Prediction of Photovoltaic Time Series by Recurrent Neural Networks and Genetic Embedding

Antonello Rosato*, Rodolfo Araneo† and Massimo Panella*
*Dept. of Information Engineering, Electronics and Telecommunications (DIET)
University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, Italy
Email: {antonello.rosato, massimo.panella}@uniroma1.it
†Dept. of Astronautical, Electrical and Energy Engineering (DIAEE) - Electrical Engineering Division
University of Rome "La Sapienza", via Eudossiana, 18, 00184 Italy
Email: rodolfo.araneo@uniroma1.it

***Abstract*—The need of reliable prediction algorithms of energy production is increasing due to the spread of smart solution for grid, plant and resource management. Recurrent neural networks are a viable solution for prediction but their performance is somewhat insufficient when the time series is generated by an underlying process that behaves in a complex manner. In this paper, a new combination of echo state network and genetic algorithms is employed in order to improve the prediction accuracy of photovoltaic time series. The genetic algorithm is used to embed past samples of the time series to be used for predicting a new one. It aims at a feature extraction in order to regularize data being fed into a neural network model, so that it is able to learn more robust and generalizable prediction models. The experimental tests prove that the proposed approach is suited to the application focused in this paper.**

I. Introduction

To forecast energy production is one of the most important analysis for the cost-effective management of power plants. This is important especially in the photovoltaic (PV) generation, as the fluctuating production is much more mutable than the one from other sources and knowing its variations in advance is crucial for adapting the grid infrastructure. In this scenario, the high penetration level of renewable generation of smaller generating units (i.e. prosumers connected to the distribution network) calls for reliability and flexibility of the grid. These can be achieved only via the essential deployment of a grid and plant management system, in which the prediction PV power plant production results of paramount importance.

In this context, the prediction of complex real-world time series is a well-known hard problem [1], as the series bears some inherited characteristics such as nonlinearity and non-stationarity [2]. Statistical approaches [3] like auto regressive moving average (ARMA), auto regressive integrated moving average (ARIMA) and autoregressive integrated moving average with exogenous variables (ARIMAX) have deficient accuracy in this peculiar case.

The prediction accuracy can be improved by using other forecasting methods, which make use of computational intelligence, Neural Networks and Fuzzy Logic [4], [5]. Another popular technique relies on the Echo State Network (ESN) model [6]. ESN is a class of Recurrent Neural Networks (RNNs) in which the recurrent and the non-recurrent part, that is 'reservoir' and 'readout' respectively, are separated. The recurrent part is fixed and the non-recurrent one is solved as a standard linear regression over its weights. ESNs are widely used in different domains including, for instance, chaotic time-series prediction [7], grammatical inference [8], stock price prediction [9] and acoustic modeling problems [10].

In time series prediction, the estimation of the underlying mapping function between the sample to be predicted and the ones previously measured is often an ill-posed problem. This can dramatically affect the overall performance, especially when using advanced learning paradigms for neural networks. In most cases this problem is solved by employing an embedding technique to reconstruct the state-space evolution of the system and to choose accurately the past samples that will feed the input of the neural network [11].

Genetic Algorithms (GAs) are metaheuristics that belong to the larger set of evolutionary algorithms [12]. In this paper, we propose to adopt ESN as a reference prediction model and GAs to find the optimal choice of past samples to be embedded and fed through the input of ESNs. This allows us to improve the prediction accuracy and to reduce the dimension of the input space, therefore acting as a feature extraction process that regularizes data being fed into a neural network model, so that the latter is able to learn more robust and generalizable prediction models. Preliminary approaches in this context have been recently proposed in the literature, they consider the use of genetic algorithms and machine learning models to analyze energy time series and/or the related financial data [13]–[19].

The prediction performance is evaluated in the proposed approach by means of different tests carried out on real-world time series, comparing it to the standard embedding technique also using other regression models adopted for prediction, such as linear Least Squares Estimator (LSE), Radial Basis Function (RBF), Adaptive Neuro-Fuzzy Inference System (ANFIS), and Mixture of Gaussian (MoG) neural networks.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

II. Echo State Network Prediction Model

As a class of RNN, the ESN model can be split into three different components, as shown in Fig. 1, where the boxes enclose the different parts; their connection lines are dashed if the relative link is random, solid if it is trainable.

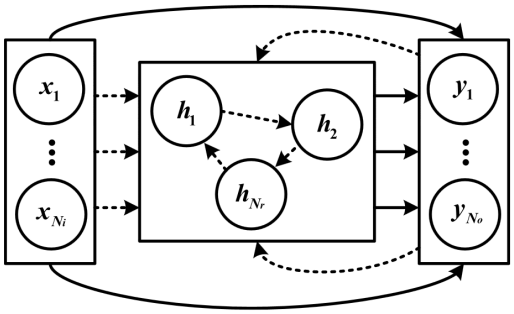


Fig. 1. General scheme of an ESN. The three layers (i.e., Input, Reservoir and Output) are boxed and their connection are explicit: dashed if they are random, solid if trainable.

The input to the network is an N_i dimensional vector $\boldsymbol{x} \in \mathbb{R}^{N_i}$ feeding a reservoir with dimension of N_r. At the generic time step n, the reservoir internal state $\boldsymbol{h} \in \mathbb{R}^{N_r}$ is then updated using the following equation:

$$\boldsymbol{h}[n] = f_{\text{res}}\left(\boldsymbol{W}_i^r \boldsymbol{x}[n] + \boldsymbol{W}_r^r \boldsymbol{h}[n-1] + \boldsymbol{W}_o^r \boldsymbol{y}[n-1]\right), \quad (1)$$

where $\boldsymbol{W}_i^r \in \mathbb{R}^{N_r \times N_i}$, $\boldsymbol{W}_r^r \in \mathbb{R}^{N_r \times N_r}$ and $\boldsymbol{W}_o^r \in \mathbb{R}^{N_r \times N_o}$ are matrices generated randomly, $f_{\text{res}}(\cdot)$ is a nonlinear function defined suitably, and $\boldsymbol{y}[n-1] \in \mathbb{R}^{N_o}$ is the previous N_o-dimensional output of the network. If stability has to be increased, adding a small uniform noise to the update is a viable option. This is done before computing the nonlinear transformation $f_{\text{res}}(\cdot)$ [20]. The output will be then evaluated as:

$$\boldsymbol{y}[n] = f_{\text{out}}(\boldsymbol{W}_i^o \boldsymbol{x}[n] + \boldsymbol{W}_r^o \boldsymbol{h}[n]), \quad (2)$$

where $\boldsymbol{W}_i^o \in \mathbb{R}^{N_o \times N_i}, \boldsymbol{W}_r^o \in \mathbb{R}^{N_o \times N_r}$ are modified and adapted in accordance with the training data, and $f_{\text{out}}(\cdot)$ is a nonlinear (invertible) function. For learning applications, it is fundamental for the reservoir to satisfy the 'echo state property' (ESP) [21]. In other words, given an input, its effect on the reservoir state must disappear in a finite number of time steps. Usually, by a rule-of-thumb, the matrix $\boldsymbol{W}_r^r$ is rescaled, resulting in $\rho(\boldsymbol{W}_r^r) < 1$, where $\rho(\cdot)$ denotes the spectral radius operator.

In a general learning problem with scalar outputs, the ESN is trained by feeding the reservoir a sequence of Q desired input-outputs pairs $\{\boldsymbol{x}[1], d[1]\}, \ldots, \{\boldsymbol{x}[Q], d[Q]\}$. The states are then 'gathered' as a sequence $\boldsymbol{h}[1], \ldots, \boldsymbol{h}[Q]$. In this phase, the desired output is used for feedback because the ESN output is not yet available. We define the hidden matrix $\boldsymbol{H}$ and output vector $\boldsymbol{d}$ as:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{x}^T[1] \;\; \boldsymbol{h}^T[1] \\ \vdots \\ \boldsymbol{x}^T[Q] \;\; \boldsymbol{h}^T[Q] \end{bmatrix} \quad (3)$$

$$\boldsymbol{d} = \begin{bmatrix} f_{\text{out}}^{-1}(d[1]) \\ \vdots \\ f_{\text{out}}^{-1}(d[Q]) \end{bmatrix} \quad (4)$$

The optimal output weight vector is then computed as the solution of the regularized least squares problem:

$$\boldsymbol{w}^* = \text{argmin}_{\boldsymbol{w} \in \mathbb{R}^{N_i + N_r}} \frac{1}{2} ||\boldsymbol{H}\boldsymbol{w} - \boldsymbol{d}||_2^2 + \frac{\lambda}{2} ||\boldsymbol{w}||_2^2, \quad (5)$$

where $\boldsymbol{w} = [\boldsymbol{W}_i^o \, \boldsymbol{W}_r^o]^T$ and $\lambda \in \mathbb{R}^+$ is a positive scalar known as *regularization factor*. A close form solution of problem in (5) can be obtained as:

$$\boldsymbol{w}^* = \left(\boldsymbol{H}^T \boldsymbol{H} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{H}^T \boldsymbol{d}. \quad (6)$$

Given their transient state, in practice it is possible to remove some initial elements from the sequence with which the least squares problem is solved. Usually, the number of these 'dropout' elements is fixed a priori.

Given a time series $S[n]$, $n > 0$, to be predicted, by this general formulation of ESN applied to prediction, the input $\boldsymbol{x}$ to the ESN is a vector of N_i consecutive past samples of the time series while the desired output d is the sample to be predicted; more precisely:

$$\begin{aligned} &\boldsymbol{x}[n - N_i + 1] = \left[S[n] \; S[n-1] \; \cdots \; S[n - N_i + 1]\right]^T, \\ &d[n - N_i + 1] = S[n + m], \\ &n = N_i, \;\; N_i + 1, \;\; \ldots \;\; Q + N_i - 1, \end{aligned} \quad (7)$$

where $m > 0$ is the prediction distance, which is usually set to $m = 1$. The available samples of the time series should range from $S[1]$ up to $S[Q + N_i + m - 1]$. In the following, we propose a generalized approach for which, at the generic time n, the input to the ESN reservoir can be any combination of past available samples, which are used to predict the next one $S[n + m]$.

III. Time Series Embedding by Genetic Algorithms

As the relation between predicted and past samples usually is not given, in a prediction problem it is important to study which samples of the sequence are relevant to the prediction task. The naive approach described in (7) resides in assuming that all the past samples of the sequence to be predicted are equally important, without finding an optimal subset. However, a more fruitful approach consists in selecting the samples via embedding of the sequence [22]. The sequences generated by a complex, often chaotic and noisy system like the PV under examination, can be observed by its output only. So, in order to reconstruct the evolution of the system, the original sequence $S[n]$ must be embedded. This is done by determining two

parameters [2]: the embedding dimension D and the time lag T, in such a way the reconstructed state at time n can be found as:

$$\overline{\boldsymbol{S}}[n] = \Big[S[n]\ S[n-T]\ \ldots\ S[n-T(D-1)]\Big]^T. \quad (8)$$

Looking at (7), it is evident that the ESN input, feeding the reservoir, is an estimate of the unfolded, reconstructed state of the unknown system that generates the observed time series, having assumed $T = 1$ and $D = N_i$. However, to set the embedding parameters at such values might not be an optimal choice. In fact, these parameters should be estimated using the Average Mutual Information (AMI) procedure for T and the False Nearest Neighbors (FNN) procedure for D [23]. Nonetheless, as these algorithms often suffer from a too high sensitivity to noise and to some numeric thresholds, in this paper we propose a different and more general embedding procedure, employing GAs to select the optimal subset of past samples, thus reducing the dimensionality of the ESN's input space while improving the accuracy of the prediction.

As introduced in the Sect. I, GAs are optimization techniques that follow natural selection principles. The evolution of a population of individuals is considered, where each individual is associated with a possible solution of the problem to be solved. In the present implementation, each individual is represented by a binary string associated with the time lags that define, for any time n, the subset of past samples that will feed the prediction model. In the same way, the fitness of each individual is the prediction error obtained by using that particular embedding (i.e., that subset of past samples). In the present implementation, the Normalized Mean Squared Error (NMSE), which is introduced in the following Sect. IV, has been adopted for evaluating the prediction performance and the related fitness of each individual.

In more detail, the bit string in a chromosome of each individual allows us to select which samples of the sequence are to be considered for building the input of the prediction model [24]. Let L be the length of a chromosome $\boldsymbol{C} = [C_1\, C_2 \ldots C_L]^T$. The jth bit C_j of the genetic code is set to 1 if the associated sample will feed the prediction model, 0 otherwise. Hence, $S[n-j+1]$ will be considered in the vector $\overline{\boldsymbol{S}}[n]$ estimating the reconstructed state if and only if the corresponding bit C_j, $j = 1 \ldots L$, is set to 1; the input dimension will be the number of considered samples or, equivalently, of the bits set to 1:

$$N_i = \sum_{j=1}^{L} C_j\,. \quad (9)$$

Let us illustrate further by the visual example in Fig. 2, where $L = 16$. In this case, not all of the 16 samples are considered for embedding, that is:

$$\overline{\boldsymbol{S}}[n] = \Big[S[n]\ S[n-1]\ S[n-4]\ S[n-9]\ S[n-12]]\Big]^T. \quad (10)$$

It is important to note how there can be a dimensionality reduction, in fact $N_i = 5$ and $N_i < L$.

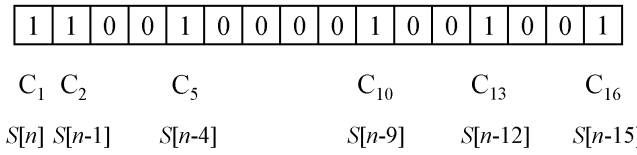


Fig. 2. A possible chromosome for $L = 16$.

In order to increase the fitness of the best individual in a population of candidates, the GA starts with a set of random individuals that evolve iteratively in successive sets of generations. Given the kth generation G_k, the successive one G_{k+1} is constructed selecting the best individuals based on their fitness, and modifying them by the application of the mutation, crossover and selection operators to form the new population. The algorithm adopted in this paper can be summarized as follows:

1) Initialization: a population G_0 with P individuals is created and set as the current generation.
2) The individuals of G_0 are sorted by ascending values of the fitness function, as the lower is the prediction error the better the fitness.
3) The next generation is created from the current one by means of elitism and using mutation and crossover.
4) The next generation becomes the current one.

Steps 2, 3 and 4 are iterated for a predefined fixed number M_{gen} of generations.

The operational functioning of the whole GA hinges on P and M_{gen} values, and on the mutation and crossover rates M_r and C_r, given as probability thresholds, respectively. The next generation G_{k+1} is produced from the current one G_k as follows:

1) The best individual of G_k is chosen and put in G_{k+1} (elitism). This assures a non-increasing behavior of the best fitness value from a generation to the successive one.
2) A pair of parents is randomly selected by using the 'Roulette Wheel' technique with a selection probability proportional to their fitness. The two parents are crossed-over using a 'two-point' crossover with a probability equal to C_r.
3) For every pair of parents, each individual resulting from crossover is mutated by bit-flip with probability equal to M_r; the two resulting individuals are placed in G_{k+1}.

Step 4 is repeated until the next generation contains exactly P individuals.

IV. Experimental Results

The proposed approach focuses on a particular application, regarding the power plant located in Sant'Eusanio del Sangro, in the Abruzzo region of Italy. The data is relative to a single PV plant that belongs to a broader system with other interconnected plants operated by the same agent. The complete numerical dataset is available at https://github.com/max-panella/panella/raw/master/CEC2020_Dataset.zip.

In the considered prediction problem, the used time series is composed by the output voltage of the plant subsampled at one sample per hour, thus we have a sequence of 24 samples per day. The time series is linearly normalized in the range $[-0.5, +0.5]$, so as to be able to satisfy the numerical requirements of the data driven prediction models. All the experiments are carried out by using prediction distance $m = 1$, standard embedding as in (8) as well as the genetic one proposed herein. For the classic embedding, the values of parameters T and D are chosen by using the AMI and FNN methods, respectively.

The proposed genetic embedding can be applied also selecting the inputs that feed any other regression model, not only the ESN. Therefore, for the sake of comparison, both standard and genetic embedding will be used with the following state-of-the-art prediction models:

- LSE: the relationship between the value to be predicted and the current ones is modeled as a linear function, a least squares algorithm is used to estimate the parameters [25];
- RBF: a neural network that builds up a function approximation model with a usually multiquadratic radial basis functions [26];
- ANFIS: a data driven fuzzy inference system based on a set of Sugeno first-order type fuzzy rules [27];
- MoG: a neural network model in which a density mixture of Gaussian components are used in the joint input-output space; the mixture parameters are estimated via the Expectation-Maximization [28].

Remark. In the present paper, we are not considering enhanced recurrent models based on deep neural networks, for instance, Long Short-Term Memory (LSTM) networks [29], Gated Recurrent Unit (GRU) networks [30], and so on. Although they can be sufficiently fast during the inference (prediction) phase, the learning process of such systems and the related cross-validation (on a huge set of hyperparameters) might be computationally expensive. As the present application deal with non-stationary PV time series, the whole system need to be retrained frequently. Moreover, despite any possible parallel implementation of GAs, both training and model selection/regularization is necessary for each individual being considered. For such a reason, it is important to investigate whether classic shallow models, which are easier to be trained, are able to attain the expected performance possibly through the support of evolutionary algorithms for feature selection and data embedding. ESNs cope perfectly with this constraint, as they rely on a randomized reservoir and the readout can be trained in one shot by using a closed-form least squares solution.

GA selection is performed by considering separately each one of the five regression models previously introduced. The adopted dataset is a single month (mostly 31 days) of the time series, then resulting in 744 samples. During fitness evaluation, given the inputs selected by the chromosome of each individual, the regression model is trained by using an input-output dataset that is built, through embedding, on the first 500 samples of the adopted time series. The remaining 244 samples are used for evaluating the fitness by using (11), as successively defined. Such samples are also used to cross-validate the adopted regression model; i.e., for each individual, a different model complexity (i.e., reservoir dimension N_r in ESN, order of linear predictor, Gaussian kernels in RBF and MoG, number of fuzzy rules in ANFIS) is considered and the one scoring the best fitness is selected for that individual. The final solution found by the proposed GA is tested on the next operation day (i.e., 24 samples) after the end of the month.

The main parameters of the genetic process are: chromosome length $L = 72$ (i.e., 3 days); population size $P = 100$; $C_r = 1$; $M_r = 0.3$; maximum number of generations $M_{\text{gen}} = 30$; Roulette Wheel selection algorithm; two-point crossover. All of the experiments had been performed using Matlab™ R2019a on a machine equipped with Intel® Core™ i7-3770K 64-bit CPU at 3.50 GHz, 32 GB RAM, and NVIDA GTX 680 GPU.

The proposed GA was run 10 times (each run with a different initial population randomly initialized) and we report successively the average results obtained in terms of the following metrics, which are commonly adopted for prediction performance:

- Normalized Mean Squared Error (NMSE)

$$\text{NMSE} = \frac{\sum_n \left(S[n] - \widetilde{S}[n]\right)^2}{\sum_n \left(S[n] - \bar{S}\right)^2}, \tag{11}$$

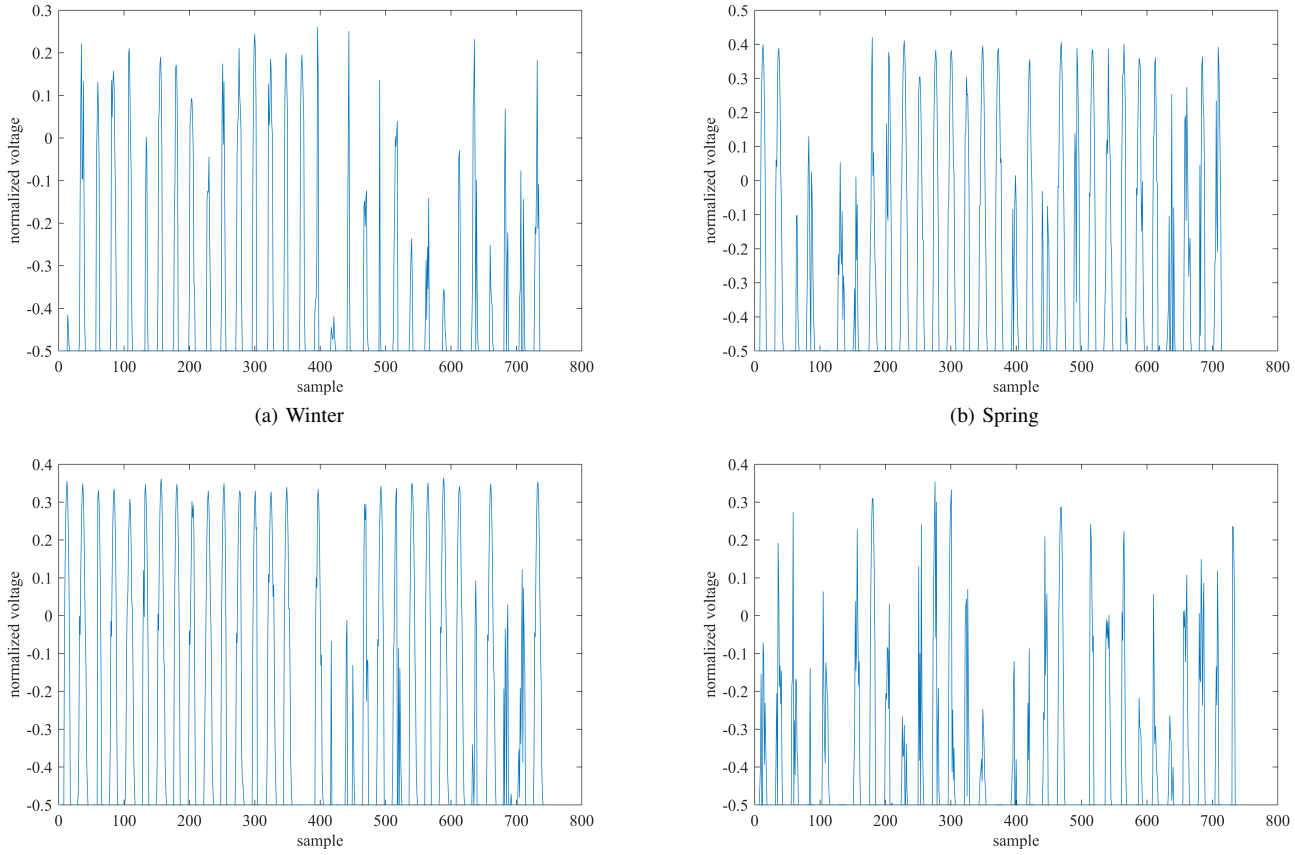
  where $\widetilde{S}[n]$ is the predicted value of $S[n]$ and $\bar{S}$ is the average on the considered values of $S[n]$;
- Error-to-Signal Ratio (ESR) (in dB)

$$\text{ESR}_{\text{dB}} = 10 \log_{10} \frac{\sum_n \left(S[n] - \widetilde{S}[n]\right)^2}{\sum_n S[n]^2}. \tag{12}$$

TABLE I
AVERAGE RESULTS (OVER 10 RUNS) FOR THE WINTER TEST SET

Prediction model	NMSE		ESR	
	Standard embedding	GA embedding	Standard embedding	GA embedding
LSE	0.825	0.745	-6.914	-7.354
RBF	0.616	0.693	-8.182	-7.669
ANFIS	0.677	0.654	-7.773	-7.923
MoG	0.753	0.603	-7.310	-8.272
ESN	0.619	0.586	-8.162	-8.398

We tested the algorithm by using four different sets of data, all taken from the described time series relative to the year 2016. These sets exhibit some different behaviors, as for the maximum values reached in a day and for the volatility of the time series. The training set is a whole month of a different season (winter, spring, summer, and autumn), as described in the following and shown in Fig. 3:



(a) Winter (b) Spring

(c) Summer (d) Autumn

Fig. 3. Adopted training sets of one month-length per season.

TABLE II
AVERAGE RESULTS (OVER 10 RUNS) FOR THE SPRING TEST SET

Prediction model	NMSE		ESR	
	Standard embedding	GA embedding	Standard embedding	GA embedding
LSE	0.066	0.056	-13.023	-13.772
RBF	0.062	0.055	-13.298	-13.820
ANFIS	0.065	0.059	-13.112	-13.559
MoG	0.063	0.061	-13.265	-13.362
ESN	0.055	0.051	-13.811	-14.151

TABLE III
AVERAGE RESULTS (OVER 10 RUNS) FOR THE SUMMER TEST SET

Prediction model	NMSE		ESR	
	Standard embedding	GA embedding	Standard embedding	GA embedding
LSE	0.057	0.037	-13.880	-15.724
RBF	0.062	0.033	-13.476	-16.250
ANFIS	0.044	0.043	-14.935	-15.021
MoG	0.045	0.041	-14.884	-15.305
ESN	0.036	0.030	-15.796	-16.631

TABLE IV
AVERAGE RESULTS (OVER 10 RUNS) FOR THE AUTUMN TEST SET

Prediction model	NMSE		ESR	
	Standard embedding	GA embedding	Standard embedding	GA embedding
LSE	0.234	0.130	-10.367	-12.917
RBF	0.132	0.082	-12.858	-14.918
ANFIS	0.159	0.107	-12.034	-13.752
MoG	0.155	0.141	-12.155	-12.557
ESN	0.099	0.072	-14.092	-15.462

- Winter: from January 1 to January 31, 2016, standard embedding parameters $T = 13$, $D = 25$;
- Spring: from March 31 to April 30, 2016, standard embedding parameters $T = 10$, $D = 20$;
- Summer: from July 1 to July 31, 2016, standard embedding parameters $T = 10$, $D = 14$;
- Autumn: from October 1 to October 31, 2016, standard embedding parameters $T = 10$, $D = 13$.

The numerical results for each test set (i.e., from sample 745 to 768) are reported in Tables I-IV, while the graphical behavior is shown in Figs. 4-7, where the time series obtained in one run of the proposed GA by using the proposed genetic embedding and ESN regression model are plotted.

TABLE V
SELECTED SAMPLES BY GENETIC EMBEDDING USING THE ESN MODEL

Dataset	Std. embedding		GA embedding	
	D	T	N_i	Selected samples
Winter	25	13	8	$S[n]$ $S[n-1]$ $S[n-2]$ $S[n-7]$ $S[n-9]$ $S[n-17]$ $S[n-21]$ $S[n-23]$
Spring	20	10	9	$S[n]$ $S[n-1]$ $S[n-4]$ $S[n-7]$ $S[n-11]$ $S[n-15]$ $S[n-17]$ $S[n-26]$ $S[n-29]$
Summer	14	10	6	$S[n]$ $S[n-2]$ $S[n-7]$ $S[n-14]$ $S[n-20]$ $S[n-24]$
Autumn	13	10	7	$S[n]$ $S[n-2]$ $S[n-6]$ $S[n-18]$ $S[n-23]$ $S[n-30]$ $S[n-35]$

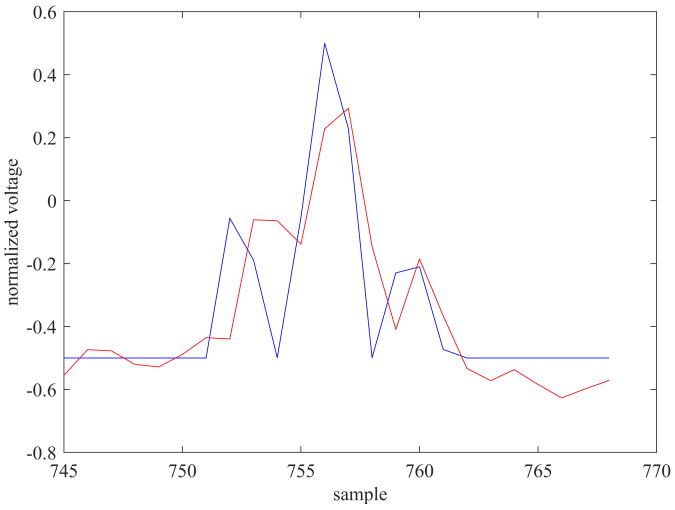


Fig. 4. Winter test set (blue) and predicted time series (red).

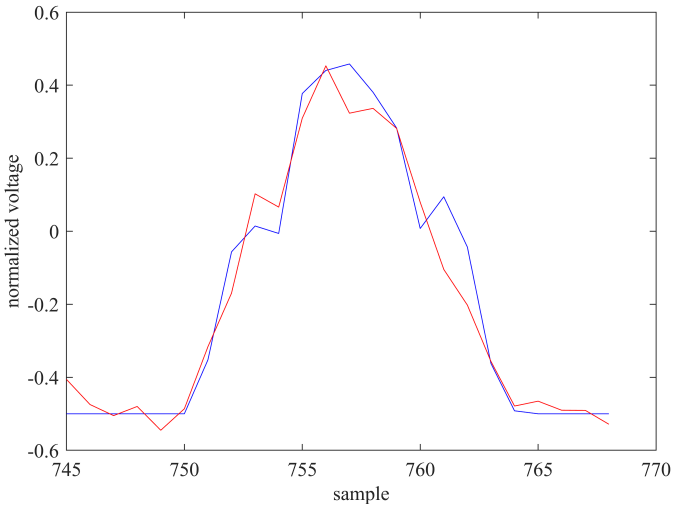


Fig. 5. Spring test set (blue) and predicted time series (red).

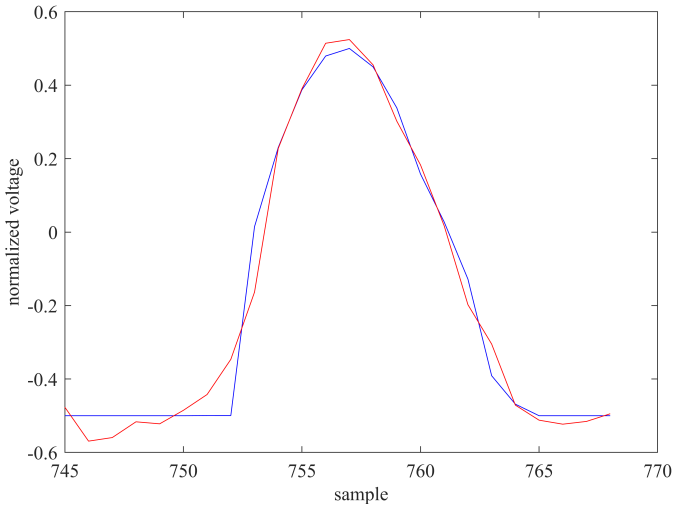


Fig. 6. Summer test set (blue) and predicted time series (red).

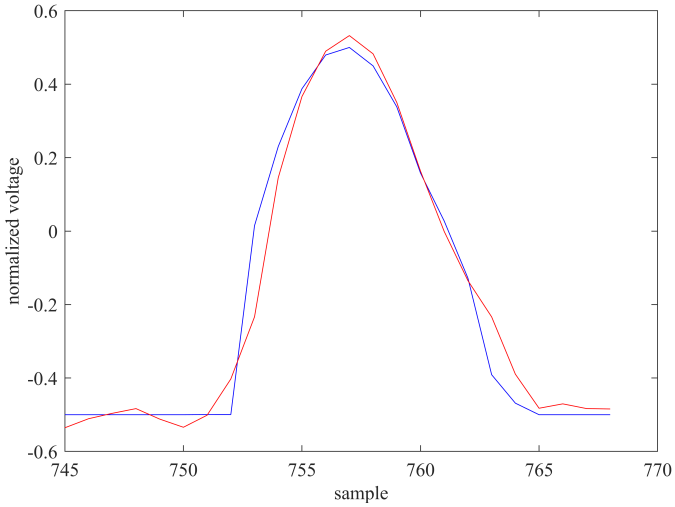


Fig. 7. Autumn test set (blue) and predicted time series (red).

There are many differences among the various seasons and winter seems to be the hardest one to be predicted. Anyway, apart from one case only, for all seasons the performance of the genetic embedding performs better than standard embedding, independently of the adopted prediction model, with an ESR gain up to 2 dB. Also, for every season the proposed ESN approach with genetic embedding always obtains the best result with respect to other prediction models.

It is also worth noticing that, by using the proposed approach, the embedding dimension, which is also the order of the predictor, is heavily reduced. For instance, we report in Table V the selected samples associated with the best individual obtained at the end of one run of the genetic process using ESN for prediction.

It is evident the reduction of the used samples with respect to standard embedding, although the prediction performance increases. This proves that the genetic embedding regularizes the prediction process and also reduces the curse of dimensionality, which is typical for data driven learning systems, like neural networks, when the dimension of the data space becomes too large with respect to the available samples in the training set.

Furthermore, the genetic optimization always tends to select the latest available sample, some in the latest hours and then, at nearly the same hour one day before. Even the use of the Roulette Wheel for selection does not create issues as the algorithm starts to converge, as a proper persistence of elitism is adopted. In general, no further samples are used. Such a result agrees with the technical rationale of prediction in PV plants, which are characterized by a daily periodicity, and substantiates the robustness of the proposed genetic selection.

It is important to outline that the samples obtained in Table V by genetic embedding are strictly related to the considered data set used during the evolutionary process and hence, they depend on the specific time window where the time series is analyzed. Commonly speaking, the selected samples cannot be generalized to other periods because of the non-stationarity and seasonality of PV time series. In spite of this, the reported results prove a stable trend of GAs, which are able to reduce data dimensionality.

V. Conclusion

In this paper, the application of ESNs and GAs to the prediction of real-world time series has been investigated, aiming at a reliable solution to the difficult problem of forecasting the power generation in PV plants. The results are promising as the proposed approach always performs better than other well-known and standard benchmarks. In future works, the proposed system might be considered for parallel and distributed implementations, in order to cope with the constraints of learning and forecasting on multiple distributed sources of energy and photovoltaic correlated data. Also, more complex recurrent models, based on deep neural networks, might be considered, provided that a sufficient computing power balanced with sustainable costs is ensured for that application.

References

[1] R. Inman, H. T.C. Pedro, and C. F.M. Coimbra, "Solar forecasting methods for renewable energy integration," *Progress in Energy and Combustion Science*, vol. 39, p. 535–576, 12 2013.

[2] H. Abarbanel, *Analysis of Observed Chaotic Data.* Springer, New York, 1996.

[3] C. Bennett, R. A. Stewart, and J. Lu, "Autoregressive with exogenous variables and neural network short-term load forecast models for residential low voltage distribution networks," *Energies*, vol. 7, no. 5, pp. 2938–2960, 2014. [Online]. Available: http://www.mdpi.com/1996-1073/7/5/2938

[4] A. Rosato, R. Altilio, R. Araneo, and M. Panella, "Takagi-sugeno fuzzy systems applied to voltage prediction of photovoltaic plants," in *2017 IEEE International Conference on Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I CPS Europe)*, June 2017, pp. 1–6.

[5] ——, "Prediction in photovoltaic power by neural networks," *Energies*, vol. 10, no. 7, 2017.

[6] H. Jaeger, "A tutorial on training recurrent neural networks, covering bppt, rtrl, ekf and the "echo state network" approach," German National Research Center for Information Technology, GMD Report 159, 2002. [Online]. Available: http://minds.jacobs-university.de/sites/default/files/uploads/papers/ESNTutorialRev.pdf

[7] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[8] M. H. Tong, A. D. Bickett, E. M. Christiansen, and G. W. Cottrell, "Learning grammatical structure with echo state networks," *Neural Networks*, vol. 20, no. 3, pp. 424–432, 2007.

[9] X. Lin, Z. Yang, and Y. Song, "Short-term stock price prediction based on echo state networks," *Expert Systems with Applications*, vol. 36, no. 3, pp. 7313–7317, 2009.

[10] F. Triefenbach, A. Jalalvand, K. Demuynck, and J.-P. Martens, "Acoustic modeling with hierarchical reservoirs," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 21, no. 11, pp. 2439–2450, Nov 2013.

[11] M. Casdagli, "Nonlinear prediction of chaotic time series," *Physica D: Nonlinear Phenomena*, vol. 35, no. 3, pp. 335 – 356, 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0167278989900742

[12] D. Goldberg, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[13] G. Dudek, "Artificial immune system with local feature selection for short-term load forecasting," *IEEE Trans. Evol. Comput.*, vol. 21, no. 1, pp. 116–130, Feb 2017.

[14] T. M. Lakshmi, A. Martin, and V. P. Venkatesan, "A genetic bankrupt ratio analysis tool using a genetic algorithm to identify influencing financial ratios," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 38–51, Feb 2016.

[15] O. García-Hinde, V. Gómez-Verdejo, M. Martínez-Ramón, C. Casanova-Mateo, J. Sanz-Justo, S. Jiménez-Fernández, and S. Salcedo-Sanz, "Feature selection in solar radiation prediction using bootstrapped SVRs," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, July 2016, pp. 3638–3645.

[16] N. Ghadimi, A. Akbarimajd, H. Shayeghi, and O. Abedinia, "Application of a new hybrid forecast engine with feature selection algorithm in a power system," *International Journal of Ambient Energy*, vol. 40, no. 5, pp. 494–503, 2019.

[17] D. Niu, H. Wang, H. Chen, and Y. Liang, "The general regression neural network based on the fruit fly optimization algorithm and the data inconsistency rate for transmission line icing prediction," *Energies*, vol. 10, p. 2066, 12 2017.

[18] S. Jiang, K.-S. Chin, L. Wang, G. Qu, and K. L. Tsui, "Modified genetic algorithm-based feature selection combined with pre-trained deep neural network for demand forecasting in outpatient department," *Expert Systems with Applications*, vol. 82, pp. 216 – 230, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417417302610

[19] A. Ahmad, N. Javaid, M. Guizani, N. Alrajeh, and Z. A. Khan, "An accurate and fast converging short-term load forecasting model for industrial applications in a smart grid," *IEEE Trans. Ind. Informat.*, vol. 13, no. 5, pp. 2587–2596, Oct 2017.

[20] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *Advances in neural information processing systems*, 2002, pp. 593–600.

[21] M. LukošEvičIus and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[22] A. Rosato, R. Altilio, R. Araneo, and M. Panella, "Embedding of time series for the prediction in photovoltaic power plants," in *2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)*, June 2016, pp. 1–4.

[23] A. Rosato, M. Panella, R. Araneo, and A. Andreotti, "A neural network based prediction system of distributed generation for the management of microgrids," *IEEE Trans. Ind. Appl.*, vol. 55, no. 6, pp. 7092–7102, 2019.

[24] R. Altilio, M. Paoloni, and M. Panella, "Selection of clinical features for pattern recognition applied to gait analysis," *Medical & Biological Engineering & Computing*, vol. 55, pp. 685–695, 2017.

[25] H. W. Bode and C. E. Shannon, "A simplified derivation of linear least square smoothing and prediction theory," *Proceedings of the IRE*, vol. 38, no. 4, pp. 417–425, 1950.

[26] A. G. Bors and I. Pitas, "Median radial basis function neural network," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1351–1364, 1996.

[27] J.-S. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 3, pp. 665–685, 1993.

[28] M. Panella, A. Rizzi, and G. Martinelli, "Refining accuracy of environmental data prediction by MoG neural networks," *Neurocomputing*, vol. 55, pp. 521–549, 2003.

[29] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, vol. 2, 1999, pp. 850–855 vol.2.

[30] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.