2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Optimal Detection of New Classes of Faults by an Invasive Weed Optimization Method

Roozbeh Razavi-Far, Vasile Palade and Enrico Zio

***Abstract*— Proper detection of unknown patterns plays an important role in diagnosing new classes of faults. This can be done by incremental learning of novel information and updating the diagnostic system by appending newly trained fault classifiers in an ensemble design.**

We consider a new-class fault detector previously developed by the authors and based on thresholding the normalized weighted average of the outputs ($NWAO$) of the base classifiers in a multi-classifier diagnostic system. A proper tuning of the thresholds in the $NWAO$ detector is necessary to achieve a satisfactory performance. This is done in this paper by specifically introducing a performance function and optimizing it within the necessary trade-off between new class false alarm and new class missed alarm rates, by means of an Invasive Weed Optimization (IWO) algorithm.

The optimal $NWAO$ detector is tested with respect to a set of simulated sensor faults in the doubly-fed induction generator (DFIG) of a wind turbine.

I. Introduction

THE growing demand for safety, reliability and higher efficiency in industrial systems is motivating an increasing interest in data-driven diagnostic systems [1]. Most of these data-driven systems apply computational intelligence methods for detecting and diagnosing faults [2–4].

Most of the fault classifiers are built based on time-series data of various feature signals in static environments, and their performance is highly dependent on the available data quantity and distribution [1].

Static fault classifiers are not valid for decision making in dynamic environments, where the datasets become available successively, over a period of time. In these circumstances, a fault classifier should be able to incrementally update and learn the novel information, as new data become available, while preserving the obtained knowledge from the preceding data [5]. One way is to use an ensemble of fault classifiers and update the ensemble in an incremental fashion without discarding the previously trained fault classifiers [5], which allows to learn the new relations between the input signals and output classes in the new operational regions.

New class faults are inevitable in most dynamic systems, since in practice, it is not feasible to have datasets containing patterns of all the possible faulty classes. Thus, it is possible that the subsequent datasets contain patterns of new classes of faults that do not exist in the preceding datasets. Albeit the ensemble of fault classifiers is more confident compared with the single fault classifier [6] and capable of incremental learning [5], it is doomed to misclassify patterns from classes on which it was not trained.

The problem of diagnosing new class faults was addressed in [7, 8] by means of a dynamic weighting ensemble, called $Learn^{++}.NewClass(NC)$ [9]. This algorithm learns the new classes due to a voting mechanism, called dynamically weighted consult and vote ($DW-CAV$) [9].

The dynamic weighting ensemble (DWE) of fault classifiers was able to incrementally learn and diagnose multiple new classes of faults in a Boiling Water Reactor (BWR) [7, 8]. The DWE classification module has also been successfully used along with the multiple observers scheme to diagnose multiple classes of new faults in the sensors of the doubly-fed induction generator (DFIG) of a wind turbine [10–12].

In [8], an unknown class detector has been devised based on thresholding the Normalized Weighted Average of Outputs ($NWAO$) of the base classifiers of the DWE, which detects the patterns of unseen classes in upcoming datasets. The performance of the DWE diagnostic system depends on some preset parameters that need to be tuned automatically. The key parameters are the low and high thresholds of the $NWAO$ detector.

In this work, in order to tune these thresholds automatically, a proper multi-objective performance function is defined. Then, a bio-inspired numerical optimization algorithm, called Invasive Weed Optimization (IWO) [13], is used to find the most suitable set of parameters that minimizes the performance function.

The rest of this paper is organized as follows. Section II presents a brief description of the diagnostic system. To optimize the diagnostic system, a proper performance function is proposed in Section III along with the problem formulation. The IWO algorithm, its properties and pseudocode are presented in Section IV. Section V presents some results of the optimized diagnostic system in detecting unknown patterns of multiple classes of faults in the wind turbine application. Conclusions are drawn in Section VI.

II. The Diagnostic System

A fault classifier maps each pattern of the input vector (i.e., generated residuals by means of multiple observers) to one of the pre-assigned available classes of faults or the fault-free

Roozbeh Razavi-Far is with the Department of Energy, Politecnico di Milano, via Ponzio 34/3, 20133 Milan, Italy (emails: roozbeh.razavi@gmail.com; roozbeh.razavi@mail.polimi.it).

Vasile Palade is with the Faculty of Engineering and Computing, Coventry University, Priory Street, Coventry, United Kingdom (email: vasile.palade@coventry.ac.uk).

Enrico Zio is with the Systems Science and the Energetic Challenge, European Foundation for New Energy-Electricité de France, Ecole Centrale Paris and Supelec, Paris, 92295 Chatenay-Malabry Cedex, France, Department of Energy, Politecnico di Milano, via Ponzio 34/3, 20133 Milan, Italy (emails: enrico.zio@polimi.it; enrico.zio@ecp.fr; enrico.zio@supelec.fr).

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

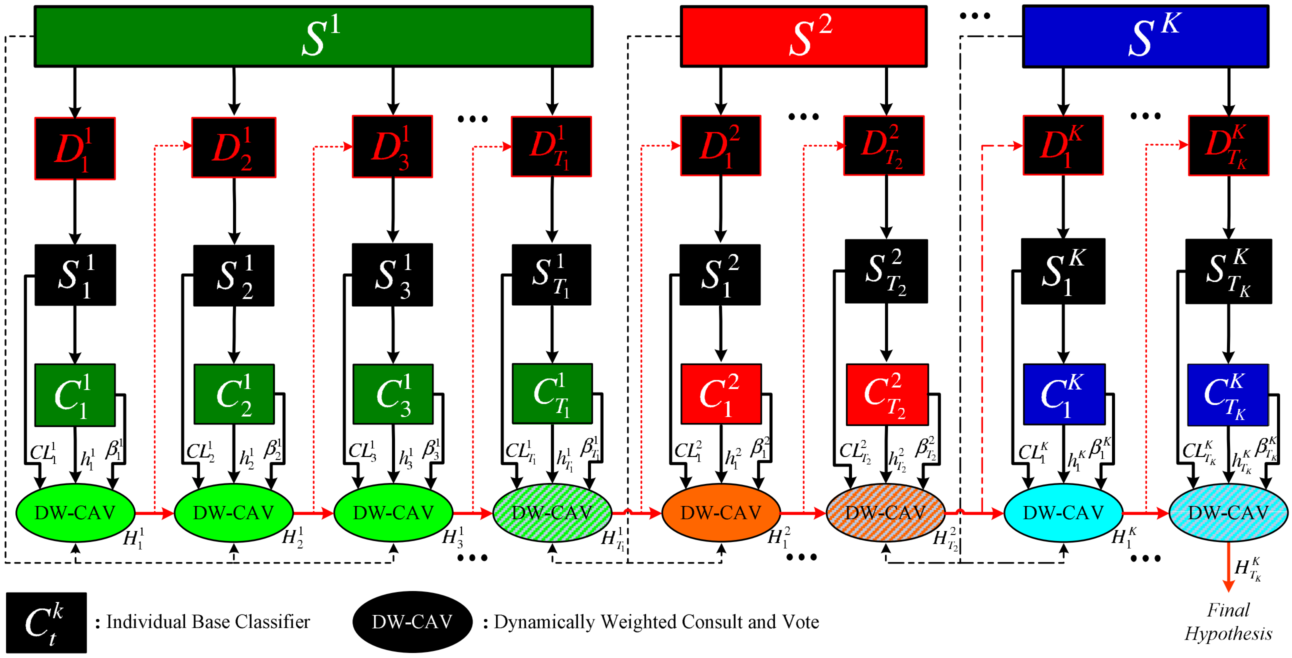

Fig. 1. Block diagram of the Dynamic Weighting Ensemble algorithm [11].

case [10, 11]. Here, the diagnostic system is the dynamic weighting ensemble of fault classifiers [8, 10, 11], that is able to dynamically diagnose the new classes of faults. The DWE algorithm along with the $DW-CAV$ subroutine and $NWAO$ detector are briefly explained in this section, as they are the prerequisites for the problem statement in this paper. The detailed explanation and pseudocodes can be found in [8, 11].

A. Dynamic Weighting Ensemble

Figure 1 shows the block diagram of the DWE algorithm.

The DWE algorithm generates and trains a new member of the ensemble $\mathcal{E}^k$ (i.e., a preset number of MultiLayer Perceptron MLP-based classifiers T_k) for dataset S^k, $k = 1, \ldots, K$.

It then assigns a normalized weight w_t^k, $t = 1, \ldots, T_k$ (see [9, 11]) to each pattern of the S^k, to form a weight distribution D_t^k, which extracts a training subset S_t^k from S^k for the corresponding classifier C_t^k. The initial distribution D_1^1 is uniform, providing an equal probability for all patterns of S^1 to be selected for S_1^1.

The normalized error of the trained classifier β_t^k is computed and then fed to the $DW - CAV$ subroutine, along with the class labels CL_t^k (i.e., fault numbers) of the patterns used to train the classifier. The $DW - CAV$ combines all hypotheses h_t^k generated thus far, evaluates all patterns of S^k and computes the normalized error of the combined hypothesis error B_t^k $(0 < B_t^k < 1)$ [9, 11].

The DWE uses the B_t^k, computed by $DW - CAV$, to update the weight $w_t^k(i)$, $i = 1, \ldots, m_k$ of each pattern (m_k is the number of patterns in S^k). The weights are iteratively updated in a way that the weights of correctly classified patterns are decreased by a multiple of B_t^k, increasing the probable collection of the misclassified patterns (as well as patterns of newly introduced classes) into the following training subset. Thanks to this iterative update, the DWE focuses progressively on the misclassified patterns of the current dataset and, when a new dataset becomes available, it focuses on the fraction of patterns of unseen classes [9].

B. Dynamically Weighted Consult and Vote

The $DW - CAV$ subroutine receives β_t^k, h_t^k and CL_t^k as inputs, assigns a voting weight $W_t^k = log(1/\beta_t^k)$ to each classifier and, then, for each pattern, calculates the class-specific confidence (see [9, 11]), which allows the classifiers to consult with each other (i.e., cross-reference their decisions with respect to the CL_t^k) and dynamically adjust their voting weight [9]. The final decision is, then, obtained as the weighted majority voting of all classifiers.

The fault classifiers of the former ensemble member are doomed to misclassify the patterns of the new class fault and outvote the decisions of the newly trained classifiers, which see the patterns of the new class fault in their training sessions. This delays the incremental learning of the patterns of new class faults until an adequate number of fault classifiers are added into the ensemble, but thanks to the consultation mechanism of the $DW - CAV$, the generation of unnecessary fault classifiers can be avoided [9].

C. MLP-Based Fault Classifiers

Any supervised classifier with controllable weakness (i.e., to guarantee a satisfactory diversity) can be used to form the ensemble. In this work, the widely-used MLP-based neural

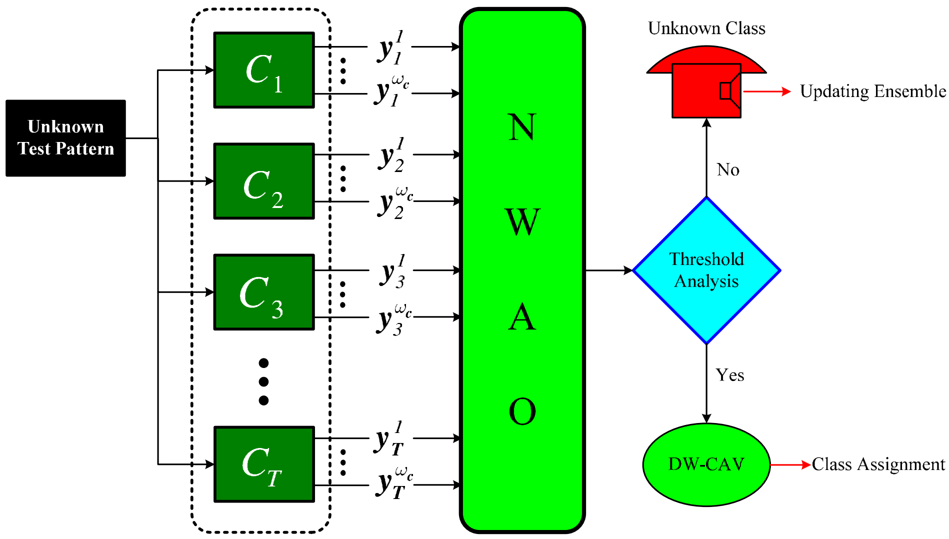

Fig. 2. The new class fault detection and diagnostic scheme [8].

networks are used, whose weakness can be controlled by tuning the training parameters (e.g. error goal or size of the network) [9, 14]. Each MLP is a three layer network in which the number of neurons in the input layer is equal to the number of features or signals used for the diagnosis, the number of neurons in the output layer is equal to the number of faulty classes (i.e, this can vary for each dataset) and the number of neurons in the hidden layer is properly chosen in a trial-and-error procedure [11].

D. Unknown Class Detector

It is crucial for the diagnostic system to be able to detect new classes of faults. Typically, patterns of some classes of faults are needed to train an experimental fault classifier. However, new classes of faults (i.e., not used in the training) are inevitable and emerge during the system life. In these circumstances, it is necessary for the diagnostic system to detect the new classes of faults while keeping the ability of correctly discriminating the previously trained classes of faults [15, 16].

The DWE-based diagnostic algorithm proposed in [8] dynamically learns and diagnose the patterns of new classes of faults. The DWE algorithm acts in a supervised way and, thus, all class labels (i.e., fault numbers) should be known in advance.

Besides, adding a new ensemble member each time a new dataset emerges, significantly increases the complexity of the system and, thus, to control the proliferation of classifiers, a new ensemble member can be added only if the newly emerged dataset contains some patterns of new classes. For these reasons, it is necessary that, the DWE-based diagnostic system can detect the new classes of faults in the upcoming datasets. The detected new class patterns are discriminated from the other classes as unknown, until a correct label is assigned to them [8]. The patterns of new faults (i.e., unknown patterns) have been detected by resorting to the Normalized Weighted Average of Outputs ($NWAO$) of the base classifiers of the DWE [8]. The $NWAO$ detects the presence of the unknown patterns based on preset thresholds.

Figure 2 shows the overall updating procedure for the new class fault detection and diagnosis. The major steps to detect an unknown pattern (i.e., a pattern of a new fault) are as follows [8]:

- The outputs of the MLP networks. Consider that a base classifier of the ensemble is trained on patterns of ω_c number of faults; then, the output of the $t-th$ fault classifier C_t of the ensemble is a vector of size ω_c. Each member of the output vector y_t^j represents the degree of confidence in the assignment of the test pattern to the $j-th$ class, $j=1,...,\omega_c$. If none of the output values y_t^j, $j=1,...,\omega_c$ takes a value close to 1, the test pattern likely belongs to a new class fault.
- The agreement between the fault classifiers of the ensemble. It is expected that the T different base classifiers of the ensemble assign the test patterns of a new class fault to different faulty classes.

Considering these two steps, a heuristic index has been proposed in [8] to detect an unknown pattern (i.e., a pattern of a new class fault). For the $j-th$ class, the normalized weighted average of all the ensemble outputs $NWAO^j$ is defined as follows:

$$NWAO^j = \frac{\sum_{t=1}^{T} W_t y_t^j}{\sum_{t=1}^{T} W_t} \qquad j=1,...,\omega_c \qquad (1)$$

where y_t^j is the $j-th$ output of the $t-th$ MLP-based fault classifier and W_t stands for the weight assigned to the $t-th$ classifier by the DWE algorithm. In [8], to detect an unknown pattern, the $NWAO^j$ values are compared with two preset high σ_h and low σ_l thresholds. A test pattern is assigned to a new class fault if the maximum $NWAO^j$

value is less than σ_h and, simultaneously, there exists another $NWAO^{j'}$, $j' \neq j$ with a value larger than σ_l.

In other words, the area between two preset thresholds is considered to be a low confidence area and the presence of the $NWAO$ values in the low confidence area activates an alarm indicating the detection of an unknown class.

A test pattern is sent to the current ensemble of classifiers, $C_t, t = 1, ..., T$, and their $NWAO^j$ values are calculated for all classes, $j = 1, ..., \omega_c$; forming a set $\mathbb{S} = \{NWAO^j\}_{j=1}^{\omega_c}$. By comparing the values of $\mathbb{S}$ with the thresholds σ_h and σ_l, a decision can be made on whether the test pattern belongs to one of the classes used for training or to a new fault (i.e., $\neg\exists\ NWAO^j \in \mathbb{S} :\ NWAO^j \succ \sigma_h\ \wedge\ \exists\ NWAO^{j', j' \neq j} \in \mathbb{S} :\ NWAO^{j'} \succ \sigma_l$). In the former case, the test pattern is classified by means of the $DW-CAV$ subroutine, whereas in the latter case an unknown class alarm is activated (see Figure 2).

The DWE-based diagnostic system [8] is updated only after a certain number of alarms (i.e., unknown patterns), to avoid an unnecessary updating due to false alarms. The number of ignorable alarms can be determined based on the application and criticality of missed and false new class alarms with respect to system safety and performance [8]. After the emergence of several unknown (i.e., new classes of faults) patterns, one can assign a label (i.e., fault number) to them, and update the DWE-based diagnostic system by adding newly trained classifiers to the ensemble.

III. Problem Formulation

A proper tuning of the $NWAO$ detector (i.e., the high σ_h and the low σ_l thresholds) is of paramount importance to detecting the new classes of faults, control the incremental learning of the DWE and avoid proliferation of unnecessary updates.

This can be done by defining a proper objective function and, then, finding a proper method to tune the thresholds.

The objective function contains several performance indices, e.g., the trade-off between the new class false alarm and new class missed alarm rates. Tuning the thresholds is the task of finding optimal values of thresholds that optimize the objective function.

Figure 3 shows the $NWAO$ profile that helps to define the necessary performance indices. Considering a dataset of m patterns of ω_c classes, each $NWAO^j$ is an m-dimensional vector, starting from $NWAO^j_1$ to $NWAO^j_m$. Suppose that $NWAO^j_i$ corresponds to the pattern at which the new class of fault occurs. The indices that form the objective function are:

1) The New Class False Alarm Rate: F_f is defined as follows:

$$F_f = \frac{N_{\{1-i\}}}{i-1} \tag{2}$$

where the numerator $N_{\{1-i\}}$ stands for the number of alarms activated in the interval between the $1-st$ pattern and the $i-th$ pattern, and the denominator is the total number of patterns between the $1-st$ pattern and the $i-th$ pattern.

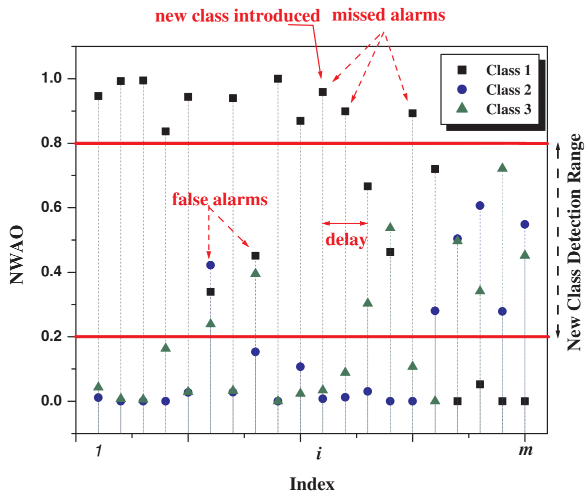

Fig. 3. Solid squares, circles, and triangles represent the $NWAO$ values for each class (i.e., the system is trained with three classes). The red lines stand for the high σ_h and the low σ_l thresholds. The range between the two thresholds is the low confidence area. The presence of the $NWAO^j$ values in the low confidence area activates the new class alarm.

2) The New Class Missed Alarm Rate: F_m is calculated as:

$$F_m = 1 - \frac{N_{\{i-m\}}}{m-i} \tag{3}$$

where the numerator $N_{\{i-m\}}$ stands for the number of alarms activated in the interval between the $i-th$ pattern and the $m-th$ pattern, and the denominator is the total number of patterns between the $i-th$ pattern and the $m-th$ pattern.

3) The New Class Detection Delay: F_d is the number of patterns in the interval from the $i-th$ pattern (i.e., a pattern corresponding to a new class fault) to the first next pattern detected as unknown.

The multi-objective function for the optimal tuning of the thresholds can be defined as a weighted sum of the above defined indices, as follows:

$$F = \xi_f F_f + \xi_m F_m + \xi_d F_d \tag{4}$$

where the $\xi_{(.)}$s are positive weights and can be selected by the user. In this work, the weights ξ_f and ξ_m are equal to 1, since their corresponding indices F_f and F_m take value in the $[0, 1]$ interval. To be able to consider F_d in the performance function, the weight of ξ_d should be normalized.

Here, two different multi-objective functions are defined:

$$F_\alpha = F_f + F_m \tag{5}$$

$$F_\beta = F_f + F_m + \frac{1}{m-i} F_d \tag{6}$$

The first multi-objective function F_α is only based on the trade-off between the new class false alarm and the new

INPUTS:

$P_0 \leftarrow 2$; is a limited number of initial seeds

$\mathcal{S}_{min} \leftarrow 0$ and $\mathcal{S}_{max} \leftarrow 5$; are the minimum and maximum possible seeds production

$iter_{max} \leftarrow 200$; indicates the maximum allowed number of iteration cycles

$\sigma_{initial} \leftarrow 1$ and $\sigma_{final} \leftarrow 0.05$; denote the pre-defined initial and final standard deviations

$n \leftarrow 3$; is the nonlinear modulation index

$P_{max} \leftarrow 30$; is the maximum population size

DEFINITIONS:

$\mathcal{F}_i$ is the fitness of the $i - th$ plant

$\mathcal{F}_{min}$ and $\mathcal{F}_{max}$ stand for the lowest and highest fitness values in the weed population

GENERATE a random population of P_0 individuals from a set of feasible solutions $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_{P_0}\}^T$

for *iter* $= 1$ *to* $iter_{max}$ **do**

EVALUATE the fitness function for each individual in Γ

COMPUTE the maximum and minimum fitness in the colony $\mathcal{F}_{max}$ and $\mathcal{F}_{min}$

for *each individual* γ_i **do**

COMPUTE the number of seeds for γ_i, $\mathcal{S}_i = \lfloor (\mathcal{F}_i - \mathcal{F}_{min})(\mathcal{S}_{max} - \mathcal{S}_{min}) / (\mathcal{F}_{max} - \mathcal{F}_{min}) + \mathcal{S}_{min} \rfloor$

RANDOMLY distribute seeds over the search space with normal distribution $\mathcal{N}\left(0, \sigma_{iter}^2\right)$ around the parent plant γ, with zero mean and an adaptive standard deviation:

$$\sigma_{iter} = \sigma_{final} + (\sigma_{initial} - \sigma_{final})(iter_{max} - iter)^n / (iter_{max})^n$$

ADD the generated seeds to the solution set, Γ

end

if $(|\Gamma| = P) > P_{max}$ **then**

SORT the population Γ in descending order of their fitness

TRUNCATE population of weeds with smaller fitness, so-called competitive exclusion, until $P = P_{max}$

end

end

BEST solution is the plant γ_{best} with minimum fitness in the last population

Fig. 4. The pseudo-code for the IWO algorithm [17].

class missed alarm rates. On the contrary, the second multi-objective function F_β will also considers reducing the new class detection delay by appending the F_d.

As a result of the choice of these two different multi-objective functions F_α and F_β, two different invasive weed optimization tasks have been performed: in the former, the focus is to optimize the position of the thresholds in the $\mathcal{D}$-dimensional feature space; in the latter, a mechanism has been also devised to reduce the new class detection delay.

IV. INVASIVE WEED OPTIMIZATION (IWO)

The IWO algorithm is a bio-inspired numerical optimization algorithm that simulates the behavior of weeds in nature when colonizing and finding a suitable place for growth and reproduction [13].

Since its primitive development for the optimization and tuning of a robust controller, the IWO algorithm has been extensively used in a variety of practical applications [17–21].

In this work, the invasive weed optimization algorithm is used to find the optimal values of the thresholds to detect and isolate the new classes of faults. For the optimal design, a proper multi-objective function is defined.

A. The Invasive Weed Optimization Algorithm

The key terms of the IWO algorithm are firstly defined [13, 22].

Seed: each individual in the colony, that includes a value for each variable in the optimization problem prior to fitness evaluation.

Fitness: a value which represents the merit of the solution for each seed.

Weed/Plant: each evaluated seed grows to a flowering plant or weed in the colony. Therefore, growing a seed to a plant corresponds to evaluating an individual's fitness.

Colony: the set of all agents or seeds.

Population size: the number of plants in the colony.

Maximum weed population: a predefined parameter that represents the maximum allowed number of weeds in the colony posterior to fitness evaluation.

A pseudocode of the IWO algorithm is given in Figure 4 [17]. Further details about the main steps of the IWO algorithm can be found in [13, 22].

To perform the IWO algorithm, initially, the number of parameters that need to be optimized has to be defined, (hereafter denoted by $\mathcal{D}$), consequently, for each parameter in the $\mathcal{D}$-dimensional search space, a minimum and maximum values are assigned. Here, the number of parameters $\mathcal{D}$ is equal to 2, i.e., the high σ_h and the low σ_l detection thresholds. Then, a limited number P_0 of initial seeds, $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_{P_0}\}^T$ are being randomly spread through the defined search space; subsequently, each seed catches a random position in the 2-dimensional space.

Next, the flowering plants are ranked based on their fitness values relative to others and, subsequently, a number of new seeds are produced, depending on the ranking: i.e., the plants with higher fitness value, which are more adapted to the environment, can produce more seeds that can solve the problem better [22]. The generated seeds are being randomly spread in the 2-dimensional space by using normally distributed numbers with zero mean and adaptive standard deviation, σ_{iter}, which is computed adaptively in each iteration as follows:

$$\sigma_{iter} = \sigma_{final} + (\sigma_{initial} - \sigma_{final}) \frac{(iter_{max} - iter)^n}{(iter_{max})^n} \quad (7)$$

where $\sigma_{initial}$ and σ_{final} stand for the user-defined initial and final standard deviations, correspondingly. n indicates the nonlinear modulation index and $iter_{max}$ denotes the pre-defined maximum allowed number of iterations [13].

The adaptive standard deviation equation shows that the σ_{iter} can be reduced from the $\sigma_{initial}$ to the σ_{final} values with different velocities in accordance with the chosen nonlinear modulation index, n. Thanks to the high value of initial standard deviation $\sigma_{initial}$, the algorithm can explore the whole search space. Then, the standard deviation σ_{iter} is gradually reduced with increasing the number of iterations. This gradual reduction guarantees to preserve only fitter plants and to discard plants with lower fitness, and forces the algorithm to focus around the local minima to find the global optimum. The generated seeds, along with their parents, are taken into account as the potential solutions for the following population.

This fast reproduction mechanism increases the population size to its pre-defined maximum value P_{max} and, after a few iterations, activates a competitive exclusion mechanism to eliminate the plant with poor fitness. The best survived plants generate new seeds based on their fitness rank in the colony and, consequently, the algorithm is repeated until the termination criterion has been reached.

V. Experimental Results

Here, the IWO algorithm is used to tune the thresholds of the $NWAO$ detector, minimizing the multi-objective functions F_α and F_β with respect to the simulated sensor faults of a DFIG-based wind turbine [10, 11]. The DFIG dynamics and notations are not described here for the sake of conciseness (the reader can refer to [11] for the detailed presentation). The simulations are performed around the nominal operating condition following the same reference values, and the operating conditions in [10, 11].

A. Design of the diagnostic system

Fault diagnosis is performed in two major steps: the residual generation and the fault classification. Firstly, residual signals reflecting faults in the DFIG system are generated from sampled sensor measurements and command inputs. The generated residuals have zero mean in the fault-free case, and some of them are subject to a change in the mean upon occurrence of a sensor fault. The generated residuals are then evaluated by the DWE algorithm, in order to classify the faults.

The residual generator (completely described in [11]) contains multiple observers and complementary filters in three integrated sub-modules to detect all possible classes of faults in the rotor and the stator sensors. These multiple observers create a set of residuals $(r_1, \ldots, r_9)$ that are robust against modeling uncertainties and change in the operating points, but sensitive to a subset of faults [10, 11]. The generated residuals are then resampled (down-sampled) and fed to the DWE algorithm. Each residual is a two-dimensional vector which yields a feature space of 18 input signals for the DWE-based fault classifier.

The whole simulation contains 10 different classes of faults: ff stands for the fault-free case, f_1 to f_3 stand for faults in the stator voltage sensors for phases a, b and c, respectively, f_4 to f_6 indicate faults in the stator current sensors for phases a, b and c, respectively, and f_7 to f_9 are faults in the rotor current sensors for phases a, b and c, respectively.

Here, three steps of simulations, including a different subset of faults at each step, have been considered to form the datasets for incremental learning. By performing each step of the simulation, a set of residual data is generated. This forms three residual datasets S^1, S^2 and S^3. More specifically, each dataset is formed with the residual data patterns of the fault-free case ff and a subset of faults.

The first step of the simulation forms the first residual dataset S^1, including patterns of the fault-free case ff and three faults (classes ff, f_1-f_3). Upon emergence of S^1, the DWE algorithm creates an ensemble of ten classifiers $\mathcal{E}^1$, each one trained on a different subset of the available dataset S^1, which is drawn according to an iteratively updated distribution.

The second step of the simulation generates S^2, which is made of different residual patterns of the ff case and six classes of faults (classes $f_1 - f_6$). The emergence of the S^2 introduces three new classes of faults to the DWE algorithm. Then, S^2 is tested with the base classifiers of the current ensemble $\mathcal{E}^1$. The detection of unknown patterns (i.e., patterns of new classes of faults) in the S^2, by means of the $NWAO$ detector, leads to the incremental update of the diagnostic system to $\mathcal{E}^2$, i.e., the DWE algorithm appends ten newly trained classifiers to the ensemble, each one trained on a different subset of the current dataset S^2.

Finally, the third step of simulation generates S^3 including residual patterns of the ff case and nine faults (classes $f_1 - f_9$), thus introducing three additional classes of faults.

TABLE I

NUMBER OF RESIDUAL PATTERNS IN EACH DATASET, FOR EACH CLASS.

Dataset	Number of Patterns										
	Total	ff	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
S^1	280	70	70	70	70	-	-	-	-	-	-
S^2	609	72	72	72	72	107	107	107	-	-	-
S^3	1251	72	72	72	72	107	107	107	214	214	214

Similarly, the newly introduced dataset S^3 is tested with the base classifiers of the current ensemble $\mathcal{E}^2$. The detection of unknown patterns in the S^3 by means of the $NWAO$ detector leads to the incremental update of the diagnostic system to $\mathcal{E}^3$. The DWE algorithm appends another ten new classifiers to the ensemble, each one trained on a different subset of the current dataset S^3.

A summary of the datasets characteristics is reported in Table 1. The dynamic weighting ensemble algorithm successfully classifies the multiple classes of faults, including new classes. The detailed explanation of the fault classification results is not reported here for the sake of conciseness. The fault classification performances are reported in [11].

B. Tuning the thresholds

Figure 5 presents the block diagram of the optimal tuning of the thresholds σ_h and σ_l, by means of the IWO algorithm.

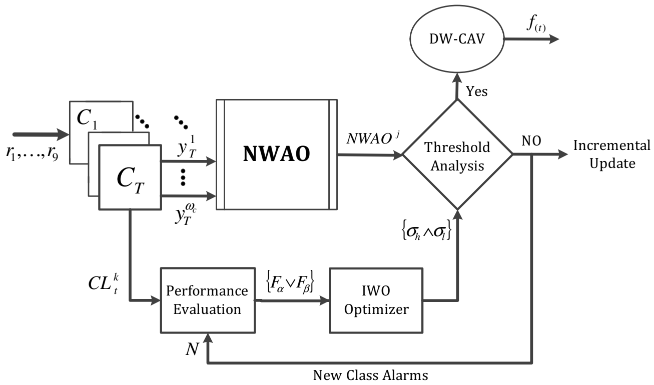

Fig. 5. Block diagram of the optimal tuning of the $NWAO$ thresholds for new class fault detection.

Here, two incremental updates occur corresponding to the test of $\mathcal{E}^1$ ($\mathcal{E}^2$) with respect to the patterns of S^2 (S^3). It is important to tune the thresholds, once the patterns of new classes of faults become available.

In the first tuning procedure, the IWO algorithm finds the optimal values of the thresholds σ_h and σ_l, on the normalized weighted average of the output values of $\mathcal{E}^1$ with respect to the patterns of S^2, while minimizing the multi-objective function F_α (i.e., minimizing the number of new class false and missed alarms). This procedure is repeated for the multi-objective function F_β (i.e., also reducing the new class detection delay).

In the second tuning procedure, the IWO algorithm finds the optimal values of the thresholds on the normalized weighted average of the output values of $\mathcal{E}^2$ with respect to the patterns of S^3, while minimizing the multi-objective function F_α (i.e., minimizing the number of new class false and missed alarms). This procedure is also repeated for the function F_β.

The IWO parameters are set as shown in Figure 4. It is necessary to bound the thresholds within the interval $[0, 1]$, following the same range of the $NWAO$ values. Thus, the search space of the IWO algorithm is limited in a way that the seed with a value larger (smaller) than one (zero) are clamped to one (zero). The tuning results obtained with each function in each step are presented in the following Tables.

TABLE II

THE PERFORMANCE INDICES OF F_α AND OPTIMAL THRESHOLDS.

	F_f	F_m	F_d	F_α	σ_h	σ_l
$\mathcal{E}^1$ tested on S^2	0.021	0	-	0.021	0.815	0.183
$\mathcal{E}^2$ tested on S^3	0.016	0.06	-	0.076	0.803	0.191

TABLE III

THE PERFORMANCE INDICES OF F_β AND OPTIMAL THRESHOLDS.

	F_f	F_m	F_d	F_β	σ_h	σ_l
$\mathcal{E}^1$ tested on S^2	0.021	0	0	0.021	0.815	0.183
$\mathcal{E}^2$ tested on S^3	0.016	0.057	0.001	0.074	0.807	0.190

Tables 3 and 4 report the optimal set of threshold values for each test, which minimizes the number of new class false and missed alarms. The minimum performances along with the values taken by each of the performance indices are also reported. In order to reduce the risk of new class false and missed alarms, the thresholds should be fixed according to their optimal values to guarantee a satisfactory low number of new class false and missed alarms.

Tables 3 and 4 show that the optimal thresholds with respect to the two multi-objective performance functions (i.e., F_α and F_β), take similar values. This is due to the type of simulated faults considered, which are additive step-like faults. The impact of the F_d performance index is expected to be more significant in the presence of drift-like faults, which will be investigated in future research.

VI. CONCLUSION

In this work, an invasive weed optimization algorithm has been used to identify an optimal set of threshold values for new class fault detection. The detection of the new classes of faults was based on thresholding the normalized weighted average of the outputs of the base classifiers in the diagnostic ensemble system.

A proper multi-objective performance function was defined as a trade-off between the new class false alarm and new class missed alarm rates, and the new class detection delay has also been taken into account as a complementary performance index. The IWO algorithm tunes the thresholds in a way that minimizes the multi-objective performance function. The method has been applied for the diagnosis of sensor faults in a DFIG-based wind turbine. The simulation results showed a proper tuning of the thresholds, as proven by the achieved performance.

References

[1] P. Baraldi, R. Razavi-Far, and E. Zio, "Bagged ensemble of FCM classifier for nuclear transient identification," *Ann. Nucl. Energy*, vol. 38, pp. 1161–1171, 2011.

[2] M. Embrechts and S. Benedek, "Hybrid identification of nuclear power plant transients with artificial neural networks," *IEEE Trans. on Ind. Elec.*, vol. 51, pp. 686–693, 2004.

[3] R. Razavi-Far, H. Davilu, V. Palade, and C. Lucas, "Neuro-fuzzy based fault diagnosis of a steam generator," in *7th IFAC Conf. on Fault Detection, Supervision and Safety of Technical Processes*, Barcelona, Spain, 2009, pp. 1180–1185.

[4] R. Razavi-Far, H. Davilu, V. Palade, and C. Lucas, "Model-based fault detection and isolation of a steam generator using neuro-fuzzy networks," *Neurocomputing Journal*, vol. 72(13-15), pp. 2939–2951, 2009.

[5] P. Baraldi, R. Razavi-Far, and E. Zio, "Classifier-ensemble incremental-learning procedure for nuclear transient identification at different operational conditions," *Reliab. Eng. Syst. Safety*, vol. 96, pp. 480–488, 2011.

[6] P. Baraldi, R. Razavi-Far, and E. Zio, "A method for estimating the confidence in the identification of nuclear transients by a bagged ensemble of FCM classifiers," in *Proc. NPIC&HMIT*, Las Vegas, NV, Nov. 7-11 2010, pp. 283–293.

[7] R. Razavi-Far, P. Baraldi, and E. Zio, "Ensemble of neural networks for detection and classification of faults in nuclear power systems," in *Proc. 10th International FLINS Conf. on Uncertainty Modeling in Knowledge Engineering and Decision Making, World Scientific Proc. Ser. on Computer Engineering and Information Science*, vol. 7, Istanbul, Turkey, 2012, pp. 1202–1207.

[8] R. Razavi-Far, P. Baraldi, and E. Zio, "Dynamic weighting ensembles for incremental learning and diagnosing new concept class faults in nuclear power systems," *IEEE Trans. Nucl. Sci.*, vol. 59, pp. 2520–2530, 2012.

[9] M. Muhlbaier, A. Topalis, and R. Polikar, "Learn++NC: combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes," *IEEE Trans. Neur. Net.*, vol. 20, pp. 152–168, 2009.

[10] R. Razavi-Far and M. Kinnaert, "Incremental design of a decision system for residual evaluation: A wind turbine application," in *8th IFAC Conf. Fault Detection, Supervision and Safety of Technical Processes*, vol. 8(1), 2012, pp. 343–348.

[11] R. Razavi-Far and M. Kinnaert, "A multiple observers and dynamic weighting ensembles scheme for diagnosing new class faults in wind turbines," *Control Engineering Practice*, vol. 21(9), pp. 1165–1177, 2013.

[12] R. Razavi-Far, E. Zio, and V. Palade, "Efficient residuals pre-processing for diagnosing multi-class faults in a doubly fed induction generator, under missing data scenarios," Expert Systems with Applications, in press, 2014.

[13] A. R. Mehrabian and C. Lucas, "A novel numerical optimization algorithm inspired from weed colonization," *Ecological Informatics*, vol. 1, pp. 355–366, 2006.

[14] D. A. G. Vieira, R. H. C. Takahashi, V. Palade, J. Vasconcelos, and W. Caminhas, "The Q-norm complexity measure and the minimum gradient method: A novel approach to the machine learning structural risk minimization problem," *IEEE Trans. Neur. Net.*, vol. 19, pp. 1415–1430, 2008.

[15] M. Markou and S. Singh, "Novelty detection: A review-part 1: Statistical approaches," *Signal Processing*, vol. 83(12), pp. 2481–2497, 2003.

[16] M. Markou and S. Singh, "Novelty detection: A review-part 2: Neural network based approaches," *Signal Processing*, vol. 83(12), pp. 2499–2521, 2003.

[17] A. R. Mehrabian and A. Yousefi-Koma, "Optimal positioning of piezoelectric actuators on a smart fin using bio-inspired algorithms," *Aerospace Science and Technology*, vol. 11, pp. 174–182, 2007.

[18] T. Green, R. Razavi-Far, R. Izadi-Zamanazabdi, and H. Niemann, "Plant-wide performance optimization, the refrigeration system case," in *IEEE Multi-Conf. on Systems and Control (MSC)*, Dubrovnik, Croatia, 2012, pp. 208–213.

[19] R. Razavi-Far, V. Palade, and J. Sun, "Optimizing the performance of a refrigeration system using an invasive weed optimization algorithm," in *Combinations of Intelligent Methods and Applications*, ser. Smart Innovation, Systems and Technologies. Springer Berlin Heidelberg, 2013, vol. 23, pp. 79–93.

[20] T. Green, R. Izadi-Zamanabadi, R. Razavi-Far, and H. Niemann, "Plant-wide dynamic and static optimisation of supermarket refrigeration systems," *Int. J. of Refrigeration*, vol. 38, pp. 106–117, 2014.

[21] R. Razavi-Far, V. Palade, and E. Zio, "Invasive weed classification," Neural Computing and Applications, in press, 2014.

[22] S. Karimkashi and A. A. Kishk, "Invasive weed optimization and its features in electromagnetics," *IEEE Trans. Ante. Pro.*, vol. 58, no. 4, pp. 1269–1278, 2010.