LOCAL FUSION OF AN ENSEMBLE OF SEMI-SUPERVISED SELF ORGANIZING MAPS FOR POST-PROCESSING ACCIDENTAL SCENARIOS

Francesco Di Maio[1], Roberta Rossetti[1], Enrico Zio[1,2]

[1]*Energy Department, Politecnico di Milano, Via La Masa 34, 20156 Milano, Italy*
[2]*Chair on System Science and the Energy Challenge, Fondation Electricite' de France (EDF), CentraleSupélec, Université Paris-Saclay, Grande Voie des Vignes, 92290 Chatenay-Malabry, France*
francesco.dimaio@polimi.it

Integrated Deterministic and Probabilistic Safety Analysis (IDPSA) of dynamic systems is challenged by the need of implementing efficient methods for accidental scenarios generation (that are to be increased with respect to conventional PSA, due to the necessary consideration of failure events timing and sequencing along the scenarios) and for their post-processing for retrieving safety relevant information regarding the system behavior (that, in the context of IDPSA consists in the classification of the generated scenarios as safe, failed, Near Misses (NMs) and Prime Implicants (PIs)). The large amount of generated scenarios makes the computational cost for scenario post-processing enormous and the retrieved information difficult to interpret. To address this issue, in this paper we propose the use of an ensemble of Semi-Supervised Self Organizing Maps (SSSOM) whose outcomes are combined by a locally weighted aggregation: we resort to the Local Fusion (LF) principle for accounting the classification reliability of the different SSSOM classifiers, for the type of scenario to be classified. The strategy is applied for the post-processing of the accidental scenarios of a dynamic U-Tube Steam Generator (UTSG).

Keywords: Integrated Deterministic and Probabilistic Safety Analysis, scenarios post-processing, Self-Organizing Map, local fusion, ensemble of classifiers.

I. INTRODUCTION

The number of dynamic scenarios considered in an Integrated Deterministic and Probabilistic Safety Analysis (IDPSA) of dynamic systems increases with the number of failure events that can occur and the consideration of their timing and sequencing. This can make the computational cost for scenario post-processing enormous and the retrieved information difficult to interpret (Refs. 1, 2, 3, and 4). The main goal of post-processing is the classification of the dynamic scenarios generated as safe, failed, Near Misses (NM) and Prime Implicants (PIs) clusters. Safe scenarios are those that, even if including several components failures, keep the system working in safe conditions. Failed scenarios, instead, result from a combination of failure events that lead the system into a failed condition. Among failed scenarios, PIs are those scenarios containing events representing the minimal combinations of component failure necessary for system failure (Ref. 5) (i.e., the dynamic systems equivalent of Minimal Cut Sets (MCSs)). Among safe scenarios, NMs are dangerous sequences of events that lead the system to a quasi-fault state (Ref. 6).

Many methods have been proposed in literature for the classification task. A first step could be distinguishing failed scenarios from safe scenarios, for example by a Fuzzy-*c*-Means (FCM) classifier (Ref. 6), a Mean-Shift Methodology (MSM) (Ref. 7), or a decision tree (Ref. 8). Methods have been proposed for the identification also of PIs and Near Misses. For example, PIs identification has been performed with a differential evolution-based method (Ref. 9) or a visual interactive method (Ref. 10), where the number of components whose behavior is specified in the accident sequence, is selected as most important feature for the PIs identification: the accident sequences associated with the lowest literal cost are selected and stored as PIs (most reduced sequences, i.e., with least number of events, that cannot be covered by any other implicant, i.e., PI by definition. Regarding the identification of the Near Misses sequences, an unsupervised clustering problem based on an optimized wrapper algorithm and the K-means clustering algorithm has been proposed (Ref. 4). A comprehensive method for accidental scenarios classification can be provided by Self-Organizing Maps (Refs. 11, 12, 13, and 14), which have been widely used in various engineering and physical applications, including fault detection and diagnosis in complex systems (Refs. 15, and 16). SOMs capture non-linear relationships of high-dimensional data and visualize them on a low dimensional interface, normally a 2-D structure of, so called, neurons. In this structure, data are assigned to the most similar neuron called Best Matching Unit (BMU) (usually by measuring the smallest Euclidean distance), so that the available data are divided into regions with common characteristics (i.e., data with high similarity to the same BMU are mapped close to each other). Three kind of SOM exist: the Unsupervised SOM (USOM), the Semi-Supervised SOM (SSSOM)

and the Supervised SOM (SSOM). We have shown in (Ref. 17) a SSSOM performs best in identifying safe, failed, NMs and PIs groups of scenarios. In particular, assigning the set of discrete variables (i.e., the failure sequences) to a BMU, a SSSOM (implemented with a Manhattan distance as similarity measure) is particularly suitable to properly treat the Multi-Valued Logic (MVL) approximation needed the representation of the dynamic scenarios (the usual binary variables representation used in Boolean Logic, in which the modeling is limited only to the occurrence or not of certain events (Refs. 2, 3, 4, 6, 10, and 18) is not sufficient). In this work, it will be shown that the SSSOM performance in classifying different groups of scenarios depend on the feature of the SSSOM that is used as discriminating characteristics for choosing the BMU (for example, assigning the data to the cluster with the geometric barycenter more similar to the input data, or to the cluster with the maximum (minimum) neuron (i.e., with the maximum (minimum) weights) more similar to the input data). The results confirm that depending on this, some classifiers over-perform the stand-alone SSSOM for some classes and vice versa. This suggest to adopt an ensemble approach for an improved classification of accidental scenarios.

The main objective of this work is to propose a post-processing tool for dynamic accidental scenarios, that exploits an ensemble of classifiers. In fact, by doing so, it is possible to leverage the classifiers complementary characteristics and to boost overall classification accuracy (in terms of the multi-objective precision sensitivity and specificity) (Ref. 19). In general, strategies for boosting diversity include: i) using different types of classifiers (this is the technique we adopt for our application); ii) training individual classifiers with different data sets; iii) using different subsets of features. Various methodologies exist for aggregation the outcomes of individual classifiers: majority vote (Ref. 20), Borda count (Ref. 21), threshold voting, weighted average (Ref. 22), fuzzy integral (Ref. 23), fuzzy templates (Ref. 24) and Dempster-Shafer theory (Ref. 25). Furthermore, methods have been developed to dynamically select a classifier from the set of available ones, based on local information (Ref. 26): different classifiers perform best in different regions and this aggregation can lead to improving classification results; in a supervised setting, the individual classifier performance can be calibrated based on historical data with known target values; each individual classifier performance value reflects the degree to which we want each classifier to contribute in the ensemble aggregation: the best performing classifier for a given scenario type should contribute most (Ref. 19). On these premises, we a locally weighted aggregation of SSSOMs outcomes: we resort to the Local Fusion (LF) principle (Ref. 22) for building the ensemble outcome, based on each classifier local performance, measured by the classification accuracy on scenarios in the neighborhood of (i.e., similar to) the test scenario considered.

In practice, we ensemble the classification outcomes of the SSSOMs whose assignments to a BMU are given with respect to the different features characterizing the SSSOM (e.g. the Mean Quantization Error (MQE) based SSSOM, the barycenter based SSSOM, the minimum neuron based SSSOM, the maximum neuron based SSSOM and the stand-alone SSSOM).

The feasibility of combining local information for post-processing IDPSA scenarios for their classification into safe, failed, NMs and PIs, is demonstrated with respect to a dynamic U-Tube Steam Generator (UTSG) of a NPP (Ref. 27). For IDPSA scenarios generation, a dynamic simulation model has been implemented in SIMULINK and a Multi-Valued Logic (MVL) scheme (Ref. 4) has been adopted for describing the different component operational states in the scenarios.

The paper is organized as follows. In Section II, the UTSG and its SIMULINK model are presented. In Section III, the SSSOMs are presented and different features are considered as discriminating characteristics for the classification; also the LF process for ensembling is outlined. In Section IV, the locally weighted ensemble of SSSOMs is presented, and the results on the case study considered are reported. In Section V, some conclusions and final remarks are given.

II. CASE STUDY

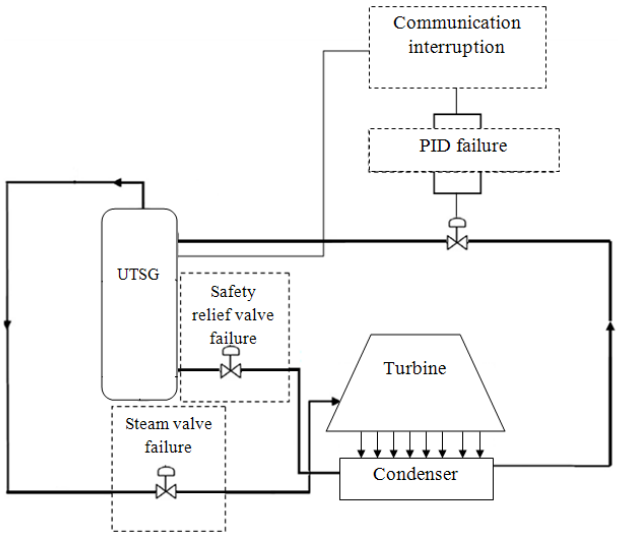

Fig. 1. Sketch of the failures that can be injected into the system.

A SIMULINK model has been used to describe the UTSG response at different power levels P_0 (Ref. 4). The component failures considered for UTSG are (Fig. 1): the steam valve failure, the safety relief valve failure, the interruption of the communication between the sensor that monitors the water level (governed by the balance between the incoming and exiting feed water) and the Proportional Integrative Derivative (PID) controller, and the PID failure. A mission time (T_{miss}) of 4000 (s) has been considered for allowing complete development also of slow dynamic accident scenarios occurring at early/medium times. The component failures are

considered occurring at any continuous time instant, with any order in the sequence and magnitude

Assumptions on the failure occurrence process have been made in order to i) favor the occurrence of multiple failures in the scenarios, ii) capture the dynamic influence of all factors of interest and iii) treat a comprehensive (but still manageable) problem for which scenarios post-processing is required for a robust risk quantification.

For the tractability of the problem, we resort to a Multi Logic Value (MVL) computational framework in which the components can fail at discrete times and magnitudes (Ref. 9). The discretization consists in:

- Time: for each component, the mission time (T_{miss}) is divided into four intervals, labelled t=1 for a failure in [0, 1000] (s), t=2 in [1000, 2000] (s), t=3 in [2000, 3000] and t=4 in [3000,4000]. If t=0 the component does not fail in T_{miss}.
- Component failure magnitudes:
  - the steam valve failure magnitude is indicated as 1, 2 or 3 for failure states corresponding to stuck at 0%, at 50% and at 150% of the Q_e value that should be provided at power level P_o, respectively; if the steam valve magnitude is indicated as 0, the component does not fail in T_{miss};
  - the safety relief valve failure magnitude is indicated as 1, 2, 3 and 4, if it is stuck between [0.5, 12.6] (kg/s), (12.6, 25.27] (kg/s), (25.27, 37.91] (kg/s) and (37.91, 50.5] (kg/s), respectively; if the safety relief valve magnitude is indicated as 0, the component does not fail in T_{miss};
  - the communication between the sensor measuring the water level and the PID controller is labeled with 0 if the communication works, with 1 otherwise;
  - the PID controller failure magnitude is discretized into 8 equally spaced magnitude intervals, labeled from 1 to 8, representative of failure states corresponding to discrete intervals of output value belonging to [-18,18]% of the Q_e value that should be provided at P_o; if the PID controller magnitude is labeled as 0, the component does not fail in T_{miss}.

All possible combinations of multiple component failures, each represented by the values of the multi-valued variables of time, magnitude and order of occurrence, lead to a total of N=100509 accidental scenarios to be treated for the quantification of the risk related to the UTSG operation. For each scenario, these variables are resumed into a sequence vector: each sequence is, thus, an MVL vector $\bar{X} = [x_1, x_2, \dots, x_d]$, of length d=12. For example [2,3,1,3,1,3,2,1,2,4,6,4] corresponds to a scenario where:

- the steam valve fails stuck at its maximum allowable value (3) at a time (2) in [1001, 2000] (s) and it is the first (1) event occurring along the sequence;
- the safety relief valve fails third (3) in the time interval (3) equal to [2001, 3000] (s), with a magnitude (1) belonging to [0.5, 12.6] (kg/s);
- the communication between the sensor measuring the water level and the PID controller is the second (3) failure event (1) in the sequence, and it occurs in the time interval (2) of [1001, 2000] [1001, 2000] (s);
- the PID controller fails stuck as fourth (4) in the time interval (4) of [3001, 4000] (s), with a magnitude (6) belonging to [6, 10] % of the Q_e value that should be provided at P_o.

III. THE ENSEMBLE

The design of a successful ensemble consists of two important parts (Refs. 23, and 28): 1) the design of the individual classifiers (Section III.A); 2) the design of the aggregation mechanism (Section III.B) (Ref. 29).

III.A. Design of Classifiers

For post-processing the N=100509 multi-valued dynamic scenarios of the UTSG, we resort to a Semi-Supervised Self-Organizing Map (SSSOM) based on the Manhattan distance (shown in Fig. 2, center). This SSSOM has been shown in Ref. 17 to be efficient for grouping the scenarios in four distinct regions of the map and retrieving safety-relevant information, and it is hereafter shown to be capable of further improvement when trained to classify new data based on different features of this same SSSOM to be used as BMUs and, then, their outcomes are ensembled into the final classification: we shall see, certain classifiers over-perform the others for certain classes and vice versa. Specifically, we build K=5 classifiers the stand-alone SSSOM, the MQE based SSSOM, the Barycenter based SSSOM, the Minimum neuron based SSSOM, the Maximum neuron based SSSOM, and show how for different classes, none of these is the best and all would mutually benefit from each others, namely.

III.A.1. The Stand-Alone SSSOM

A SSSOM of M=3025 neurons $C=[c_1, c_2, \dots, c_M]$, each of which is assigned a weight vector $\bar{w}_m = [w_1, w_2, \dots, w_d]$ is trained on the N=100509 UTSG dynamic scenarios $\bar{X}$ belonging to a d=12-dimensional space, say $\bar{\bar{X}} = [\bar{X}_1, \bar{X}_2, \dots \bar{X}_N]$, where the n-th sample is $\bar{X}_n = [x_1, x_2, \dots x_d]$. In particular, this SSSOM is constructed by replacing the Euclidean distance as similarity measure between the generic scenario in input $\bar{X}_n$ and the weight $\bar{w}_m$ of the M neurons of the map, with the Manhattan distance:

$$d_{Manhattan}(\bar{X}_n, \bar{w}_m) = \sum_{k=1}^{d} \|\bar{X}_k - \bar{w}_k\| \quad (1)$$

where $\|\cdot\|$ is the absolute value of the difference between the two vectors along the d-dimension (Ref. 15). By doing so, the MVL formalism is accommodated within the similarity assessment between data vector and

neurons. The map of Fig. 2 (center) has been built with t_{tot}=15 training epochs and a $\varepsilon(t=0)=0.01$ factor. Different shades of color represent the different G=4 classes t=[1,2,3,4] for safe, NMs, failed and PIs, respectively.

III.A.2. The MQE Based SSSOM

Let us consider a generic scenario $\bar{X}_n$ and a generic neuron of the map $\bar{w}_i$. A commonly used quality measure that can be used to determine the performance of the map is the Mean Quantization Error (MQE) and it can be defined as in Eq. 2:

$$MQE = \frac{1}{N}\sum_{n=1}^{N}\|\bar{X}_n - \bar{w}_i\| \qquad (2)$$

where $\bar{w}_i$ are the weights associated to the BMU neuron c_i.

Basically, the lower the MQE of the BMU, the more the scenario features vector is similar to its weight vector and, thus, the more the knowledge is learnt by the SSSOM. Computing the MQE for each input data and grouping them classwise, we can obtain the empirical probability density functions (PDF) referring to the distribution of the MQE for each class (Fig. 2 top right, left and bottom right left).

Eq. 3 shows an example of computation of the MQE for a generic class g:

$$MQE_g = \frac{1}{N_g}\sum_{n=1}^{N_g}\left\|\bar{X}_{n_g} - \bar{w}_i\right\| \qquad (3)$$

where N_g is the number of scenarios belonging to the class g, $\bar{X}_{n_g}$ is a generic scenario belonging to the class g and $\bar{w}_i$ is the weight vector of the BMU neuron in the map to which $\bar{X}_{n_g}$ is assigned. The classification of a new input to a g class with the MQE-based SSSOM proceeds as follows: its MQE_g is calculated as in Eq. 2 and, then, it is assigned to the class with the larger PDF value for the calculated MQE. The rationale is that, for a particular value of MQE, the larger the PDF, the more probable is the value: if for a class g, the PDF associated to a MQE value is larger than for the other classes, it is more probable that the scenario belongs to that class. For example, if the MQE of an input is equal to 1.5, we assign it to the safe class because the PDF of the safe class (Fig. 2 bottom right) is larger than the other classes, for this value of MQE. In general, we can notice in Fig. 2 that NMs and PIs classes have PDF skewed towards low values of MQE, whereas failed and safe classes have larger MQE values.

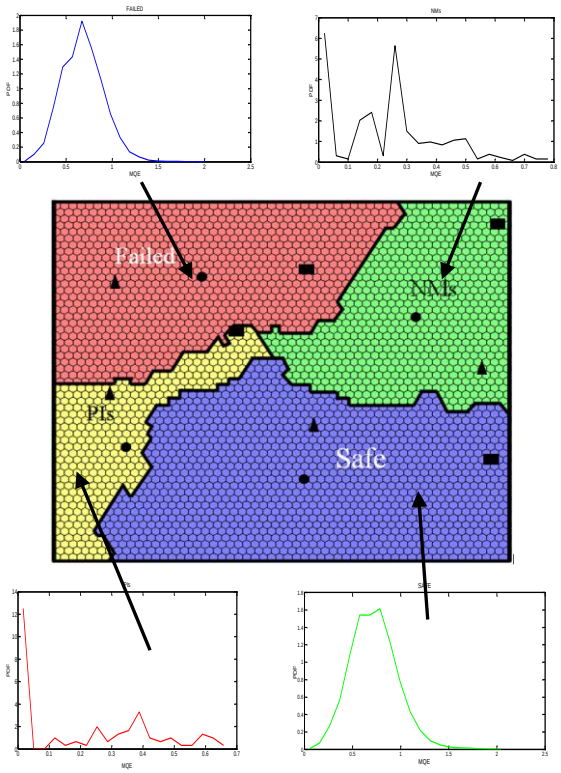

Fig. 2. The stand-alone SSSOM (center):different shades of color indicate different classes, circles are the geometric barycenters of the classes, triangle are the minimum neurons of the classes and rectangles are the maximum neurons of the classes. Top left: PDF of the MQE for failed scenarios; top right: PDF of the MQE for NMs scenarios; bottom left: PDF of the MQE for PIs scenarios; bottom right: PDF of the MQE for safe scenarios.

III.A.3. The Barycenter Based SSSOM

The same SSSOM shown in Fig. 2 (center) is exploited as an alternative classifier by using the geometric barycenter of each cluster as a reference for the choice of the BMU (circles in Fig. 2). When a new $\bar{X}_n$ is fed to this SSSOM, we select the closest of the four barycenter neurons as the most similar neuron, where similarity is quantified based on the Manhattan distance:

$$d_{Manhattan}(\bar{X}_n, \bar{w}_{g_{bar}}) = \sum_{k=1}^{d}\left\|\bar{X}_k - \bar{w}_{g_{bar}}\right\| \qquad (4)$$

where $\bar{w}_{g_{bar}}$ is anyone of the four barycenters of the four classes. The rationale is that the geometric barycenter is most representative of the characteristics of the class.

III.A.4. The Minimum Neuron Based SSSO1M

Considering again the SSSOM shown in Fig. 2 (center) for each cluster g we locate on the map the neuron with the minimum weight $\bar{w}_{g_{min}}$ (represented in the map with a triangles in the map of Fig. 2) and for the

classification we assign the new vector $\bar{X}_n$ to the cluster with the minimum neuron most similar to the considered input, based on the Manhattan distance:

$$d_{Manhattan}(\bar{X}_n, \bar{w}_{g_{\min}}) = \sum_{k=1}^{d} \left\| \bar{X}_k - \bar{w}_{g_{\min}} \right\| \quad (5)$$

The rationale is that if the vector of the features of a scenario is similar to that of neuron with the minimum weight of a specific cluster, it will be assigned to this cluster because very different to the neurons with minimum weight vectors of the other classes.

III.A.5. The Maximum Neuron Based SSSOM

The maximum neuron based SSSOM is complementary to the previous one, in that it is based on the neuron with the maximum weights for each cluster $\bar{w}_{g_{\max}}$ represented by a rectangle in the map of Fig. 2 (center).

III.A.6. Classification Performance

The four classifiers of Sections III.A.2-III.A.5 are compared to the stand-alone SSSOM of (Ref. 17), on the UTSG scenario post-processing task. The performances of the classifiers are quantified by the calculation of (Ref. 29):

- Precision: the larger, the better the capability of the *k*-th classifier to not include samples of other classes in the considered *g*-th class (Eq. 6):

$$Pr_g = \frac{n_{gg}}{n'_g} \quad (6)$$

  where n'_g is the total number of scenarios assigned to the *g*-th class and n_{gg} is the number of scenarios belonging to class *g* and correctly assigned to class *g*;
- Sensitivity: the larger, the better the capability of the *k*-th classifier to correctly recognize samples belonging to the *g*-th class (Eq. 7):

$$Sn_g = \frac{n_{gg}}{n_g} \quad (7)$$

  where n_g is the total number of scenarios belonging to the *g*-th class.
- Specificity: the larger, the better the capability of each *g*-th class of the *k*-th classifier to reject the samples of all the others (Eq. 8):

$$Sp_g = \frac{\sum_{k=1}^{G}(n'_k - n_{gk})}{N - n_g} \qquad for\ k \neq g \quad (8)$$

  where n'_k is the total number of samples assigned to the *k*-th class:

$$n'_k = \sum_{g=1}^{G} n_{gk} \quad (9)$$

In TABLEs I-V, the performances for the MQE based, the barycenter based, the minimum neuron based, the maximum neuron based and the stand-alone SSSOMs, for each class, are reported.

TABLE I. MQE Based SSSOM Performances: Precision, Sensitivity and Specificity for Each Class

MQE based	Safe	Failed	NMs	PIs
Precision	0.674	0.3803	0.0406	0.0083
Sensitivity	0.4699	0.475	0.491	0.6333
Specificity	0.6006	0.5674	0.9616	0.9326

.

TABLE II. Barycenter Based SSSOM Performances: Precision, Sensitivity and Specificity for Each Class.

Barycenter based	Safe	Failed	NMs	PIs
Precision	0.6459	0.4816	0.0097	0.0014
Sensitivity	0.2975	0.3859	0.5934	0.3444
Specificity	0.7134	0.7678	0.7996	0.7827

TABLE III. Minimum Neuron Based SSSOM Performances: Precision, Sensitivity and Specificity for Each Class.

Minimum neuron based	Safe	Failed	NMs	PIs
Precision	0.7437	0.5418	0.0325	0.0025
Sensitivity	0.7145	0.427	0.0994	0.2667
Specificity	0.5666	0.7981	0.9902	0.9054

TABLE IV. Maximum Neuron Based SSSOM Performances: Precision, Sensitivity and Specificity for Each Class.

Maximum neuron based	Safe	Failed	NMs	PIs
Precision	0.833	0.3864	0.0066	0.002
Sensitivity	0.1278	0.3514	0.5813	0.6222
Specificity	0.955	0.6881	0.7084	0.7167

TABLE V. Stand-Alone SSSOM Performances: Precision, Sensitivity and Specificity for Each Class.

SSSOM (Ref. 17)	Safe	Failed	NMs	PIs
Precision	0.949	0.83	0.034	0.016
Sensitivity	0.78	0.773	0.957	0.911
Specificity	0.927	0.911	0.911	0.949

For failed and PIs scenarios the parameters of the stand-alone SSSOM (precision 0.83 and 0.016, sensitivity 0.773 and 0.911, and specificity 0.911 and 0.949, respectively) are larger than for all the other classifiers. For example, for the minimum neuron based SSSOM, the precision for failed scenarios is equal to

0.5418, which is much lower than the precision obtained with the stand-alone SSSOM. It is worth noticing that this is always true for all the parameters values when dealing with failed and PIs scenarios. On the contrary, looking at on the NMs and safe scenarios, we see that the other classifiers overcome the stand-alone SSSOM performances. For example, the specificity in classifying safe scenarios is higher for the maximum neuron based SSSOM than for the stand-alone SSSOM (0.955 vs. 0.927), and the precision in classifying NMs is higher for the MQE based SSSOM than for the stand-alone SSSOM (0.0406 vs. 0.034)), and also the specificity in classifying NMs, for both the MQE based SSSSOM (0.9616) and the minimum neuron based SSSOM (0.9902) is higher than the stand-alone SSSOM (0.911).

In Fig. 3, a 3-D representation of the performance parameters values of TABLEs I-V is given for each implemented SSSOM and each scenario class: stars indicate the MQE based SSSOM values, circles the barycenter based SSSOM values, squares the minimum neuron based SSSOM values, diamonds the maximum neuron based SSSOM and crosses the stand-alone SSSOM values.

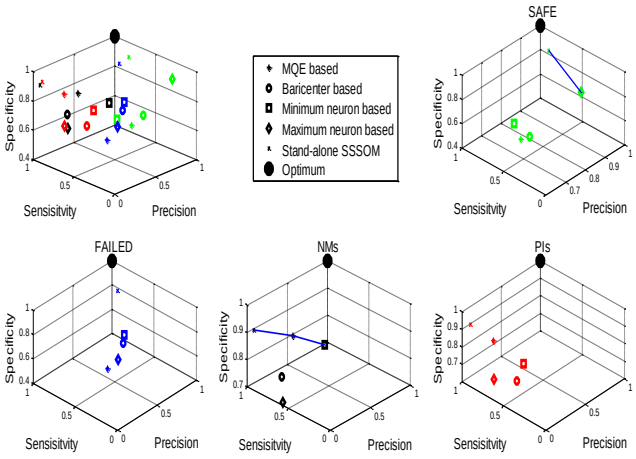

Fig. 3. 3-D representation of the performance parameters.

Fig. 3 (upper left) confirms that, the stand-alone SSSOM, on average, over performs the other classifiers, except for safe and NMs classes: for these scenarios in Fig. 3 (top right) and Fig. 3 (bottom center) respectively, a Pareto front can be identified and highlighted with a solid line for the sub-optimal solutions of classifiers that do not dominate all the others with respect to all the three performance objectives. For example, we can see that the precision in classifying the NMs is higher for the MQE based SSSOM (0.0406) than for the stand-alone SSSOM (0.034) and so is the specificity in classifying NMs (0.9616 vs. 0.911), but the sensitivity for the same class is higher for the stand-alone SSSOM (0.957 vs. 0.491) These results suggest the possibility of developing a method for aggregating the multiple classifiers outputs considered as an ensemble.

III.B. Design of the Aggregation Mechanism: the Locally Weighted Fusion

Let w_Q^k be the weight that classifier k carries in assigning scenario $\bar{X}_n$ to a class of a dataset of N scenarios to be classified:

$$w_{\bar{X}_n}^k = \frac{1}{me_{\bar{X}_n}^k} \tag{10}$$

where the Mean Error (ME) $me_{\bar{X}_n}^k$ is the error that classifier k makes in classifying the scenario $\bar{X}_n$, defined as:

$$me_{\bar{X}_n}^k = \frac{\sum_{n=1}^{N} e_n^k}{N} \tag{11}$$

and e_n^k is the error that the classifier k commits in classifying the n-th scenario whose real class is t_n (with n=1… N). In this work, the error e_n^k is computed in two different ways: being $\hat{y}_n^k$ =1,…,G the class the k-th classifier assigns to $\bar{X}_n$, the first way for computing $e_{n\,(1)}^k$ is given in Eq. 12:

$$e_{n\,(1)}^k = \begin{cases} f(x) = \begin{cases} 0, & if\ \hat{y}_n^k = t_n \\ 1, & if\ \hat{y}_n^k \neq t_n \end{cases} \end{cases} \tag{12}$$

where the error is null if the estimated class $\hat{y}_n^k$ is the same as the real class t_n (where $\hat{y}_n^k = 1$ and t_n =1 means that the estimated and real class of the scenario are safe, respectively, $\hat{y}_n^k = t_n$=2 means failed, $\hat{y}_n^k = t_n$=3 means NMs and $\hat{y}_n^k = t_n$=4 means PIs), whereas in the second way $e_{n\,(2)}^k$is calculated as the Manhattan distance between the real and the predicted class by Eq. 13:

$$e_{n\,(2)}^k = \left\| \hat{y}_n^k - t_n \right\| \tag{13}$$

Usually the error is computed by relying on a subset of N, called neighbor set of scenarios to $\bar{X}_n$ and defined as in Eq. 10:

$$P(\bar{X}_n) = \{u_j |\, u_j \epsilon\, N(\bar{X}_n)\} \tag{14}$$

where $u_j = \langle x_{1,j}, x_{2,j}, \ldots x_{d,j} \rangle$ is a set of d-dimensional scenarios, $N(\bar{X}_n)$ is the neighborhood of $\bar{X}_n$ that is in this work defined as a set of $N_{\bar{X}_n}$=100 scenarios (i.e. a subset of N scenarios) whose Manhattan distance for the instance to be classified is lower than 10 (i.e., being d=12, a threshold value equal to 10 means $\bar{X}_n$ and its neighboors have not to differ too much) and, thus, j=1,…100.

$$(dist_{Manhattan}(\bar{X}_n, \mathrm{u_j}) = \sum_{l=1}^{d} \left\| \bar{X}_{n_l} - \mathrm{u_l} \right\| \tag{15}$$

In this way, the k-th classifier performance is expected to be similar to the one that would be obtained with a new (unknown) scenario. A weight $w_{\bar{X}_n}^k$ can, thus,

be associated to each individual *k* classifiers of the ensemble depending on its performance, as it will be shown in the next Section.

IV. THE PROPOSED ENSEMBLE STRATEGY

In the following, we describe the details of the implemented ensemble strategy, namely the locally weighted ensemble of SSSOMs. This approach relies on the five classifiers introduced in Sections III.A.1- III.A.5 (the stand-alone SSSOM, the MQE based SSSOM, the barycenter based SSSOM, the minimum neuron based SSSOM and the maximum neuron based SSSOM, respectively), whose classification outcomes are combined.

We directly apply the algorithm of the neighborhood based approach, as described in Section III.B to the *N*=100509 dynamic scenarios. For each scenario to be classified we retrieve the 100 neighbors based on the considerations made before: relying on the neighborhood of each scenario we compute the classification errors (both with Eq. 12 and Eq. 13) and, through the errors, also the weights associated. The classification outcomes of the five different trained SSSOMs are ensembled and the assignment to a class is given accounting for the different performances of these classifiers when assigning the weight (the larger the number of neighbors of the input scenario correctly classified, the lower the error, the larger the weight and the reliability for the *k*-th classifier). For the computation of the weight associated to each classifier *k* for each scenario, thus, we resort Eq. 10 and Eq. 11 where for the *n-th* generic scenario and the *k-th* classifier, we calculate:

$$w_{\bar{X}_n}^k = \frac{1}{me_{\bar{X}_n}^k} \tag{16}$$

where $\bar{X}_n$ is one of the *N*=100509 dynamic scenarios, $w_{\bar{X}_n}^k$ is the weight associated to this scenario and $me_{\bar{X}_n}^k$ is the ME associated to this scenario and computed as in Eq. 13:

$$me_{\bar{X}_n}^k = \frac{\sum_{j=1}^{N_{\bar{X}_n}} e_j^k}{N_{\bar{X}_n}} \tag{17}$$

where $N_{\bar{X}_n}$=100 and e_j^k is the classification error. Once the weights are computed for all the *K*=5 classifier, the input data $\bar{X}_n$ is assigned to the class with the larger weight $k=arg((\max_k(w_{\bar{X}_n}^k)))$, because this is the most reliable classifier for the *n*-th vector.

IV.A. Training of the Locally Weighted Ensemble of SSSOMs

TABLE VI shows the classification results for all scenarios and for those of NMs and PIs classes. We focus on these latter two classes because these are the two more relevant for quantifying the operational risk of the system. The rows of the table report the results obtained when Eq. 12 and Eq. 13 are used for computing the classification error. We can see that, in both cases, the total number of correctly assigned scenarios exploiting the locally weighted ensemble of SSSOMs increases with respect to the stand-alone SSSOM (whose results are reported in TABLE VII): this latter, in fact, scores a total amount of 78288 rightly assigned scenarios (Ref. 17), while with the locally weighted ensemble of SSSOMs, we obtain 84141 when the error is given by Eq. 12 and 81512 when we resort to the Manhattan distance of Eq. 13 for computing the error. However both ensembles are penalized with respect to NMs and PIs classification (second and third column): the stand-alone SSSOM correctly assigns 318 out of 332 NMs and 82 out of 90 PIs (as reported in TABLE VII). It is worth pointing out that even if the ensembles do not correctly classify all NMs and PIs scenarios, we can consider these results satisfactory for the operational risk quantification which the classification is aiming at contributing to, that refers to the consequences of the scenario occurring and to its probability of occurrence: as already said, PIs normally are made of many component failures but because of this, also have low probability of occurrence and, thus, the risk that is not accounted for due to the misclassification of PI is very low; whereas for the NMs, those scenarios that are not correctly classified are either classified as safe (with no extra risk quantification being both safe and NMs leading to safe states) or failed scenarios (with a conservative overestimation of the system operational risk).

TABLE VI. Locally Weighted Ensemble Classification Results: Method Using Eq. 12 and Eq. 13

Approach \ Correctly assigned	Total	NMs	PIs
Locally weighted ensemble by using Eq. 12	84141	104	66
Locally weighted ensemble by using Eq. 13	81512	308	77

TABLE VII. Stand-Alone SSSOM Classification Results.

Approach \ Correctly assigned	Total	NMs	PIs
Stand-alone SSSOM	78288	318	82

TABLE VIII. reports the same results in terms of percentage of correct assignment.

Looking at the two locally weighted ensembles, we can say that the one based on the Manhattan distance is more effective in the assignment of NMs and PIs than the other: we see in fact that the percentage of correctly

assigned NMs is 31.33% when Eq. 12 is used and 92.77% when Eq. 13 is used, whereas for PIs, the percentage increases from 73.33% to 85.56% when the Manhattan distance is used. Even if using Eq. 12, the percentage of correct assignment is larger than when the Manhattan distance is used (83.71% vs. 81.1%), since NMs and PIs are the most safety relevant classes and, thus, are those we have to guarantee to be better classified during the post-processing of dynamic scenarios.

TABLE VIII. Locally Weighted Ensemble Percentage Classification Result Using Eq. 12 and Eq. 13

Correctly assigned / Approach	Total	NMs	PIs
Locally weighted ensemble by using Eq. 12	83.71%	31.33%	73.33%
Locally weighted ensemble by using Eq. 13	81.1%	92.77%	85.56%

TABLE IX. Locally Weighted Ensemble Performance Parameters Using Eq. 12

Ensemble (1)	Safe	Failed	NMs	PIs
Precision	0.9443	0.8162	0.0338	0.0186
Sensitivity	0.8453	0.8277	0.3133	0.7333
Specificity	0.9124	0.8958	0.9704	0.9653

TABLE X. Locally Weighted Ensemble Performance Parameters Using Eq. 13

Ensemble (2)	Safe	Failed	NMs	PIs
Precision	0.9488	0.8359	0.0437	0.0195
Sensitivity	0.8266	0.782	0.9277	0.8556
Specificity	0.9216	0.9142	0.9327	0.9615

Furthermore, TABLEs IX and X list the precision, sensitivity and specificity values for the two ensembles for all the four classes. The best performances are obtained by using Eq. 13: the precision is larger using Eq. 13 than Eq. 12 for all the four classes and what we gain in terms of sensitivity in classifying NMs and PIs scenarios and specificity in classifying safe and failed scenarios, justifies a negligible loss in terms of sensitivity in classifying safe and failed scenarios, and specificity in classifying NMs and PIs scenarios. In fact:

- the specificity for NMs and PIs decreases (from 0.9704 to 0.9327 and from 0.9653 to 0.9615, respectively);
- the sensitivity for safe and failed scenarios decreases (from 0.8453 to 0.8266 and from 0.8277 to 0.782, respectively).

Using Eq. 13, we gain a consistent benefit, viz:

- the sensitivity for NMs and PIs increases (from 0.3133 to 0.9277 and from 0.7333 to 0.8556, respectively);
- the specificity for safe and failed scenarios incerases (from 0.9124 to 0.9216 and from 0.8958 to 0.9142, respectively).

In conclusion, it is possible to assert that the approach based on Eq. 13 leads to superior results of classification.

IV.B. Test of the Locally Weighted Ensemble of SSSOMs

We test the locally weighted ensemble of SSSOMs approach with a set of scenarios in which the time is not discretized anymore, but it is continuous. A new set of input data $\bar{\bar{X}}_{test}$ of 2000 scenarios have been generated, in which components can fail randomly between 0 and the mission time of 4000 (s). Then, the trained classifiers are used to classify $\bar{\bar{X}}_{test}$. In TABLE XI, the results of the test conducted on the locally weighted ensemble of SSSOMs are reported.

TABLE XI. Ensemble Classification Results.

Correctly assigned / Approach	Total	NMs	PIs
Locally weighted ensemble by using Eq. 12	1673	2	9
Locally weighted ensemble by using Eq. 13	1599	8	11

Within the set of the 2000 input data, there are 8 NMs and 11 PIs. We can see in TABLE XI that the test classification results confirm the considerations made for the training. The ensemble based on the Manhattan distance is more efficient in the assignment of NMs and PIs than the other, even if the total correct assignment is larger for Eq. 12 than for Eq. 13: using Eq. 9 all the NMs and PIs scenarios are correctly classified, whereas only 2 NMs and 9 PIs are assigned to the right class, if the Eq. 12 is used.

V. CONCLUSIONS

The post-processing of IDPSA accidental scenarios of a dynamic system is a fundamental task for retrieving safety-relevant information for the system operation and maintenance. In practice the task can be challenged by the combinatorial explosion of the scenarios generated due to the dynamic dependences of components failure events and the consideration of timing and magnitudes of failure events in the accidental scenarios generation.

In this paper, for UTSG scenario generation a SIMULINK dynamic simulation model has been used, within a MVL scheme that describes the different component operational states, and have presented a locally weighed ensemble of SSSOMs for scenario classification. In general terms, the results obtained are satisfactory: the proposed SSSOM-based classification methods carry also fundamental insights on the risk

characterization of the UTSG operation. The methodology highlights the need of taking into account different classifiers to recover information that would have been lost if neglected. Furthermore, the dynamic scenarios that are misclassified do not cause a negative contribution to system operational risk quantification: the risk associated to a certain accidental scenario refers to the consequences of the scenario itself and to its probability of occurrence. The most probable scenarios are those with few or none accidental event (i.e., component failures) and, thus belong to safe or NMs classes, whereas the scenarios with many component failures are less probable and, thus belong to failed and PIs scenarios. Thus, when a probable scenario is misclassified, the system does not incur in a critical danger because the scenario itself is safe, while if a failed scenario is not assigned to the right class, the probability of occurrence is actually very low.

REFERENCES

1. P.E. LABEAU, C. SMIDTS, AND S. SWAMINATHAN, Dynamic reliability: towards an integrated platform for probabilistic risk assessment. *Reliab. Eng. Syst. Safe.*, 68(3), p.219-254 (2000).
2. T. ALDEMIR, A survey of dynamic methodologies for probabilistic safety assessment of nuclear power plants. *Ann. Nucl. Energy*, 52, p.113-124 (2013).
3. E. ZIO, Integrated deterministic and probabilistic safety assessment: concepts, challenges, research directions. *Nucl. Eng. Des.*, 280, pp.413-419 (2014).
4. F. DI MAIO, M. VAGNOLI and E. ZIO, Risk-based clustering for near misses identification in integrated deterministic and probabilistic safety analysis. *Sci. Tech. Nucl. Install.*, 2015.
5. W. V. QUINE, The problem of simplifying truth functions, *Am. Math. Monthly*, 59, p. 521-531 (1952).
6. E. ZIO and F. DI MAIO, Processing Dynamic Scenarios from a Reliability Analysis of a Nuclear Power Plant Digital Instrumentation and Control System, *Ann. Nucl. Energy*, 36, p.1386-1399 (2009).
7. D. MANDELLI et al., Scenario clustering and dynamic probabilistic risk assessment, *Reliab. Eng. Syst. Safe.*, 115, p. 146-160 (2013).
8. S. GALUSHIN and P. KUDINOV, An approach to grouping and classification of scenarios in Integrated Deterministic-probabilistic Safety Analisys, *Probabilistic Safety Assessment and Management (PSAM) 12*, Honolulu, Hawaii (June 2014).
9. F. DI MAIO, S. BARONCHELLI and E. ZIO, A Computational framework for Prime Implicants Identification in non-coherent Dynamic Systems, *Risk Anal.*, 1, p. 142-156 (2014).
10. F. DI MAIO, S. BARONCHELLI and E. ZIO, A visual interactive method for prime implicants identification, *IEEE Trans. Reliab.*, 99, p. 1-11 (2014).
11. T. KOHONEN, The self-organizing map, *Proc. IEEE*, 78(9), p. 1464-1480 (1990).
12. KOHONEN, The self-organizing map, *Neurocomputing*, 21, pp. 1-6, 1998.
13. D. BALLABIO and M. VASIGHI, A MATLAB toolbox for Self-Organizing Maps and supervised neural network learning strategies, *Chemometr. Intell. Lab.*, 118, p. 24-32 (2012).
14. J. VESANTO et al., *SOM Toolbox for Matlab 5*, Helsinhi University of Technology , Report A57..
15. S. WU and T. CHOW, Induction Machine Fault Detection Using SOM-Based RBF Neural Networks, *IEEE Trans. Ind. Electron.*, 51(1), p. 183-194 (2004).
16. H. YU, F. KHAN and V. GARANIYA, Risk-based fault detection using Self-Organizing Maps, *Reliab. Eng. Syst. Safe.*, 139, p. 82-96 (2015).
17. F. DI MAIO, R. ROSSETTI and E. ZIO, A Semi-Supervised Self-Organizing Map for post-processing the scenarios of an Integrated Deterministic and Probabilistic Safety Analysis (2015).
18. F. DI MAIO, S. BARONCHELLI and E. ZIO, Hierarchical Differential Evolution for Minimal Cut Sets Identification: Application to Nuclear Safety Systems, *Eur. J. Oper. Res.*, 238(2), p. 645-652 (2014).
19. P.P. BONISSONE, F. XUE and R. SUBBU, Fast meta-models for local fusion of multiple predictive models, *Appl. Soft Comput*., 11, p. 1529-1539 (2008).
20. L. LAM and C. Y. SUEN, A theoretical analysis of the application of the majority voting to pattern recognition, *Proc. the 12th International Conference on Pattern Recognition and Computer Vision, Jerusalem*, p. 418-420 (1994).
21. L. XU, A. KRZYZAK and C. Y. SUEN, Methods of combining multiple classifier and their applications to handwriting recognition, *IEEE T. Syst. Man Cy. B.*, 22(3), p. 418-435 (1992).
22. P. BARALDI et al., Local fusion of an ensemble of models for the reconstruction of faulty signals, *IEEE T. Nucl. Sci*., 57 (2), p. 793-806 (2010).
23. L. KUNCHEVA, J. BEZDEK and M. SUTTON, On combining multiple classifiers by fuzzy templates, *Proc. the 1998 annual meeting of the North America Fuzzy information Processing Society*, p. 193-197 (1998).
24. W. CHEN, P. D. GADER and H. SHI, Improving dynamic programming-based handwritten word recognition using optimal order statistics", *Proc. the International Conference Statistical and Stochastic methods in Imaging Process III*, San Diego, p. 246-256 (1997).
25. A. VERIKAS et al., Soft combination of neural classifier: a comparative study, *Pattern Recognit. Lett.*, 20, p. 429-444 (1999).

26. K. WOODS, W. P. KEGELMEYER and K. BOWYER, Combination of multiple classifier using local accuracy estimates, *IEEE T. Pattern. Anal.*, 19(4), p. 405-410 (1997).
27. J. F. AUBRY et al., *Project APPRODYN: APPROches de la fiabilité DYNamique pour modéliser des systèmes critiques*, Technical report, collaboration CRAN, EDF R&D, INRIACQFD, UTT-ICD (2012).
28. K. TUMER and J. GHOSH, Error correlation and error reduction in ensemble classifier, *Connect. Sci.*, 8, p. 385-404 (1996).
29. F. ROLI, G. GIACINTO and G. VERNAZZA, Methods for designing multiple classifier systems, *Proc. the 2001 Multi Classifier Systems (MCS'01), Lecture Notes in Computer Science 2096*, p., 78-87 (2001).
30. D. BALLABIO, V. CONSONNI and R. TODESCHINI, The Kohonen and CP-ANN toolbox: a collection of MATLAB modules for Self Organizing Maps and Counterpropagation Artificial Neural Networks, *Chemometr. Intell. Lab.*, 98, p. 115-122 (2009).