The selected hyper-parameters of the developed networks are described below and summarized in Table I.

Each of the 3 reservoirs composing the network is specialized for a type of dynamics. (slow, average, fast). These dynamics are represented by parameters of spectral radius and leakage rate more or less large according to the desired dynamics. The connectivity remains constant at 10% in order to accelerate the update of the neurons and the number of neurons is fixed at 100. The input weights are initialized with a normal distribution centered on 0 and with a standard deviation of 0.5.

In the presented study, the data did not undergo any processing (filtering, standardization ...). Indeed, even though, standardization can be useful to accelerate and promote the convergence of optimization algorithms based on gradient descent. Data standardization is necessary when several features at different scales are used. Indeed, the main objective of standardization is to generate a common scale for several features in order to help gradient descent or distance-based algorithms. When dealing with univariate non-stationary time series, according to the hyper-parameters of the neural networks, it is not always necessary because only one feature is used. In this study, given that the data are quite small, not very dispersed, and that the training of the network will be done by simple linear regression without using backpropagation through time, we have chosen not to standardize the data.

Because of the non-scaling of data, the activation function, chosen is the GELU function developed by Hendrycks and Gimpel [18]. Using the classic hyperbolic tangent (TANH) function with non-scaled data can therefore lead to problems when calculating the gradient (known as "vanishing gradient") because it is bounded in the interval [-1, 1]. An alternative would have been to use the RELU function which is the default activation function of feed-forward neural networks. However, the GELU function is recent and not much used at the moment, so it is interesting to measure its performance for time series prediction.

The selected optimization algorithm is the one presented by Loshchilov and Hutter [19], in which the Adam optimizer is coupled with weight decay (also called AdamW). The weight decay is a regularization parameter that improves the generalization of the model by limiting overlearning problems. The learning rate of the optimizer is set to 1×10^{-4} and the weight decay to 5×10^{-3}.

TABLE I. TABLE SUMMARIZING THE DIFFERENT PARAMETERS DEFINED IN THE MR-BIESN

Parameters	Values	Comments
Spectral radius (ρ)	0.1 / 0.5 / 0.9	Low / Medium / High dynamic
Leaky rate (α)	0.1 / 0.5 / 0.9	Low / Medium / High dynamic
Connectivity (*c*)	0.1	-
Number of neurons	For each reservoir : 100 Total : 600	-
Reservoirs initialization	Normal distribution centered on 0 and standard deviation of 0.5	-
Reservoir weight scaling method	Range scaling	-
Readout layer initialization	Glorot Uniform	-
Merging mode of bidirectional reservoirs	Concatenation	-
Activation function	GELU	-
Optimization	AdamW	-
Learning rate	1×10^{-4}	-
Weight decay	5×10^{-3}	-
Batch size	8	-
Sequences length	2/3 of learning database	-
Rate of data dedicated used to validated	20 %	-
Objective funcion	Mean square error	-

A sequence of samples from the database is used to predict the next value. The length of the sequence is chosen empirically at the last 2/3 of the values available at time *k* to predict the value at time *k+1*. This allows considering a significant part of the previous data in order to predict the future, this favors the capture of the long-term dynamics. An alternative would have been to keep the same number of values in a sequence all the time, regardless of the number of samples in the database. However, given that the first predictions are made from the first days of the system's life, thus with a reduced quantity of data (and therefore small sequences), the predictions made after several hours/days/months of operation only consider the events included in the short sequence and not the older ones which allow a better estimation of the trend.

B. *Database*

Data used to train and validate the network comes from the 2014 IEEE PHM challenge [7]. A 1kW LT-PEMFC on which long-term tests (1000 hours) have been performed in quasi-static, the fuel cell is subjected to a constant current density (0.7 A/cm²). The acquisition frequency of the voltage in this database is about 1 Hz. However, it was chosen to select 1 point every 600 seconds. This allows both improving the computation time but also capturing only the aging trend of the fuel cell. It is not necessary to use all the data if the objective is to predict a trend.

IV. Results & Discussion

Once the network has been trained with the indications given in section III, it is necessary to define good metrics to determine whether or not it is capable of predicting the future voltage of the fuel cell. For a more refined evaluation, it is preferable to adopt several metrics. In this study, the ones chosen are the Mean Square Error (MSE), which is commonly used due to its simplicity, and the Mean Absolute Percentage Error (MAPE), which has the advantage of being robust to extreme values and insensitive to data scaling. However, even if these metrics are good indicators to estimate the prediction quality, the network optimization process is stochastic due to the random initialization of the weights. It is therefore necessary to perform the prediction several times in order to obtain a distribution of possible results. This provides a clearer picture of the repeatability and reliability of predictions. The study in this section aims to evaluate three performances. The first concerns the quality of the prediction. For this, the network is first trained with a low training base and this base is progressively increased. Then, a comparison with a classical ESN is performed to demonstrate the usefulness of the bidirectionality. Finally, another comparison is performed with an 3 stacked bidirectional LSTM (BiLSTM).

For each performance measure, the predicted voltage will be estimated 10 times.

A. Forecast performances with an increasing database

Fig. 4(a) and Fig. 4(b) show respectively the voltages predicted by the network using 30% and 70% of the data for training. In order to improve visibility, only the median curves are shown, as well as those of the 1st and 3rd quartiles. It is possible to observe, especially on the Fig. 4 (a), that for a short prediction horizon, the 3 voltages represented are very close before they start to diverge at the 150*600s time step. This phenomenon is more difficult to see in Fig. 4(b). Always on Fig. 4(a), it can be observed that the median and the 3rd quartile are close, however, the curve of the 1st quartile is significantly lower. This difference can be explained by an ill-adapted regularization parameter, this one is fixed for all trainings, so it is possible that it is not optimal during the re-training of the model thus generating some variations in the repeatability of the prediction. Moreover, results shows that the network can assimilate and reproduce correctly the dynamics of degradation even if it is not able to predict the recovery phenomena which are caused by an external action of the user or the control algorithm to limit the degradation.

Fig. 5 represents the evolution of the metrics as a function of the size of the learning base used. The metrics are represented in box plots that highlight the distribution of the prediction errors obtained. The mean and median values are respectively represented by a triangular shape and a line in each box. It is possible to observe that as the size of the training increases, the error decreases but also the dispersion (especially the interquartile range) decreases. Some outliers

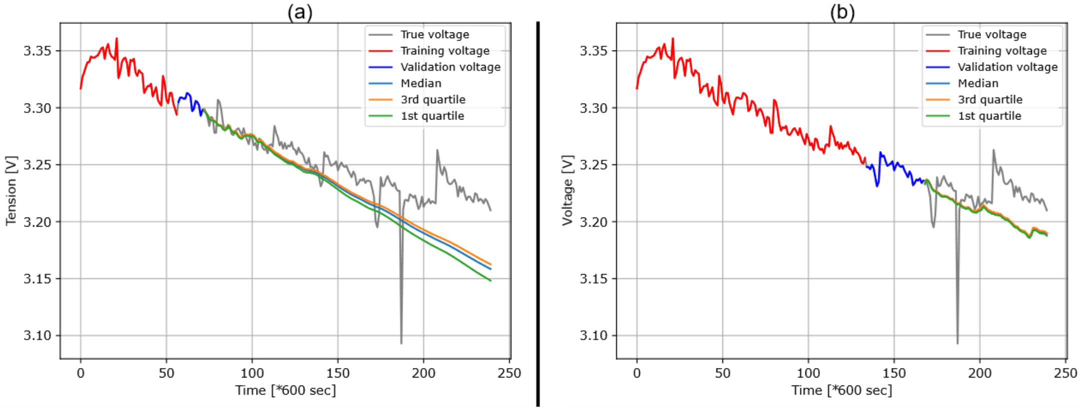

Fig. 4. Voltage predicted by the MR-BiESN using 30% (a) and 70% (b) of the data for training

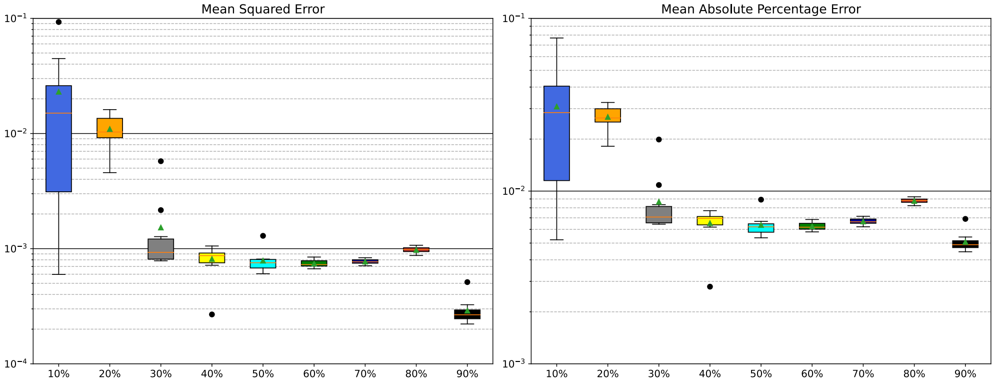

Fig. 5. Measurement of MSE (left) and MAPE (right) prediction errors after 10 iterations as a function of the size of the learning base

are however present and can be explained by the non-optimal setting of the regularization parameter leading to limited repeatability. The presence of outliers further justifies the need to perform several network trainings to reduce the risk of bad predictions. Another point that can be observed in Fig 4 is that when the percentage of training data is close to 70-80%, the errors increase slightly while the dispersion decreases. The opposite phenomenon occurs for a training percentage of 90%. This can be explained by the fact that at 80% and 90% training, the data are just before and after the energy recovery (visible in Fig 4 around time step 210). It is impossible for the network to predict such a recovery which explains the increase in error, and it is not surprising to observe an increase in the dispersion of the errors just after the recovery, as the network needs more post-recovery data to better capture the new dynamics.

B. Comparison with a unidirectional ESN

In order to observe the impact of bidirectionality, a comparison with a unidirectional ESN is done. To compare with the same number of neurons, each one-way ESN will use 200 neurons (because bidirectional reservoirs have 100 neurons in each direction). Fig. 6 shows the voltage forecasting results obtained by each network. Training steps have been done using 50% of the data and others parameters are the same as described in section III.

According to Fig. 6 and Table II, it can be observed that both architectures capture the dynamic of voltage degradation. When using only the one-way ESN the predicted voltage doesn’t contain any recovering phenomena or noise which are present with the MR-BiESN. Unidirectional learning in chronological direction will tend to filter out these patterns while bidirectional learning will tend to integrate them since it allows giving importance to weights located at the beginning but also the end of the learned sequences. It is also possible to observe that the metrics of the unidirectional network seem slightly better than those of the bidirectional network. This difference can be explained by a non-optimal regularization of the model. Indeed, as explained previously, the weight decay parameter selected is fixed and has not been optimized for the different tests. In this case, it seems that the parameter is better adapted for the unidirectional network than for the bidirectional one, even if the two results obtained are very close. Lastly, it is also interesting to note that the bidirectional network captures the rather linear dynamics of degradation throughout the prediction. In the unidirectional network, it seems that the degradation is in the form of a negative exponential, which is particularly visible on the 1st quartile curve. The impact is fairly negligible for a short- to medium-term prediction, but it could skew the results of a long-term prediction.

TABLE II. SUMMARY OF METRICS OBTAINED FOR MR-BIESN AND ONE WAY MR-ESN APPROACHES

	MR-BiESN		**One Way MR-ESN**	
	MSE	***MAPE***	***MSE***	***MAPE***
Median	7,55E-04	6,21E-03	4,15E-04	3,84E-03
1st quartile	6,79E-04	5,78E-03	3,76E-04	3,59E-03
3rd quartile	8,06E-04	6,46E-03	5,87E-04	4,88E-03
IQR	1,27E-04	6,81E-04	2,11E-04	1,28E-03
Mean	7,87E-04	6,36E-03	4,83E-04	4,21E-03
Standard deviation	1,90E-04	9,85E-04	1,70E-04	9,97E-04

C. Comparison with stacked BiLSTM network

One of the objectives of MR-BiESN is to provide good prediction results by limiting the empirical search for hyperparameters. In order to validate the developed network, it is interesting to compare it with BiLSTM which is a well-known approach used for time series forecasting. The comparative network is composed of 3 stacked BiLSTMs in series with 100 neurons for each direction. They are stacked to improve performance and divide the problem into several layers. The same number of neurons was applied to have a similar architecture between the two networks however the weight decay parameter has been decreased at 5×10^{-4} to obtain good performances. During the first simulations, the BiLSTM network associated with the GELU activation function and the non-normalization of the data showed high sensitivity to the gradient explosion phenomenon. This could be explained by the high number of trainable parameters needed to optimize the BiLSTM (725 401 parameters) in contrast to the MR-iESN (601 parameters). To solve this problem, we normalized the training data with the “tanh estimator” method proposed in [20] and changed the

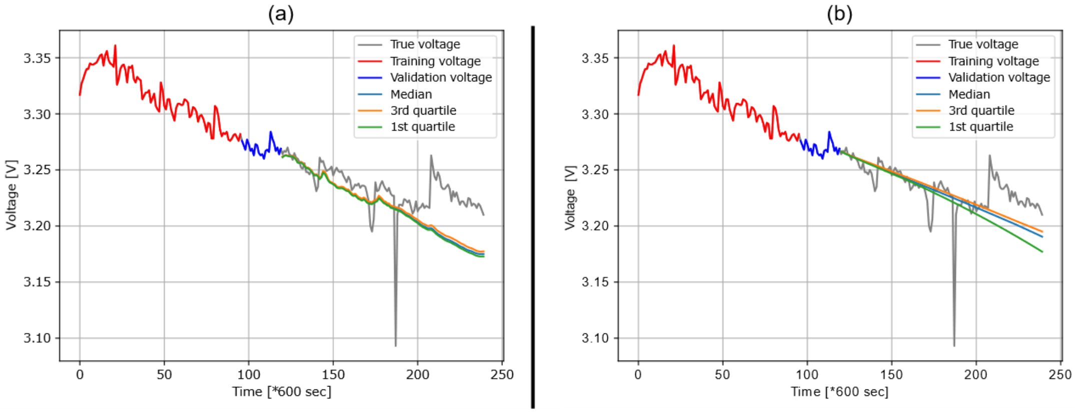

Fig. 6. Comparison of voltage predicted by the MR-BiESN (a) and One way MR-ESN (b) networks using 50% of data for training

activation function GELU to TANH. Indeed, according to the study presented in [21] tanh estimator shows better performances in time series forecasting tasks than the common standardization methods and the TANH is the common activation function used with LSTM. Moreover, the standardization of the data also allows the designed network. In this way, the two networks are compared with the same hyperparameters. In addition, it allows analyzing the results obtained with the two activation functions on the MR-BiESN network. The computation times of the two networks will not be analyzed in this article. It is known that the learning times of ESNs are much lower than those of LSTMs and a study using the activation function of the hyperbolic tangent with has already been done for gesture recognition in [22].

Fig. 7 and Table III present forecasting results obtained in the three scenarios using 50% of data for the training of networks.

Firstly, it is interesting to note that both approaches using the TANH activation function provide very good performance. The predicted voltages are also very similar between the two approaches with an MSE difference between the two medians of about 2×10^{-6}. This demonstrates that the Bi-ESN approach can compete with more classical methods while reducing both the number of parameters to be tuned (advantage of the number of reservoirs) but also the optimization time (fixing the weights of the reservoirs at initialization). In addition, it is possible to observe that GELU and TANH activation functions give quite different results. Indeed, the GELU function seems to capture a more linear dynamic while the TANH function captures a less linear dynamic closer to the real degradation. A possible explanation is that the GELU activation function is strongly linear in the interval [0, ∞] while the TANH function is only linear around 0. Depending on the knowledge of the degradation dynamics it can be interesting to use one or the other function. One possibility would be to add reservoirs to automatically capture the most suitable dynamics. Another interesting point is that data normalization does not improve the results obtained with the GELU activation function according to Fig. 5. This confirms the fact that data normalization is not always necessary with univariate time series even if it can improve the computation time.

TABLE III. SUMMARY OF METRICS OBTAINED FOR MR-BIESN AND STACKED LSTM APPROACHES WITH DATA STANDARDIZATION

	MR-BiESN - GELU		**MR-BiESN - TANH**		**Stacked LSTM - TANH**	
	MSE	***MAPE***	***MSE***	***MAPE***	***MSE***	***MAPE***
Median	6,24E-04	5,48E-03	3,02E-04	2,93E-03	3,04E-04	3,00E-03
1st quartile	5,63E-04	5,09E-03	2,94E-04	2,87E-03	2,98E-04	2,94E-03
3rd quartile	7,60E-04	6,23E-03	3,05E-04	2,96E-03	3,34E-04	3,34E-03
IQR	1,96E-04	1,14E-03	1,03E-05	9,34E-05	3,62E-05	3,94E-04
Mean	6,97E-04	5,84E-03	3,02E-04	2,95E-03	3,17E-04	3,14E-03
Standard deviation	2,22E-04	1,22E-03	1,12E-05	1,18E-04	2,86E-05	3,12E-04

V. CONCLUSION

This paper presents a prognostic approach based on the use of several bidirectional Echo State Networks reservoirs used in parallel. The objective of the developed algorithm is to predict the voltage degradation of a Low-Temperature PEMFC fuel cell with a method that combined reservoir computing advantages while reducing user expertise. Unlike methods using a single ESN reservoir, the presented architecture dispenses with the need to find an ideal set of parameters. The combination of reservoirs in parallel allows the optimization of hyper-parameters of ESN to be avoided.

The MR-BiESN has been validated by comparing it with a well-known approach which is the BiLSTM. The results showed that both methods perform well in detecting and reproducing the degradation trend when the fuel cell is operated at a constant rate. However, for the same number of neurons, the number of parameters to be optimized is much lower (more than 1200 times smaller) for the MR-BiESN, which facilitates the convergence of the optimization algorithms. The impact of the two activation functions was measured and it appears that the GELU function captures linear degradations more easily while TANH captures non-linear degradations more. Due to the small number of

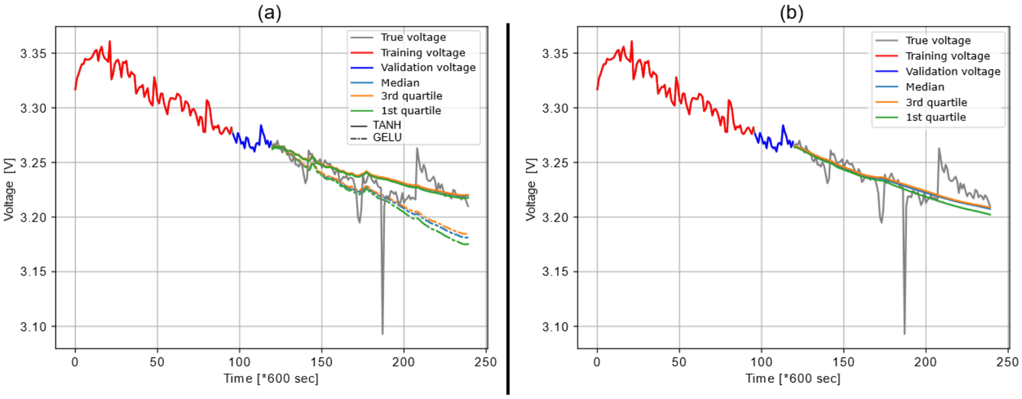

Fig. 7. Comparison of voltage predicted by the MR-BiESN (a) and BiLSTM (b) networks using 50% of data for training and data standardization

parameters to be trained, it is possible to have a combination of reservoirs using different activation functions to let the network automatically select the right one during training.

One of the main difficulties highlighted in this study is setting the regularization parameter to optimize the repeatability of the results. Therefore, future research will focus on improving the robustness of the algorithm with other databases while reducing the number of parameters to be defined by user expertise or by an empirical method (e.g. number of neurons, activation function combination, weight decay & learning rate scheduling …).

Acknowledgment

This project has received funding from the Fuel Cells and Hydrogen 2 Joint Undertaking (JU) under grant agreement no. 875047 Website: https://www.rubyproject.eu/

This work was supported by the doctoral school EIPHI (contract ANR-17- EURE-0002) and the Region Bourgogne Franche-Comté.

References

[1] "Fuel Cells | Department of Energy." https://www.energy.gov/eere/fuelcells/fuel-cells (accessed Mar. 11, 2022).

[2] R. Mezzi, S. Morando, N. Y. Steiner, M. C. Péra, D. Hissel, and L. Larger, "Multi-Reservoir Echo State Network for Proton Exchange Membrane Fuel Cell Remaining Useful Life prediction," in *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2018, pp. 1872–1877. doi: 10.1109/IECON.2018.8591345.

[3] J. Luo, M. Namburu, K. Pattipati, L. Qiao, M. Kawamoto, and S. Chigusa, "Model-based Prognostic Techniques," p. 11.

[4] X. Chen, M. Xiao, and B. Wen, "Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Cataclysmic Mutation Genetic Algorithm and Support Vector Regression," in *2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA)*, Jan. 2021, pp. 766–769. doi: 10.1109/ICPECA51329.2021.9362683.

[5] J.-E. Lee and J.-R. Jiang, "Time Series Multi-Channel Convolutional Neural Network for Bearing Remaining Useful Life Estimation," in *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*, Oct. 2019, pp. 408–410. doi: 10.1109/ECICE47484.2019.8942782.

[6] W. Teng, C. Han, Y. Hu, X. Cheng, L. Song, and Y. Liu, "A Robust Model-Based Approach for Bearing Remaining Useful Life Prognosis in Wind Turbines," *IEEE Access*, vol. 8, pp. 47133–47143, 2020, doi: 10.1109/ACCESS.2020.2978301.

[7] Harel, Fabien, "IEEE PHM Data Challenge 2014." UAR Fuel Cell Lab, Jul. 13, 2021. doi: 10.25666/DATAUBFC-2021-07-19.

[8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[9] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches." arXiv, Oct. 07, 2014. Accessed: Aug. 29, 2022. [Online]. Available: http://arxiv.org/abs/1409.1259

[10] H. Jaeger, "The" echo state" approach to analysing and training recurrent neural networks-with an erratum note'," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, Jan. 2001.

[11] W. Maass, T. Natschläger, and H. Markram, "Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations," *Neural Computation*, vol. 14, no. 11, pp. 2531–2560, Nov. 2002, doi: 10.1162/089976602760407955.

[12] M. Lukoševičius, "A Practical Guide to Applying Echo State Networks," in *Neural Networks: Tricks of the Trade*, vol. 7700, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 659–686. doi: 10.1007/978-3-642-35289-8_36.

[13] S. Lun, H. Hu, and X. Yao, "The modified sufficient conditions for echo state property and parameter optimization of leaky integrator echo state network," *Applied Soft Computing*, vol. 77, pp. 750–760, Apr. 2019, doi: 10.1016/j.asoc.2019.02.005.

[14] H. Jaeger, "A tutorial on training recurrent neural networks, covering BPTT, RTRL, EKF and the 'echo state network' approach," p. 46.

[15] "Jaeger - A tutorial on training recurrent neural networks, .pdf." Accessed: Mar. 03, 2022. [Online]. Available: https://www.ai.rug.nl/minds/uploads/ESNTutorialRev.pdf

[16] F. M. Bianchi, S. Scardapane, S. Løkse, and R. Jenssen, "Bidirectional deep-readout echo state networks," *arXiv:1711.06509 [cs]*, Feb. 2018, Accessed: Mar. 04, 2022. [Online]. Available: http://arxiv.org/abs/1711.06509

[17] C. Sun, M. Song, S. Hong, and H. Li, "A Review of Designs and Applications of Echo State Networks," *arXiv:2012.02974 [cs]*, Dec. 2020, Accessed: Mar. 12, 2021. [Online]. Available: http://arxiv.org/abs/2012.02974

[18] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *arXiv:1606.08415 [cs]*, Jul. 2020, Accessed: Mar. 08, 2022. [Online]. Available: http://arxiv.org/abs/1606.08415

[19] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," *arXiv:1711.05101 [cs, math]*, Jan. 2019, Accessed: Feb. 07, 2022. [Online]. Available: http://arxiv.org/abs/1711.05101

[20] L. Latha, "Efficient approach to Normalization of Multimodal Biometric Scores," *International Journal of Computer Applications*, vol. 32, p. 8.

[21] S. Bhanja and A. Das, "Impact of Data Normalization on Deep Neural Network for Time Series Forecasting," p. 6.

[22] D. Jirak, S. Tietz, H. Ali, and S. Wermter, "Echo State Networks and Long Short-Term Memory for Continuous Gesture Recognition: a Comparative Study," *Cogn Comput*, Oct. 2020, doi: 10.1007/s12559-020-09754-0.