DOI: http://dx.doi.org/10.1109/I2MTC.2014.6860947

A Computational Intelligence approach to Solar Panel Modelling

S. Ferrari[1], M. Lazzaroni[1,2], and V. Piuri[1]
[1]Università degli Studi di Milano
Milan, Italy
[2]INFN, Milan, Italy
Email: {stefano.ferrari, massimo.lazzaroni, vincenzo.piuri}@unimi.it

A. Salman
Doğuş University
Istanbul, Turkey
Email: asalman@dogus.edu.tr

L. Cristaldi, M. Faifer, and S. Toscani
Politecnico di Milano
Milan, Italy
Email: {loredana.cristaldi, marco.faifer, sergio.toscani}@polimi.it

***Abstract*—The power produced by a solar panel depends on several parameters. In order to optimize the production, the ability to operate in the Maximum Power Point (MPP) condition is requested. The ability to identify and reach the MPP condition is therefore critical to an efficient conversion of the photovoltaic energy. In this paper, several computational intelligence paradigms are challenged in the task of identifying the MPP power from the working condition directly measurable from the solar panel, such as the voltage, V, the current, I, and the temperature, T, of the panel.**

I. Introduction

The renewable energy industry has developed significantly in recent years. In this context, the solar energy is one of the more accessible and cheaper energy resources. For this reason, the industry working in this scenario has seen a rapid expansion in the last ten years with the result that now the electricity produced by this technology is shared with the grid. Moreover, in more recent years, the problems associated with the production of electricity are becoming more important. The more rational use of energy resources, and also the production from renewable sources, is the strategies currently adopted worldwide in order to achieve both a reducing emissions of pollutants and a lower environmental impact.

The interest in the production by means of photovoltaic (PV) systems is reasonable in country, such as Italy or in Mediterranean area, where sunny days are particularly frequent and with high intensity. Further consideration that justify the large use of this plants is that this system can be realized in small size, can be easily connected to the national grid in order to obtain a network of distributed generation (grid), or used for combined high efficiency, local level of thermal and electrical energy. While the short term forecast of the produced power is useful in order to manage the different energy sources, for this type of plant the ability to obtain accurate prediction [1][2] and even to program the maintenance of the plant without the needs of expensive equipment [3][4] is of paramount importance.

In this scenario the prediction of the Maximum Power Point (MPP) is of particular importance. The MPP is the maximum point of the Voltage-Power graph. The knowledge, or even better, the ability to predict its value using indirect measurements is of paramount importance. In this paper a methodology for performing the MPP prediction from measures of some features of the working conditions of a panel will be presented. In particular, we will consider some features that are extremely simple to acquire, and prediction will be realized with the soft-computing techniques. In this contest, the considered variables are: the voltage, the current and the temperature. It should be emphasized that, unlike what has been shown in previous papers, in the present work we propose to perform the MPP prediction using only electrical quantities directly measurable at the terminal of the panel or by a sensor installed in the proximity of the panel itself. It will be demonstrated that measurement much more complicated and expensive, such as the measurement of the solar radiation, can be replaced by information much more accessible [5].

In Section II, the models used for prediction will be introduced. In Section III, the experiments run will be described and the results obtained will be reported and discussed in Section IV. The conclusion and directions for future works will be reported in Section V.

II. The Prediction Models

In order to be able to predict the MPP from the voltage-current-temperature state, a mapping between the state space and the MPP space have to be defined. In the present work, several computational intelligence paradigms have been challenged in this task. Namely, the Fuzzy C-means clustering (FCM) and the Radial Basis Functions network (RBF) [6] will be used to this aim. For the sake of comparison, the k-Nearest

Neighbors (k-NN) predictor [7] will be used as baseline for the performance of the predictors.

The use of computational intelligence paradigms has the advantage that no model of the input-output mapping have to be provided in advance, since these paradigms can learn it directly from a finite set of input-output pairs (possibly affected by error), called training set.

A. Fuzzy C-means Clustering

Clustering is an unsupervised process to partition a dataset in order to minimize the dissimilarity inside the partitions (clusters) subject to some constraints, such as a given number of partitions. Depending on the application, a suitable similarity measure can be devised. The clusters can be used to give a compact description of the original dataset, by considering the centroid of each clusters. This representation can be also exploited to obtain an input-output relationship: the centroids represent the input prototypes, which can be associated to the average of the output values corresponding to the elements of the cluster. This representation can then be used to compute the output corresponding to an input point as in the k-NN predictor.

Fuzzy clustering is a generalization of the clustering where each input point can belong to more than one clusters, with different degrees of membership. Among the fuzzy clustering techniques, the Fuzzy C-means (FCM) [8][9] is one of the most used. It is an iterative clustering algorithm in which at every step each point is assigned to each cluster (i.e., associated to the corresponding centroid) with a membership degree that is inversely proportional to the distance from the cluster's centroid. Then, the centroid positions are updated as the average of the points belonging to the cluster; the contribution of each point to the average is weighted with its membership degree. The training can be stopped when the position of the centroids does not changes or using other criteria (e.g., the maximum number of iterations). At the end of the training, the centroids can be used as for the traditional clustering.

B. Radial Basis Functions Networks

Neural networks constitutes a variegated class of models for classification and function approximation [10][11]. Among these, the Radial Basis Function (RBF) networks are very used, because of their simplicity and approximation power. In fact, they enjoy the universal approximation property (i.e., for every continuous function, exists an RBF network that approximates the considered function arbitrarily well). Although several basis functions guarantees this property, the RBF model described in (1), with Gaussian basis functions, is generally used. The mapping can be expressed as a linear combination of basis functions:

$$f(x) = \sum_{i=1}^{L} \beta_i \, G(x;\, \mu_i,\, \sigma_i) + b \tag{1}$$

where L is the number of basis functions, G is the Gaussians function, μ_i, σ_i, and β_i are respectively the center, the width and the coefficient of the i-th Gaussian, and b is an optional bias.

The learning algorithm for a RBF defines a procedure that allow to obtain the parameters (L, $\{\mu_i\}$, $\{\sigma_i\}$, $\{\beta_i\}$, b) from the training set. In particular, several learning algorithms has been provided in literature, with different characteristics (e.g., hybrid learning [6], incremental learning [12], global optimization [13]) and several extensions (e.g., Hierarchical RBF [14][15], Extreme Learning Machine [16]). Among them, the hybrid learning allows to face the estimate of the parameters in different subsequent steps, reducing the complexity of the optimization. Given the training set $\{(x_j,\, y_j) \,|\, x_j \in \mathbb{R}^D,\, y_j \in \mathbb{R},\, j = 1,\, \ldots,\, N\}$, firstly the centers of the Gaussians, $\{\mu_i\}$, are estimated: this is usually operated through a clustering algorithm, since the number of the Gaussians, L can be derived by the given computational budget, and $\{\mu_i\}$ is a set of $\mathbb{R}^D$ points and can be obtained clustering $\{x_j\}$. Since the width of the Gaussians determines their influence region, for each Gaussian it can be set in order to cover at least the region of the corresponding cluster and allowing a given degree of overlapping with the neighboring units. Once these parameters have been set, the weights $\{\beta_i\}$ can be computed as the solution of a linear system. In fact, the output of the RBF network (1) give rise to N equations that can be expressed in matricial notation as:

$$H\beta = \hat{Y} \tag{2}$$

where H is a $N \times L$ matrix such that $H_{j,i} = G(x_j; \mu_i, \sigma_i)$, $\beta = [\beta_1 \cdots \beta_L]^T$, and $\hat{Y} = [\hat{y}_1 \cdots \hat{y}_N]^T$. Given the training dataset and the hidden neurons parameters, the weights β are the only unknown of the linear system described in (2), and, under mild conditions, they can be computed as:

$$\hat{\beta} = (H^T G)^{-1} H^T \hat{Y} = H^{\dagger} \hat{Y} \tag{3}$$

where $H^{\dagger} = (H^T H)^{-1} H^T$ denotes the Moore-Penrose pseudo-inverse of the matrix H.

C. k-Nearest Neighbor Interpolator

The k-Nearest Neighbor (k-NN) model is a instance-based or lazy learning paradigm used both for function approximation and classification [7]. It is used to predict the value of a function, f, in unknown points, given a sampling of the function itself (training data), $\{(x_i,\, y_i) \,|\, y_i = f(x_i)\}$. For an unknown point, x, the value of $f(x)$ is estimated from the value of its k nearest neighbors, for a given k, using a suitable voting scheme or an average. The most simple scheme, often used in classification, estimates $f(x)$ as the most common output value among its neighbors, while in function approximation the average output value is commonly used. More complex schemes, such as the use of weighted averaging, or a sophisticated norm for computing the distance can be used as well.

III. Experimental Activity

The experiments have been carried out on a Linux machine equipped with an Intel Core i7 vPro CPU and 16 GiB RAM.

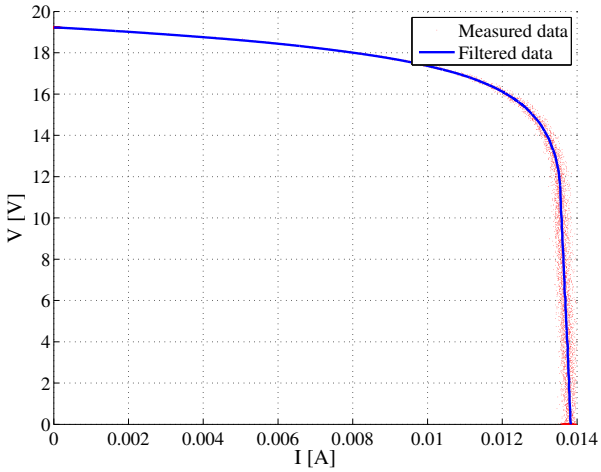


Fig. 1. A typical Voltage vs. Current curve for a solar panel. The sampled data are reported as red points, while the solid line is the curve resulting by filtering the data with a lowpass Gaussian filter.

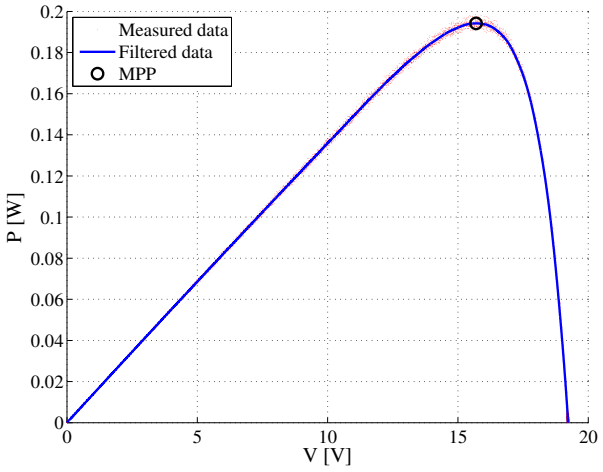


Fig. 2. A typical Power vs. Voltage curve for a solar panel. The power provided by a solar panel depends also on the electrical load applied to the panel. The working condition which allows to provide the maximum power is called Maximum Power Point (MPP) and is reported in the figure as a circle.

The simulations has been implemented and run on Matlab 2012a.

A. Dataset preprocessing

The solar panel dataset includes a set of measured voltage-current (I-V) characteristic curves (the produced power depends also on the applied load), the working temperature and solar radiation which cover most of the possible working conditions of the photovoltaic panel. The rated parameters of the panel are the following: the maximum power, $P_{\text{MAX}} = 5$ W; the voltage and the current at which maximum power is produced, $V_{\text{PM}} = 17.5$ V and $I_{\text{PM}} = 0.285$ A; the open circuit voltage, $V_{\text{OC}} = 21.3$ V; and the short circuit current, $I_{\text{SC}} = 0.31$ A. In order to explore the behavior of the panel under different working condition, a measurement campaign has been performed and starting from the I-V curves the corresponding MPP values have been estimated.

A typical I-V curve is described in Fig. 1, where the sampled data are reported as red points. It has been sampled by increasing the electrical load connected to the panel from to 0 (short circuit condition) to (virtually) infinite (open circuit condition). The sampling of each curve has been repeated every 69 seconds. Since the measurement noise, the data belonging to each curve sampling have been filtered through convolution with a Gaussian filter. The resulting I-V data have been used to compute the power provided for each sample and the MPP of the curve (as reported in Fig. 2). Each of the filtered data, joined with the temperature of the panel and the MPP of the corresponding curve constitutes a sample of the dataset to be used for the experiments.

The data has been collected from May to June 2013 for a total of 15862 curves sampled and more than 82 millions of samples.

The input domain of the resulting dataset is reported in Fig. 3, where the space of the states is represented. Since it is a 3D point cloud, several view have been used. In particular, in panels (a)–(c), the data are projected on a bidimensional subdomain, while in panel (d) the three-dimensional space occupied by the samples is rendered as the surface that encloses the large part of the samples. It can be noticed that the samples are not uniformly distributed in the input domain, since some regions are densely populated while others are empty. Hence, some correlation between the input variables can be present and can be exploited. The two views in panels (e) and (f), where the value of the MPP is depicted by the color, show that the function under study is smooth in almost all the input space, but in the region close to the open circuit condition.

Since the size of the dataset is too large, we subsampled it to obtain a representative training set and randomly distribute the remaining examples in the validation and in the testing datasets. The training set is used to set the parameters of the approximation model. The performance of the trained models on this dataset are not meaningful to assess the ability of generalization (a 1-NN always achieves zero error on the training dataset). For this task, a set of data never used in the training process have to be used. Besides, since we want to compare several models, another dataset is needed. So, we use the validation dataset to choose the best model, and the testing set to assess the performance.

We experimented several subsampling step to assess the performance of the predictors on different level of knowledge of the problem. In particular, we set the sampling step to the following values:

$$s = \{10000,\ 5000,\ 1000\} \tag{4}$$

Depending on the size of the training set, the size of the validation and the testing sets change. In Table I the dataset cardinality resulting by the the subsampling step in (4) are reported. It can be noticed that while the size of the training set changes considerably, in the experimented set-up the size of

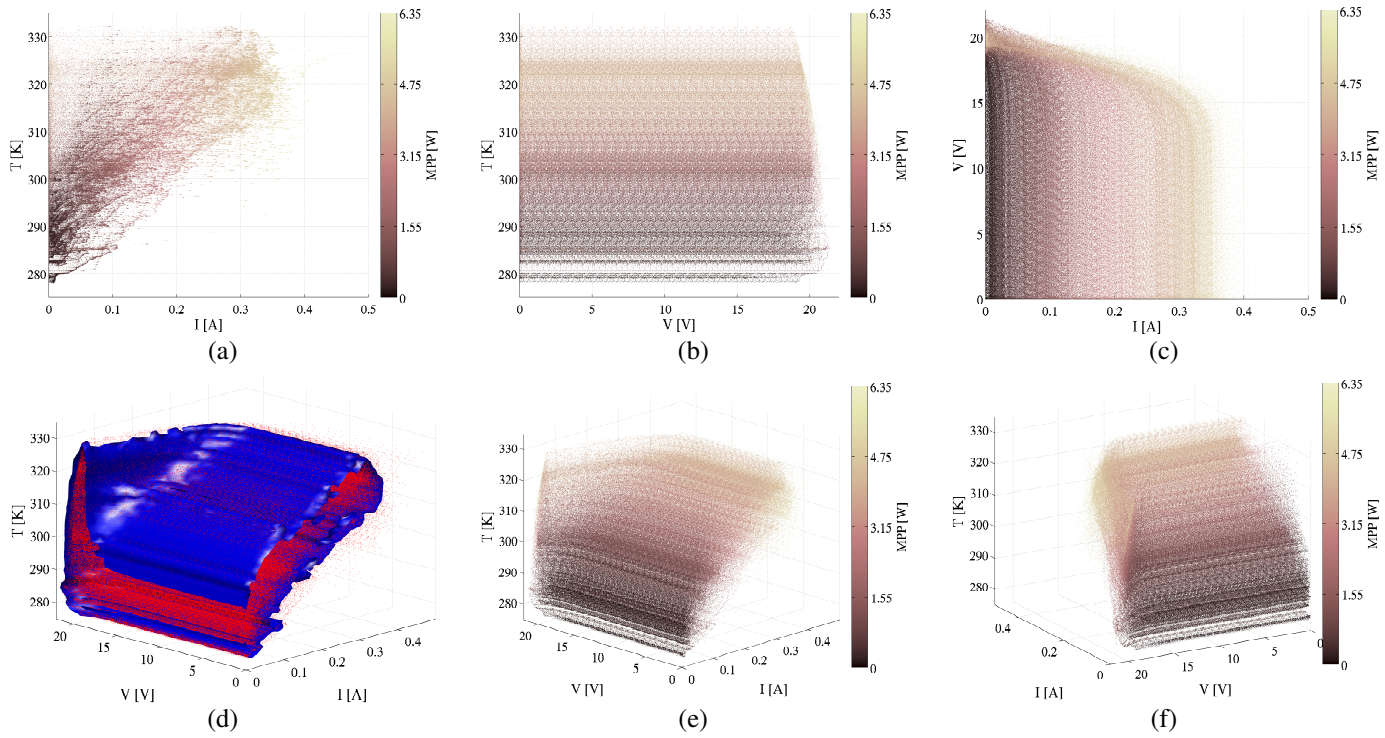


Fig. 3. The space of the states of a solar panel. In panels (a)-(c), the data are projected onto the I-T, V-T, and I-V subspaces, respectively. Here the MPP value is depicted as a color. In panel (d), the surface encloses most of the samples and is used to visualize the region effectively occupied by the data. The two views in panels (e) and (f), where the value of the MPP is depicted by the color, allow to appreciate the smoothness of the function under study in almost all the input space, but in the region close to the open circuit condition (i.e., where the voltage is high and the current is low).

TABLE I
CARDINALITY OF THE DATASETS USED IN THE EXPERIMENTS

subsampling step, s	training	validation	testing
10 000	8 268	41 333 235	41 333 235
5 000	16 535	41 329 102	41 329 101
1 000	82 675	41 296 032	41 296 031

the validation and testing sets remains substantially the same.

Since RBF, FCM, and k-NN require to evaluate distances in the input domain, the input data have been normalized using the standard deviation of each variable as normalizing factor. The normalization has been carried out using only the training data.

B. Performance Evaluation

For the evaluation of the performance, the prediction error has been measured by means of the average of the absolute error achieved on the testing set data:

$$\mathrm{Err}(f) = E(|y - f(x)|) \tag{5}$$

where $f(x)$ is the value predicted by the model for the sample $(x,\, y)$, where x is a point in the $I \times V \times T$ space, while y is the corresponding measured MPP.

In order to have a figure of merit that provides the accuracy with respect to the order of the measured value, we choose the average relative error, weighted with the relative importance of each measure:

$$\mathrm{Rel}(f) = \sum_i \frac{|y_i|}{\sum_j |y_j|} \frac{|y_i - f(x_i)|}{|y_i|} = \frac{\sum_i |y_i - f(x_i)|}{\sum_j |y_j|} \tag{6}$$

C. Prediction through FCM models

Since the starting position of the centroids is arbitrary and is usually chosen randomly, to some extent the FCM algorithm is subjected to randomness. However, for smooth function and for a small number of centroid with respect to the number of points, the effect of the randomness is negligible.

The behavior of the FCM predictor is ruled by the number of training points and the number of clusters, L. The former depends by the subsampling step, s, while L has been arbitrarily chosen in a wide range of values. In particular

- s, the subsampling step: $\{1\,000,\ 5\,000,\ 10\,000\}$;
- L, the number of clusters:
  $\{100,\ 200,\ 500,\ 1\,000,\ 2\,000,\ 5\,000\}$.

The Euclidean norm has been used to compute the distance in the input space. For each value of s, the FCM clustering has been carried out and the average output of resulting clusters has been associated to the corresponding centroid. The output of the FCM predictor has been defined as the output of the cluster which the input point belongs to. For all the training set-ups the maximum number of iterations has been set to 100.

D. Prediction through RBF models

We choose to train the RBF predictors using the hybrid learning technique. In order to improve the comparison be-

TABLE II
PERFORMANCE

Model	Err (std) [W]	Rel
FCM	0.133 (0.209)	0.158
RBF	0.0155 (0.0429)	0.0230
k-NN	0.0323 (0.0737)	0.0438

tween the computational intelligence techniques, the centroids resulting from the FCM training sessions have been used as the position of the center of the units, $\{\mu_i\}$, and width of each Gaussian, σ_i has been set as proportional to the average distance of the points in the cluster from the centroid, μ_i. Once these parameters has been chosen, the weights, $\{\beta_i\}$ can be computed as the solution of a linear system.

The hyperparameters challenged are hence:

- s, the subsampling step: $\{1\,000,\ 5\,000,\ 10\,000\}$;
- L, the number of units: $\{100,\ 200,\ 500,\ 1\,000,\ 2\,000,\ 5\,000\}$;
- r, the width proportionality factor: $\{1,\ 2,\ 3\}$.

E. Prediction through k-NN Models

The performance of a k-NN predictor depends on several hyperparameters. Since it does not requires other training process than just storing the training values, all the hyperparameters of a k-NN predictor operate in the prediction stage. In particular, the behavior of the k-NN predictor is ruled by:

- k: the number of neighbors;
- the weighting scheme: the law to assign the weights for the weighted averaging prediction;
- the norm of the input space.

The following values for the hyperparameter k have been challenged:

$$k \in [1, 15] \tag{7}$$

Three weighting schemes have been tried: equal weight, weight proportional to the inverse of the neighborhood rank, and weight proportional to the inverse of the distance. Only the Euclidean norm has been used to compute the distance in the input space. All the above described training set-ups have been experimented with the three dataset configurations characterized by s, the subsampling step: $\{1\,000,\ 5\,000,\ 10\,000\}$.

IV. RESULTS AND DISCUSSION

The test error of the challenged models are reported in Table II from which can be noted that the best performing model is the RBF network, both in term of average absolute error, Err, and relative weighted error, Rel.

In particular, k-NN and RBF outperform the FCM predictor, which achieve the best performance of 0.133 W (corresponding to a relative error of 15.8%) using $L = 5\,000$ centroids on the smallest set ($s = 10\,000$). The best performing k-NN makes use of of the largest set, $s = 1\,000$, and achieve the absolute error of 0.0323 W (Rel = 4.38%) combining the output of $k = 10$ neighbors using the inverted rank weighting scheme.

The RBF model that achieved the lowest average validation error used $L = 2\,000$ Gaussians with a width proportionality $r = 4$. Using the largest set as training set ($s = 1\,000$), it achieved an error of 0.0155 W and a relative error of 2.30%.

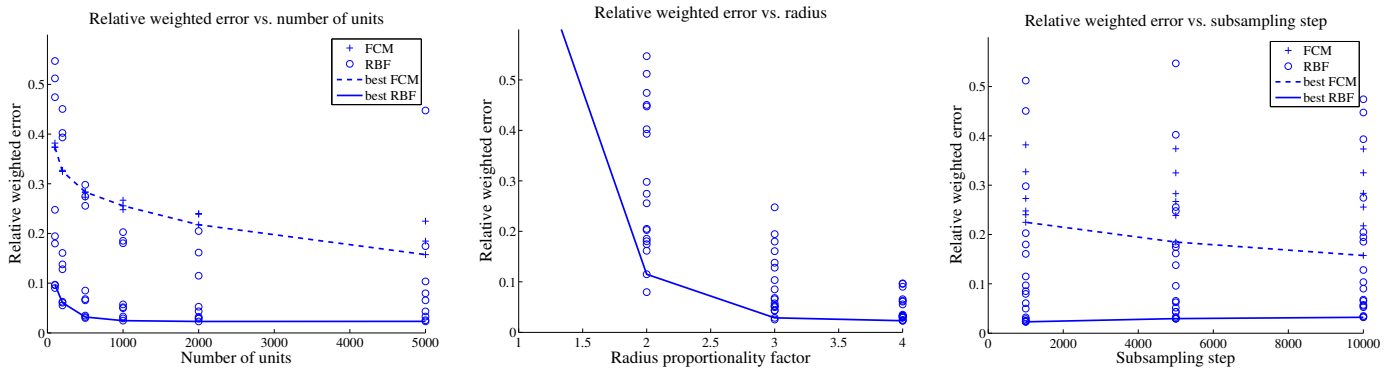
The comparison between the error achieved by FCM and k-NN shows that, despite the similarity between the operation of the two paradigms, the k-NN can exploit better the knowledge of the training set. However, the better performance of the k-NN is obtained at a higher computational cost. In fact, while the FCM makes use of only $L = 5\,000$ clusters, the k-NN stores $82\,675$ training points. This fact affects both the computational time for obtaining the output (both the models require to find the nearest centroids or training points) and the memory requirements to store the model parameters, although the computational costs can be mitigated by using a suitable data structure to store the units of the models. Besides, the computational costs for the FCM training are obviously higher than that for the k-NN, which need only to store the data. Since the best FCM model uses the smallest training set, the limitation on the training iterations can be questioned. Moreover, as the number of clusters increases. also the initial position of the centroids can affects the resulting clustering. However, when both the number of training points and the number of clusters increase, the computational time required to allow a slow convergence becomes unfeasible.

The use of the FCM clustering as processing step of the hybrid learning for the RBF models allows to compare directly the performance of the models. This perspective can also be reversed: the RBF can be considered a post-processing of the FCM clustering in order to improve its performance. In this sense, the RBF apparently improves the achievements of the FCM: results reported in Table II clearly shows that the errors are one order of magnitude smaller for the RBF and considering also the standard deviation of the absolute error, Err, it is evident that the distributions of the error are different. Moreover, the RBF makes use of only $2\,000$ units vs. the $5\,000$ of FCM. Similar considerations apply also to the k-NN and RBF comparison.

Although the RBF approximator achieves a relative weighted error 2.55%, the distribution of this error in not uniform in the I-V-T space. In Fig. 5, the distribution of the current I is studied in two cases: the testing cases are partitioned depending on their relative error, computed as the ratio between the error achieved and the measured value. The threshold of 100% has been arbitrarily chosen for distinguishing the cases where the error is small from those where the error is considered large. For this two sets, the histogram of the corresponding values of I are depicted in panels (a) and (b) respectively. Two considerations can be done: first, the number of occurrences of the large error is about 100 times smaller than the small error cases; second, the large errors occur when the current is very small (open circuit condition).

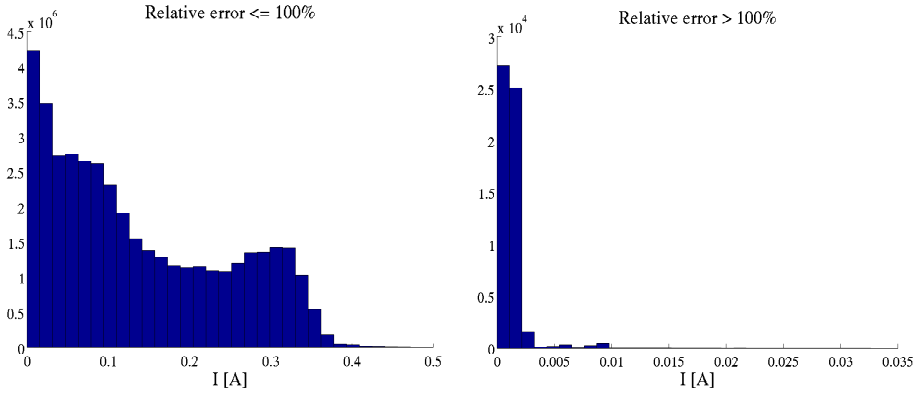
V. CONCLUSIONS

In this paper, some computational intelligence paradigms have been challenged in the task of modelling the electrical



(a) (b) (c)

Fig. 4. In panel (a), the relative weighted error with respect to the number of units of the FCM and RBF approximators is reported. Panels (b) and (c), instead, report the relative weighted error achieved by the RBF approximator with respect to the width factor and the subsampling step, respectively. The performance achieved by the different set-ups are reported using the circle for the RBF models and the cross for the FCM. The lines traces the performance of the model when the other parameters of the training are set to their best value. Hence, in panel (a) the continuous line joins the performance of the RBF approximator when $r = 4$ and $s = 1\,000$, while the dashed line describes the performance of the FCM approximator as a function of the number of clusters when $s = 10\,000$. Similarly, in panel (b) the continuous line describes the RBF performance for $s = 1\,000$ and $L = 5\,000$, while in panel (c) the continuous line refers to the RBF approximators when $r = 4$ and $L = 2\,000$, and the dashed line refers to the FCM approximator when $L = 5\,000$.



(a) (b)

Fig. 5. Distribution of the current, I, with respect to the relative error for RBF approximator.

behavior of a solar panel.

Among the challenged paradigms (FCM, RBF, and k-NN) the RBF model achieve the lowest error, which is 1.72 and 6.12 times smaller than the error of k-NN and FCM respectively.

The analysis of the condition under which the large errors occur reveals that they are limited to low current regions of the input space. Although the low current condition can be scarcely appealing for practical uses, it can be a challenge from the theoretical point of view. In order to tackle this problem, a more advanced learning strategy (e.g., in [14]) can help to improve the accuracy, while preserving the simplicity of the approach to provide good MPP estimate using basic measurements.

REFERENCES

[1] S. Ferrari, M. Lazzaroni, V. Piuri, A. Salman, L. Cristaldi, and M. Faifer, "Computational intelligence models for solar radiation prediction," in *Proceedings of I2MTC 2013 (2013 IEEE International Instrumentation and Measurement Technology Comference)*, 2013, pp. 757–762.

[2] S. Ferrari, M. Lazzaroni, V. Piuri, L. Cristaldi, and M. Faifer, "Statistical models for solar radiation prediction," in *Proceedings of I2MTC 2013 (2013 IEEE International Instrumentation and Measurement Technology Comference)*, 2013, pp. 1734–1739.

[3] L. Cristaldi, M. Faifer, S. Ierace, M. Lazzaroni, and M. Rossi, "An approach based on electric signature analysis for photovoltaic maintenance," in *IEEE International Energy Conference and Exhibition (ENERGYCON 2012)*, Sep. 2012, pp. 1–8.

[4] S. Ferrari, M. Lazzaroni, V. Piuri, A. Salman, L. Cristaldi, M. Rossi, and T. Poli, "A data approximation based approach to photovoltaic system maintenance," in *Environmental Energy and Structural Monitoring Systems (EESMS), 2012 IEEE Workshop on*, Sep. 2013.

[5] L. Cristaldi, M. Faifer, M. Rossi, and S. Toscani, "A new approach to maximum power point tracking for photovoltaic panels," in *Clean Electrical Power (ICCEP), 2013 International Conference on*, 2013, pp. 461–465.

[6] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, no. 2, pp. 281–294, 1989.

[7] T. Cover and P. Hart, "Nearest neighbor pattern classification," *Information Theory, IEEE Trans. on*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[8] J. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact, well separated clusters," *J. Cybernetics*, no. 3, pp. 32–57, 1974.

[9] J. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[10] L. Fausett, *Fundamentals of Neural Networks: Architectures, Algo-*

rithms, and Applications, ser. Prentice Hall international editions. Prentice-Hall, 1994.

[11] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York, NY, USA: Oxford University Press, Inc., 1995.

[12] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, pp. 213–225, 1991.

[13] T. Poggio and F. Girosi, "Network for approximation and learning," in *Proceedings of the IEEE*, vol. 78, no. 9, 1990, pp. 1481–1497.

[14] N. A. Borghese and S. Ferrari, "Hierarchical RBF networks and local parameter estimate," *Neurocomputing*, vol. 19, no. 1–3, pp. 259–283, 1998.

[15] S. Ferrari, F. Bellocchio, V. Piuri, and N. A. Borghese, "A hierarchical RBF online learning algorithm for real-time 3-D scanner," *IEEE Trans. on Neural Networks*, vol. 21, no. 2, pp. 275–285, Feb. 2010.

[16] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, Dec. 2006.