Final version published in Control Engineering Practice 73 (2018) 134–148. https://doi.org/10.1016/j.conengprac.2018.01.007

Model-Free Fault Detection and Isolation of a Benchmark Process Control System based on Multiple Classifiers Techniques-a Comparative Study

Hasan Abbasi Nozari[a*], Sina Nazeri[b], Hamed Dehghan Banadaki[c], Paolo Castaldi[d]

[a] Young Researchers and Elites Club, Sari Branch, Islamic Azad University, Sari, Iran
[b] Faculty of Interactive arts and technology , Simon Fraser University, Vancouver, Canada
[c] Young Researchers and Elites Club, Yazd Branch, Islamic Azad University, Yazd, Iran
[d] Department of Electrical, Electronic and Information Engineering, University of Bologna, Via Fontanelle40,47121,Forlí (FC), Italy

[*] Corresponding Author (Email: hasan.abbasi.nozari@gmail.com, Mobile:+989115710933)

A R T I C L E I N F O

Keywords:
Fault detection and isolation
Damadics Benchmark
Blended learning
Ensemble classification scheme
Process actuator systems

A B S T R A C T

This paper presents a combined data-driven framework for fault detection and isolation (FDI) based on the ensemble of diverse classification schemes. The proposed FDI scheme is configured in series and parallel forms in the sense that in series form the decision on the occurrence of fault is made in FD module, and subsequently, the FI module coupled to the FD module will be activated for fault indication purposes. On the other hand, in parallel form a single module is employed for FDI purposes, simultaneously. In other words, two separate multiple-classifiers schemes are presented by using fourteen various statistical and non-statistical classification schemes. Furthermore, in this study, a novel ensemble classification scheme namely blended learning (BL) is proposed for the first time where single and boosted classifiers are blended as the local classifiers in order to enrich the classification performance. Single-classifier schemes are also exploited in FDI modules along with the ensemble-classifier methods for comparison purposes. In order to show the performance of proposed FDI method, it was also tested and validated on DAMADICS actuator system benchmark. Besides, comparative study with the related works done on this benchmark is provided to show the pros and cons of the proposed FDI method.

Nomenclature and Abbreviations	
P51_05	P1 – juice pressure (value inlet)
P51_06	P2 – juice pressure (value outlet)
T51_01	T – juice temperature (value outlet)
P51_01	F – juice flow (1st evaporator inlet)
LC51_03CV	CV – control value (controller output)
LC51_03X	X – servo motor rod displacement
LC51_03PV	PV – process value (juice level 1st evaporator)
TC51_05	Juice temperature (1st evaporator inlet)
T51_08	Juice temperature (1st evaporator outlet)
D51_01	Juice density (1st evaporator inlet)
D51_02	Juice density (1st evaporator outlet)
F51_01	Steam flow
PC51_01	Steam pressure
T51_06	Steam temperature
T51_03	Vapour pressure
T51_07	Vapour temperature

1. Introduction

Nowadays, reliability and availability have become crucial issues in process control system design and received great attention during last decades. An intelligent diagnostics is one of the essential components of modern process control system. Due to manufacturing defects, erosion and corrosion, and other kinds of performance deterioration in system components, and in order to prevent major collapses in plant and system shutdowns, ‘‘early’’ diagnosis of faults is an important factor [1]. There are two approaches that contribute to increase the availability and reliability of industrial process plants: 1) using more solid constructions, redundant equipment that yields additional cost per plant, 2) control systems based on advanced FDI and/or fault tolerant control (FTC) that results in increased costs in development phase [2]. However, the FDI approach has received much of attention as the major contributor to find a simple, low-cost fix to remedy the maintenance problem associated with the industrial process plants. Therefore, it is important to design diagnostics that can automatically detect and isolate occurred faults, maintain the overall functionality of system, and provide an acceptable performance for the faulty system without an unnecessary need to shut down the system.

Generally, a fault diagnosis system is a monitoring system that is used to detect faults and diagnose their location and magnitude in a system [3]. The diagnosis system performs the following tasks: fault detection (FD)—to indicate if a fault occurred or not in the system, fault isolation (FI)—to determine the kind, location and time of detection, and fault identification—to estimate the time behavior of the fault signal. The first two tasks of the system: fault detection and isolation are considered the most important. Fault diagnosis is then very often considered as fault detection and isolation [4]. Moreover, fault diagnosis task can be realized in terms of model-based, knowledge-based, and data-driven approaches which the latter is also referred to as model-free approach [5] and considered in this research. The basic idea is to monitor on-line the measurements of the control system variables without the need for defining explicit dependence laws in time-domain among them. By analysis of the measured variables the operator can decide about the operating mode of the plant and raise alarms. In this context, sometimes the problem can be assimilated to a pattern-recognition problem (see, *e.g.*, [5]). In order to develop data-driven methods, measurements from several input-output process variables are recorded in fault-free and faulty operating conditions from the real process plants. However, in most cases, it is not possible to acquire much of data from the real operating process and it will be more difficult to collect data in faulty operating situation due to safety issues. Therefore, it is more commonplace to employ high-fidelity full-scale simulator of the process plants which is tested and validated against real measurements to collect as much data as required in different operating conditions. Then, the fault diagnosis can be considered as classification problem to assign each pattern of process variable vector to one of the pre-specified class of process operating condition (i.e., nominal and faulty operating situations). In the available literature, many types of classification techniques based on a single classifier have been introduced and employed to address the FDI problems of industrial process control systems. They can range from classic to intelligent methods including the similarity between patterns in the feature space, probabilistic methods, methods based on black box models and/or their combinations [6]. In the last few years, multiple classifier algorithms have become the most important directions of the research in the domain of supervised learning and can find critical roles in developing accurate model-free process monitoring systems. The key aspects of these algorithms are to generate and then combine an ensemble of classifiers, where each classifier is trained on a different sub-set of the data [7].

This paper aims at presenting a general model-free diagnostic system for MIMO nonlinear process plants where reliability and safety issues represent a great concern. Toward this goal, a data-driven fault diagnosis system is designed for nonlinear process plants and tested on a complex process control valve, regulating the inlet flow to the first evaporator of a three-stage evaporation process unit, subsystem found in the sugar factory Lublin, S.A. in Poland, and studied under the European Commission project DAMADICS (Development and Application of Methods for Actuator Diagnosis in Industrial Control Systems) [8]. Both the control valve and the evaporator models are also experimentally tuned and validated by means of real industrial process data [9] and a full-scale simulator is available to generate much of data from all normal and faulty operation conditions. It must be also noted that control valve malfunction is significant when these components are installed in harsh environments like high temperature, humidity, pollution, chemical solvents etc. The determination of the development of small (incipient-hard to detect) [1], [4] faults before they become serious clearly an important influence on the control valve’s predicted lifetime. Valve faults causing process disturbance and shutdown are of major economic concern and shutdown is of major economic concern and can do sometimes be an issue of safety and environmental pollution. In any case, when actuators do not perform correctly, the final product quality is influenced. Therefore, the monitoring of the development of incipient faults is an issue not only for predicting maintenance schedules but also for monitoring the performance of the process concerned [10]. In the reviewed literature, while there are many research methods proposed for model-based FDI of industrial control valves particularly that one proposed in DAMADICS benchmark (see, for example, [11, 12, 13, 14, 15, 16]), a few research efforts have been dedicated to model-free FDI of this industrial actuator system as opposed to the more common model-based approaches. In most cases, proposed data-driven FDI schemes were elaborated in terms of a single classifier, which returns a diagnostic signal corresponding to fault or faultless states of the device. For example, in [17], a single neural network-based classification scheme was proposed to tackle the FDI problem of this benchmark. A model-free faults detection and identification method based on the clusterisation of the multiple diagnostic signals by employing self-organizing maps is presented in [18]. A Two-stage data-driven FDI method on the basis of single perceptron neural network classifier was introduced in [7]. In [19], a fuzzy qualitative simulation algorithm used for FD purposes, coupled with a hierarchical structure of fuzzy neural networks used to perform the fault isolation task on DAMADICS benchmark. A model-free FDtechnique based on the use of a specific spectral analysis tool, namely, the Squared Coherency Functions (SCFs) was proposed in [20]. The detection of a fault was achieved by on-line monitoring the estimate of the squared coherency function, which is sensitive to the occurrence of significant changes in the plant dynamics. The alarm thresholds are based on the estimates of the confidence intervals of the SCF. The Moreover, ensemble-based classification approach has also been exploited to tackle the FDI problem of industrial actuator systems. In [21], a hierarchical diagnosis structure based on ensemble learning with specific feature extraction steps for each branch is presented and Radial Basis Function networks are used as local classifiers. Finally, fuzzy logic aggregates the results for each fault. In [22], a stochastic gradient boosted decision trees based classifier has been developed for the DAMADICS benchmark problem and generates a series of trees with the output of one tree going into the next tree in the series. In [10], ensemble classification schemes are compared to single classifier schemes to detect and isolate abrupt faults of this benchmark problem. Classic classification algorithms such as K-nearest neighborhood, naive Bayes, decision tree, and rules inductions are considered as local and base classifiers in bagging and boosting classification methods. However, although only abrupt faults were considered for detection and isolation purposes, very promising results were not presented and cases of misclassifications were reported.

To our best knowledge, lesser research efforts has been devoted to data-driven FDI of DAMADICS benchmark compared to proposed model based methods. However, among the available model-free FDI methods on process actuator systems, a few ensemble-based classifier solutions have been proposed to tackle this challenging problem. However, it must be noted that data-driven FDI methods can find critical roles when it is not possible to acquire an accurate model of the process in reality. Thus, designing a general model-free FDI system for nonlinear process plants represents a great importance in reliability context. According to the available literature, in all the proposed model-free methods, detection and isolation of incipient faults were excluded. This can be due to the fact that such faults are hardly detectable because they develop very slowly over the time and may be masked by control loop action [9]. In addition, there are 33 process variables provided by the DAMADICS full-scale simulator that are collected from the whole operating process and 17 of them are associated with the first control valve system feeding its related evaporator that should be used in developing a data-driven FDI framework (see Fig. 2). However, in most of the present works, lesser variables have been used for FDI task. In the present paper, we will investigate the detection and isolation of all possible 19 faulty scenarios, while exploiting all 17 available input-output process variables as the inputs to the proposed FDI schemes. On the other hand, it is worth stressing that a large ensemble of diverse collaborative models leads to an accurate prediction system. Furthermore, a large ensemble of supervised learning techniques can improve the accuracy of the whole ensemble by clever blending. Accordingly, the number of input attributes in the second-level dataset is multiplied by the number of classes. This can be seen as a limitation of the proposed BL method, and therefore, a compromise has to be made between the accuracy of the multiple-classifiers scheme and the number of collaborative models which are used in the ensemble.

The remainder of the paper is organized as follows. In section 2, description of the considered process control system as well as the possible faulty scenarios is elaborated. Section 3 introduced the proposed FDI systems on the basis of series and parallel forms. In Section 3.1 and its subsequent subsections 3.1.1 and 3.1.2, multiple classifier based FDI schemes based on dependent and independent resembling techniques are described. The proposed multiple classifier FDI method based on blended learning is explained in Section 3.2. Section 3.3 is concerned with the employed statistical and non-statistical single classifiers and their descriptions. In Section 4 and its Subsections 4.1 and 4.2, simulation results, related discussions, and a brief comparative study are provided, respectively. Finally, the main concluding remarks are given in Section 5.

2. Monitored process plant and faulty scenarios descriptions

The monitored system comprises an industrial control valve pneumatically actuated and an evaporator located downstream the valve, where a mixture of sugar and water (juice) pass through, as shown in Fig. 1

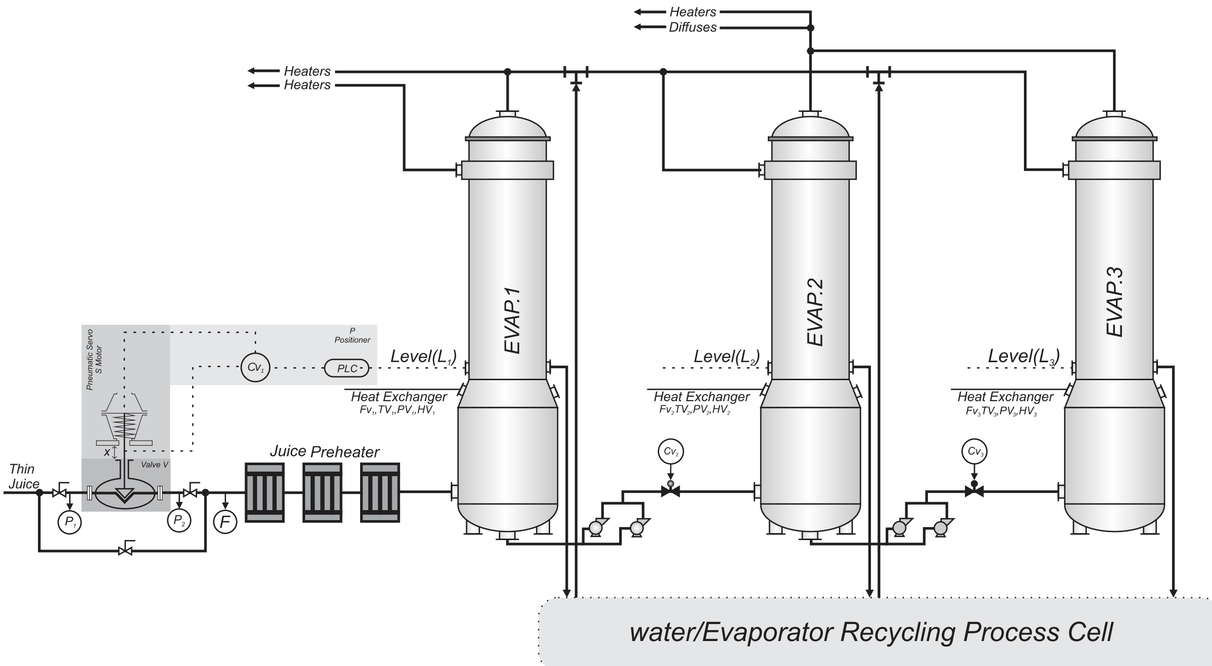


Fig. 1 Piping and instrumentation diagram of the monitored Damadics process plant.

Table 1 Fault scenarios description.

Fault	Description	Abrupt			Incipient	Direction
		Small	Medium	Big		
f1	Valve clogging	×	×	×		↗
f2	Valve plug or valve seat sedimentation			×	×[1]	↗
f3	Valve plug or valve seat erosion				×[1]	↗
f4	Increased of valve or bushing friction				×[2]	↗
f5	External leakage (leaky bushing, covers, terminals)				×[1]	↗
f6	Internal leakage (valve tightness)				×[1]	↗
f7	Medium evaporation or critical flow	×	×	×		↗
f8	Twisted servo-motor's piston rod	×	×	×		↗
f9	Servo-motor's housing or terminals tightness				×[1]	↗
f10	Servo-motor's diaphragm perforation	×	×	×		↗
f11	Servo-motor's spring fault			×	×[1]	↗
f12	Electro-pneumatic transducer fault	×	×	×		↘
f13	Rod displacement sensor fault	×	×	×	×[3]	↗
f14	Pressure sensor fault	×	×	×		↘
f15	Positioner feedback fault			×		↗
f16	Positioner supply pressure drop	×	×	×		↗
f17	Unexpected pressure change across the valve			×	×[2]	↗
f18	Fully or partly opened bypass valves	×	×	×	×[1]	↗
f19	Flow rate sensor fault	×	×	×		↘

[1] Low fault development speed preset for simulation as (100% Fault Strength / 1 day).
[2] Medium fault development speed preset for simulation as (100% Fault Strength / 1 hour).
[3] High fault development speed preset for simulation as (100% Fault Strength / 15 minutes).
'↗', ascending development, '↘' , descending development.

The pneumatic actuator with a PID controller determines the rod position (Xrod1) of the control valve determining the inlet flow (Fi) towards the evaporator. The evaporator is basically a heat exchanger consisting of an inlet and an outlet flow of juice and a separate outlet flow of evaporated water circulating in a long pipeline. The working principle of the evaporator is simple: a fluid mixture of water and sugar enters at the inlet with a given ratio sugar mass/total mass (Xi); this is designated the inlet flow Fi; part of the water content of the mixture evaporates due to the temperature raise achieved with the heat exchanger; evaporated water is removed through the vapour outlet flow (W); the resulting fluid mixture, with an increased ratio sugar mass/total mass (Xo), is returned at the outlet; this is designated the outlet flow Fo available from industrial decentralized control systems (DCS) and/or supervisory control and data acquisition systems (SCADA) [8, 9]. All 19 pre-specified fault scenarios occur at the valve, so that, for diagnosis purposes the evaporator acts only as providing extra variables for a more robust and reliable diagnosis. A complete analytical model, reproducing normal and faulty operation of both valve and the evaporator were implemented as SIMULINK models in MATLAB standard platform to generate the training and test data. The full-scale DAMADICS simulator allows choosing only one from nineteen available faults (due to this, only scenarios with single faults were taken into account). A part of them is considered only as incipient faults or as abrupt faults (there are three sizes of abrupt faults: small, medium and big) and some of them as both. Descriptions regarding all faulty conditions which occurred in four different parts of the monitored process control system namely control valve, pneumatic servo motor, control valve positioner (*i.e.*, I/P convertor), and compressed air supply are givens in Table 1. The characteristics of each fault are described in [9]. It is also worth noting that some methods, e.g. structural analysis FDI methods will not be suitable as system model dynamics are not provided. FDI methods based on the use of identification, together with unknown input observers, Kalman filters, neural networks or neuro-fuzzy methods are most suitable as the models derived do not depend upon analytical knowledge of the dynamics of the overall actuator system. Pattern recognition-based methods (as proposed in this paper) can also be exploited here as there is no need of analytical system model dynamics [8, 9].

3. Proposed data-driven FDI schemes

The proposed hybrid FDI schemes are shown in Fig. 2 and split into two forms: series and parallel. In the series form, two separate modules are considered for detection and isolation tasks where in FD module the decision on whether a fault has occurred throughout the monitored process plant or not is made and once a fault is detected and alarm will be signaled and activated the fault isolation (FI) module. In this module, the location of the occurred fault in the process will be indicated to allow the field operators to take actions such as reconfiguration, maintenance, repair or other operations, While, in the parallel form only one module is considered to perform both detection and isolation tasks in parallel. However, in both forms, each class of operating condition is coded with an n-bit (binary) representation. The real-valued response of each module is transformed into the binary one by considering the distance between the module output and each class of system behavior. Hence, the binary representation which results in the shortest Euclidean distance is selected as the module binary output:

$$i = \arg\left(\min\left(\left\|\hat{T} - \left(T_j\right)_2\right\|\right)\right), j = 1 \ldots n_f \qquad (1)$$

where $\|.\|$ denotes Euclidean distance and n_f is the number of classes, and $\hat{T}$ denotes the outputs of the classifier, and $(T_j)_2$ is the binary representation of j-th class. Consequently, the binary representation of the classifier output can be determined in the form of $\hat{T} = \left(T_j\right)_2$

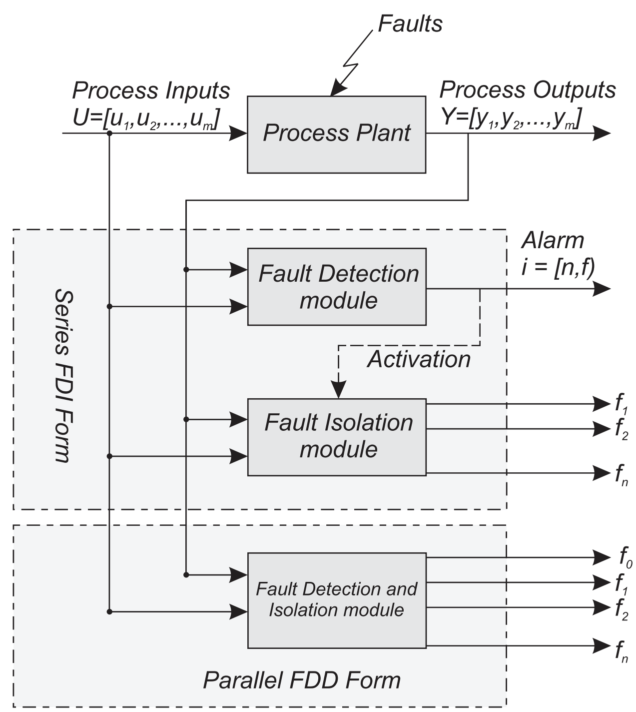


Fig. 2 Proposed model-free FDI frameworks: series forms versus parallel form.

Each module of the given FDI forms must include a decision making scheme in order to associate each pattern of process variables vector (features) to each class of system behavior. In the following sections, several classifications schemes are introduced and will be tailored as the decision making logics to FDI modules.

3.1. Multiple classifier based FDI methods

In general, the ensemble classification method weighs several individual classifiers, and aggregates them in order to acquire a classifier that outperforms every one of them. Ensembles can be realized through dependent and independent schemes [7]. That is, in a dependent scheme the output of a classifier is used to construct the next classifier, while, in an independent scheme each classifier is constructed independently and their outputs are combined using data fusion techniques. The most well-known dependent and independent multiple classification approaches are boosting and bagging techniques.

3.1.1. Dependent multiple classifier-based FDI scheme

Boosting also referred to as arcing is a general technique for reinforcing the accuracy of a weak classifier such as rules set or decision trees. In other words, it operates by iteratively running a weak classifier on various training data, and then, the classifiers produced by the weak learners are blended into a single composite strong classifier in order to achieve a better performance compared to those of weak classifiers. Adaptive Boosting (AdaBoost) [23] is a popular ensemble technique that improves the simple boosting algorithm via an iterative process shown in Fig. 3. However, boosting techniques such as AdaBoost are usually incorporated into other powerful decision schemes such as decision trees to enhance the robustness of the decision schemes against sudden changes and outliers in training data [24]. Therefore, in the present research, four advanced and well-practiced boosted decision trees namely C5.0, Conditional inference tree (CIT), tuned normal CART (TN-CART) and CART ordinal responses (CART-OR) are briefly introduced and exploited for FDI tasks.

Algorithm1. (AdaBoost)

C: Base Classifier/Learner

N: Number of Iterations

D: Data Set $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$

D_1(i)=1/n

For j=1 to N

Construct classifier Ci with distribution Di

Calculate: $\varepsilon = \sum_{j:C\neq yi} D_j(i)$

If (ε>0.5) Then

Break

End If

Calculate: $\alpha_j = \frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$

Update: $D_{j+1}(i) = D_j(i)\exp\left(-\alpha_j y_i C_j(x_i)\right)$

Normalize $D_{j+1}(i)$ so that it will be a distribution.

End For

Fig. 3 Adaptive Boosting algorithm pseudo-code.

A. C5.0 Tree

C.5.0 tree algorithm (also known as boosted tree) is a boosting method based on boosting and winnowing that performs similar to AdaBoost. Winnowing is a feature selection step conducted before modeling [25]. When the first tree is created, weights are determined and, in subsequent iterations, weighted trees or rule-sets are generated. Successive trees (or rule-sets) are constrained to be about the same size as the initial model. The final prediction is a simple average of class probabilities generated from each tree or rule-set. In other words, the dataset is randomly split and an initial model is fit. Each predictor is removed in turn and the effect on model performance is determined using the other half of the random split. Predictors are flagged if their removal does not increase the error rate. The final model is fit to all samples of the training set using only the un-flagged predictors [26].

Generally, C5.0 algorithm follows basic rule given a set of S cases and it firstly grows an initial tree using the following two-step divide-and-conquer algorithm [27]:

- If all the cases in S belong to the same class or S is small, the tree is a leaf labeled with the most frequent class in S.
- Otherwise, choose a test based on a single attribute with two or more outcomes. Then, make this test the root of the tree with one branch for each outcome of the test, partition S into corresponding subsets S1, S2, according to the outcome for each case, and apply the same procedure recursively to each subset.

B. Conditional Inference Tree

Conditional inference tree (CIT) also known as C-Tree is a non-parametric class of regression trees for recursive binary splitting embedding tree-structured regression models into a well-defined theory of conditional inference procedures based on permutation tests, attempting to distinguish between significant and insignificant improvements [28]. It is applicable to all kinds of regression and classification problems, including nominal, ordinal, numeric, censored as well as multivariate response variables and arbitrary measurement scales of the covariates. CIT algorithm is known as unbiased recursive conditioning algorithm [29] that recursively perform univariate splits of the dependent variable based on values on a set of covariates. Such algorithms usually employ information measures (*e.g.* Gini coefficient) for selecting the current covariate. CIT also separates the variable selection from the splitting procedure. This results in basically three steps in the conditional inference tree procedure. The first one concerns variable selection, the second one chooses the splitting methodology, and the last one is the recursive application of the first two steps. The reader is referred to [30] for a detailed descriptions of these steps. It is also worth stressing that CITs use well-known, established statistical concepts for variable selection and stopping. The resulting tree models are easier to communicate to practitioners [29].

C. *Tuned Nominal CART and CART Ordinal Responses*

Classification and Regression Tree (CART) employs decision trees to build predictive model. The model achieved promising results to discriminate evolutionary divergence [31]. It is worth noting that CART does not require variables to be selected in advance, because it will itself identify the most significant variables and eliminate non-significant ones. The CART model is also known as Recursive Partitioning (RPART) and available in RPART library of R programming Repository [32]. The tuned nominal CART (TN-CART) algorithm is briefed in the following 6 steps:

Step1: Dataset is split by asking the questions $x_j<a$ and searching for the "best" split -"best" variable and value

Step2: "Best" split is defined by the splitting rule (by maximizing the Gini function given in Eq. 2)

Step3: Parent nodes are always split into exactly two child nodes

Step4: Procedure is repeated by treating each child node as a parent

Step5: Stopping rule decides when to stop splitting and tree is complete

Step6: Classes are assigned to terminal nodes

The Gini index (known as impurity function) [31] is given as follows:

$$P_1 \sum_{j=1}^{K} p^2\left(j \middle| t_1\right) + P_r \sum_{j=1}^{K} p^2\left(j \middle| t_r\right) \qquad (2)$$

Where P_r is the probability to get left and right nodes $j \in (1\sim K)$ - class index, K denotes the number of classes in a sample, and $p(j\ |t)$ is the conditional probability of class j given that we are in node t.

CART Ordinal Responses allow the user to build classification trees for ordinal responses within the CART framework [33], ordinal data is a categorical, statistical data type where the variables have natural, ordered categories and the distances between the categories is not known. The trees are grown using the extended Gini impurity function given in Eq. 3, where the misclassification costs are given by the absolute or squared differences in scores assigned to the categories of the response.

$$\sum_{k=1}^{J}\sum_{l=1}^{J} C\left(\omega_k \middle| \omega_l\right) p\left(\omega_k \middle| t\right) p\left(\omega_l \middle| t\right) \qquad (3)$$

Where $C(\omega_k|\omega_l)$ represents the misclassification cost of assigning category ω_k to a sample unit belonging to category ω_l . Clearly, $C(\omega_j\ |\omega_j\ ) = 0$ for $j = 1\sim K$. When $C(\omega_k|\omega_l) = C(\omega_l\ |\omega_k)\ \forall\ k$, these misclassification costs are symmetric and can also be interpreted as dissimilarities between pairs of categories of Y . Pruning is based on the total misclassification rate or on the total misclassification cost (for more details reader is refereed to [33]).

3.1.2. Independent multiple classifier-based FDI schemes

The most well-known independent methods are bagging (bootstrap aggregating) and random forests which will be explained in the following sub-sections.

A. *Bagging based ensemble classification scheme*

Bootstrap aggregating, also known as bagging, involves having each model in the ensemble vote. Bagging technique is designed to improve the performance in terms of accuracy and stability, it also prevents over-fitting problem by minimizing variance.

In order to promote model performance, bagging trains each model in the ensemble using a randomly drawn subset of the training set (See Fig. 4 for detailed algorithm).

Algorithm2. (Bagging)

D: Data Set $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$

N: Number of Iterations

S: Subset Size

D_1(i)=1/n

For *i*=1 to *N*

  D_i=choose *S* samples from *D* with replacement

  Construct classifier *C* using D_i

End For

Majority voting to calculate the total prediction:

$$pt = \arg\left(\max_{y\in Y} \sum_{1:pt(x)=y} (1)\right)$$

Fig. 4 Bagging algorithm pseudo-code.

Fig. 4 shows the process of bagging technique. Given some database of training data, we can take S samples from this database with replacement. In other word, our training set T is divided into T_i subsets. Using samples taken from the training example database, we can train our classifiers independently on each of these snapshots of data. After the training has completed, we are left with Cm classifiers. In this stage, an unknown new sample is presented and a prediction is made on it using each of the Cm classifiers. The final prediction is made by selecting the most common prediction from each of the classifier's Cm. The finial classification of the test sample made from the target classifiers is called a voting scheme where the prediction of each target classifier is a "vote" towards the final prediction and can be simply done by averaging the predictions.(Fig. 5)

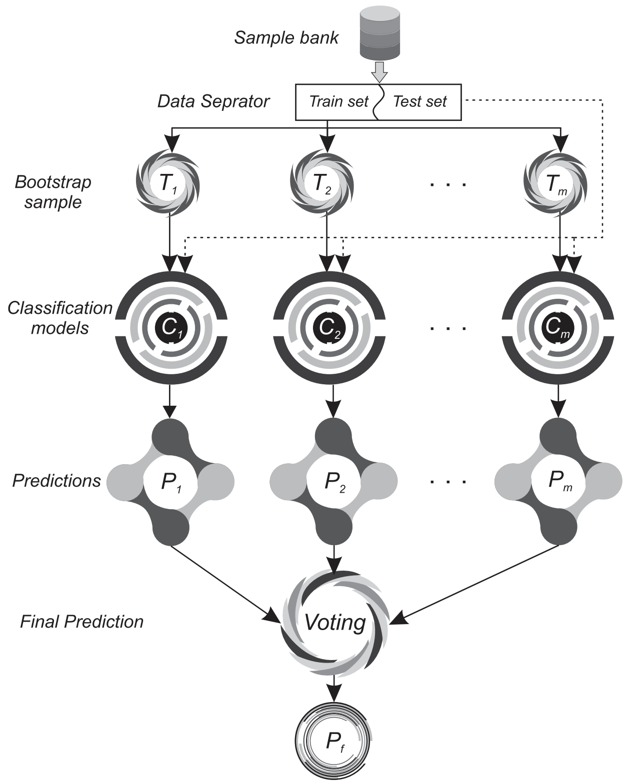


Fig. 5 Structure of bagging based ensemble classification scheme

B. Random Forest based ensemble classification scheme

A Random Forest ensemble (also known as random subspace) [34] uses a large number of individual, un-pruned decision trees. The individual trees are constructed using the algorithm presented in Fig. 6.The input parameter M represents the number of input variables that will be used to determine the decision at a node of the tree. This number should be much less than the number of attributes in the training set. Note that Bagging can be thought of as a special case of Random Forests obtained when M is set to the number of attributes in the original training set.

Algorithm3. (Random Forest)

DTC: Base decision tree classifier

N: Number of iterations

D:Data set $\{(x_1,x_1),(x_2,x_2),\ldots,(x_n,x_n)\}$

S: Subset size

M: Attribute number in each node

For *i*=1 to *N*

D_i=choose S samples from D with replacement

Construct classifier *C* using *DTC (M)* on *D*

End For

Fig. 6 Random Forest algorithm pseudo-code.

3.2. Proposed multiple classifier-based FDI method based on blended learning

The basic ways of aggregating classification methods is bagging that creates solutions to be combined. However, the proposed ensemble classifier combines the existing solutions by blending technique in that in order to determine the value of each new local classifier, the algorithm re-evaluated the aggregation set while stacking [35] each new model sequentially. The BL algorithm concerned with combining multiple classifiers generated by different learning algorithms $L_1,\ldots,L_N$ on a single dataset D, which is composed by a feature vector $si=(xi,yi)$. The stacking process can be broken into two phases: 1) Generate a set of base-level classifiers $C1,\ldots,$, Where $Ci=Li(S)$, and 2) Train a meta-level classifier to combine the outputs of the base-level classifiers. Unlike averaging method for aggregating prediction in bagging method, using final classifier in stacking can cast weight to predictions to prioritize those with higher performance. The proposed ensemble classifier scheme is based on stacking to aggregate classifiers and employs diverse classifiers (see Section 3.3) and integrates single and boosted classification schemes to obtain the best accuracy rate. The briefed algorithm of the proposed BL-based multiple classifier based decision making scheme is also provided in Fig. 7.

Algorithm4. (Blended Learning)

D: Data Set $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$

C: Individual learner / base classifier

Ct: Meta classifier/voting scheme

First-level learning algorithms $LA_1, \ldots, LA_m$

Second-level learning algorithm LA

First-level predictions $P_1, \ldots, P_m$

Second-level prediction P_t

$D_{new}=\emptyset$

For i=1 to m

  $C_i=LA_i(D)$

End

For i=1 to n

  For j=1 to m

    $z_{ij}=Ct(x_i)$

  End

  $D_{new} = D_{new} \cup \{(z_{i1},z_{i2}, \ldots, z_{im}),y_i)\}$

End

$Ct=LA(D_{new})$

$Pt=Ct(P_1, P_2, \ldots, P_m)$

Fig. 7 Proposed blended learning based classifiers ensembling algorithm pseudo-code.

3.2.1. Employed single classifier schemes

The model-free FDI systems presented in Sections 3 can be realized on the basis of basic classification methods. However, the following classification schemes can be both used as single classification schemes and base/local classifiers in multiple classification schemes. In the present comparative study, diverse range of classifiers is obtained to leverage the process features. These classifiers have been widely utilized in most literature for sequence classification and regression tasks.

A. Support Vector Machine

A support vector machine (SVM) model is a representation of the samples as points in space, mapped so that the samples of the separate categories are divided by a clear gap that is as wide as possible. New samples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using kernel bases, implicitly mapping their inputs into high-dimensional feature spaces($\phi: X \rightarrow H$) by a kernel function, , *i.e.*, a function returning the inner product $\langle \phi(x), \phi(x') \rangle$ between the images of two data points x, x' in the feature space as follows:

$$k(x,x') = \langle \phi(x), \phi(x') \rangle \tag{4}$$

The Gaussian or radial basis function (RBF) is a popular kernel function used in various kernelised learning algorithms [36]. In classification, support vector machines separate the different classes of data by a hyper-plane given as follows:

$$\langle w, \phi(x) \rangle + b = 0 \tag{5}$$

Corresponding to the decision function as follows:

$$f(x) = sign(\langle w, \phi(x) \rangle + b) \tag{6}$$

B. *Weighted k-Nearest Neighbors*

K-nearest Neighbor rule (KNN) has been one of the most well-known supervised learning algorithms in classic pattern classification. The class label assigned to an example is based on the similarity of this example to one or more prototypes. Typically, the similarity is defined in a geometrical sense using a certain distance. The Weighted k-Nearest Neighbors (WKNN) is an extended version of KNN technique, where the distances of the nearest neighbors can be taken into account [37] In WKNN, the question is" how many of the nearest neighbors out of the complete dataset fall into the different classes". In WKNN, the closer neighbors are weighted more heavily than the farther ones, using the distance-weighted function. Considering function d as distance function, the weight W_i for i-th nearest neighbor of the query x' is defined as follow:

$$W_i = \begin{cases} \dfrac{d(x', x_k^{NN}) - d(x', x_i^{NN})}{d(x', x_k^{NN}) - d(x', x_1^{NN})} & if \quad d(x', x_k^{NN}) \neq d(x', x_1^{NN}), \\ 1 & if \quad d(x', x_k^{NN}) = d(x', x_1^{NN}). \end{cases} \tag{7}$$

Then, the classification result of the query is made by the majority weighted voting:

$$y' = \arg\max_{y} \sum_{(x_i^{NN}, y_i^{NN}) \in T'} W_i' \times \delta(y = y_i^{NN}). \tag{8}$$

According to the Eq. above, it can be see that a neighbor with smaller distance is weighted more heavily than one with greater distance: the nearest neighbor gets weight of 1, the furthest neighbor a weight of 0 and the other neighbors' weights are scaled linearly to the interval in between.

C. *Discriminant Function Analysis*

Discriminant function analysis (DFA) is a statistical analysis to predict a categorical dependent variable (called a grouping variable) by one or more continuous or binary independent variables (called predictor variables) [38]. Discriminant function analysis is useful in determining whether a set of variables is effective in predicting category membership. Discriminant analysis works by creating one or more linear combinations of predictors, creating a new latent variable – that are not directly observed but are rather inferred (through a mathematical model) from other variables that are observed (directly measured) - for each function. These functions are called discriminant functions. The number of functions possible is either Ng-1 where Ng = number of groups, or p (the number of predictors), whichever is smaller. Given group j, with R_j sets of sample space, there is a discriminant rule such that if $x \in R_j$, then $x \in j$ .Discriminant analysis then, finds "good" regions of R_j to minimize classification error, therefore leading to a high percent correct classified in the classification table. In this study, two types of Discriminative Function Analysis namely Diagonal Discriminative Analysis (DDA) and Linear Discriminative Analysis (LDA) are used [39]. The classifier is trained using James-Stein-type shrinkage estimators. Variable selection is based on ranking predictors by CAT scores (LDA) or t-scores (DDA). A cutoff is chosen by false non-discovery rate or higher criticism thresholding (For details see [40, 41]).

D. Naive Bayes

This is a simple probabilistic classification method which is based on Bayesian theory [42, 43]. However the naive Bayes classifier considers each of existing features independently:

$$P(d_i|V_1,\cdots,V_n) = \frac{P(V_1,\cdots,V_n|d_i)P(d_i)}{P(V_1,\cdots,V_n)} \tag{9}$$

Taking into account this assumption, the Bayesian equation (4) can be transformed to (5), where the denominator of the equation is replaced by a constant C and the conditional probability is calculated by the multiplication.

$$P(d_i|V_1,\cdots,V_n) = C.P(V_1|d_i).\cdots.P(V_n|d_i).P(d_i) \tag{10}$$

The degrees of beliefs for the classification results are equal to probability values obtained from the Bayesian equation.

E. Genetic Cooperative-Competitive Fuzzy Rule Based Learning Method

Fuzzy rule-based systems (FRBSs) constitute an extension of classical rule-based systems, because they deal with fuzzy rules instead of classical logic rules [44]. Fuzzy rules are based on representing the consequent as a polynomial function of the inputs. The generic expression of the fuzzy rules is the following:

If x_1 is A_1 and $\cdots$ x_n is A_n THEN

$$y = P_1.x_1 + \cdots + P_n.x_n + P_0 \tag{11}$$

Focusing on the other system type, the FRB is composed of a collection of fuzzy rules with the following structure:

IF x_1 is A_{i1} and $\cdots$ x_n is A_{in} THEN y is B_i (12)

Where $x_1,\cdots,x_n$ and y are the input variables and the output variable, respectively, and A_{in} is membership function. It must be noted that in designing FRBSs, a compromise has to be reached between interpretability and accuracy [45]. However, such a trade-off is more difficult to reach when the problem to be solved presents a high dimensionality, that is, a high number of input features and/or a high number of instances or examples. In order to overcome the problem of the high dimensionality in Fuzzy Rule-Based Classification Systems (FRBCS), a method that carries out an embedded feature selection in the rule learning, getting a compact rule base by means of Genetic Programming (GP)was proposed in [46]. This algorithm so-called GP-COACH-FRBCS (which stands for GP-based evolutionary algorithm for the learning of Compact and Accurate FRBCS for High dimensional problems), learns disjunctive normal form fuzzy rules (generated by means a context-free grammar) and it uses a competition mechanism between rules (the token competition) which simultaneously maintains the diversity into the population and deletes irrelevant rules during the learning process, leading to compact FRBCSs (with few rules and conditions per rule) with a high generalization capability.

F. *Partial Least Squares*

Partial least squares (PLS) [47] is a statistical classification and regression model that is based on projecting independent variables (features) and dependent variables into new space in order to reduce the size of predictive variables by omitting highly correlated variables.

The general underlying model of multivariate PLS is:

$$X = TP^T + E$$

$$Y = UQ^T + F \tag{13}$$

Where X is an n*m matrix of predictors, Y is an n*p matrix of responses; T and U are n*l matrices that are projections of X (the X score, component or factor matrix) and projections of Y (the Y scores), respectively. P and Q are, respectively, m*l and p*l orthogonal loading matrices, and matrices E and F are the error terms, assumed to be independent and identically distributed random normal variables. The decompositions of X and Y are made so as to maximize the covariance between T and U.}

G. Robust Soft Independent Modeling of Class Analogies

SIMCA stands for Soft Independent Modeling of Class Analogies, and is very suitable for classifying high-dimensional observations, because it incorporates principal component analysis (PCA) for dimensionality reduction [48]. As PCA is applied to each group separately, SIMCA provides additional information on the different groups such as the relevance of the different variables and measures of separation. This method can be robustified in a straightforward way by combining a robust PCA method with a robust classification rule on the basis of robust covariance matrices. As in this case all the groups are merged, the preprocessing step with PCA is mainly used as an overall dimension reduction technique.

In the original SIMCA method, new observations are then classified by means of their deviations to the different PCA models. We will call this deviation as orthogonal distance because it represents the Euclidean distance of an observation to the PCA subspace.

Because this approach does not completely exploit the benefit of applying PCA in each group separately, it was already suggested to include another distance in the classification rule, namely the distance to the boundary of the disjoints PCA models. For each of them groups, a multidimensional box is constructed by taking into account the scores t_i^l for $i = 1,2,\cdots,n_l$ Here $t_i^l = (t_{i1}^l, t_{i2}^l, \cdots t_{ikl}^l)'$ represents the K_j dimensional score of the ith observation in the training set X. The boundary for each set of scores is defined by looking at the minimal and maximal value of the scores component-wise:

$$(\min_{i=1,\cdots,n_l}(t_{ij}^l) - cd_j^l, \max_{i=1,\cdots,n_l}(t_{ij}^l) + cd_j^l) \tag{14}$$

Here, d_j^l s the standard deviation of the jth component of the t_i^l :

$$d_j^l = \sqrt{\frac{1}{nl-1}\sum_{i=1}^{nl}(t_{ij}^l - \frac{1}{nl}\sum_{a=1}^{nl} t_{aj}^l)^2} \tag{15}$$

The parameter c can vary but is usually taken equal to 1.

H. Perceptron Neural Network

Perceptron neural network (PNN) [1, 49] is a supervised learning algorithm that learns a nonlinear function $f(.) : R^m \to R^0$ by training on a dataset, where m is the number of dimensions for input and O is the number of dimensions for output. Given a set of features $X = x_1, x_2, \cdots, x_m$ and a target y, it can learn a non-linear function approximator *f* for either classification or regression as follows:

$$y = f(x_1, x_2, \cdots, x_m) \tag{16}$$

The MLP consists three main layers where there can be one or more non-linear layers between input and output layer, called hidden layers. The neurons in the hidden layers can be hyperbolic tan function, while, the neurons in the output layer are usually linear. Training of the PNN weight parameters is performed by Levenberg- Marquardt algorithm [49.].

4. Simulation results

The learning procedure of all the presented classification schemes are carried out offline by using the available dataset (with full size of around 8000 samples) of the DAMADICS benchmark including artificial faults that were introduced at October 30, November 9, November 17, November 20, 2001(see Table 1)[9].

4.1. Results of the employed FDI methods and related discussions

One of the essential phases in training a classifier is data preparation. Fig. 8-11 display the time evolution of the process variables in transients of the selected abrupt and incipient artificial faulty cases f16, f17, f18, and f19 introduced in actuators 1, 2, 3. It is well-know that abrupt of faults are simply detectable due to sudden change in the sensitive process variable signal as a result of a particular fault [1]. For example, with reference to Fig. 8, the sudden change of LC51-03X signal can be observed at around 58800 second when the fault f18 (*i.e.*, partly opened bypass valve on actuator 1) starts affecting the process. On the other hand, the signal variations of selected process variables for incipient faulty cases such as f3 and f4 are in ramp forms. Hence, it is apparent that these faulty cases are hardly detectable compared to abrupt or intermittent fault types owing to smooth development of these types of faults over the time. Additionally, both types of faults can be also masked in the output by the feedback control action [9]. For FDpurposes, the available dataset should be split into equal snapshots, where the first part is associated with the nominal operation condition of the system (fault-free state) and the second part corresponds to faulty situations. Moreover, the prepared data should be again divided into two sets namely learning and validation sets. For the proposed BL classification scheme where a classifier is trained to make the final decision based on the outputs of base classifiers, the number of data sets will be considered as four(*i.e.*, two for learning and validation of base classifiers and two remaining sets for training and validation of the final classifier). In the present work, the overall size of the dataset for was about 10000 samples, where 5000 samples were collected from the data of nominal operating condition and remainder of them extracted from the data of all possible faulty scenarios. In addition, the prepared data sets for FI task include measurements from all potential set of faulty scenarios (*i.e.*, 19 cases) occurred in DAMADICS benchmark. The number of samples for each faulty case was the same for all sets. Similar to FD phase, four separated datasets for training and validation of the proposed BL based classification scheme were arranged. The size of data group for a single fault encompasses roughly 600 samples, while the full dataset size is around 8000 samples. After preparation of the required data sets, the FDI tasks are performed in terms of both series and parallel forms presented in Section 3. As mentioned in this section, several ensemble classification schemes based on boosting, bagging and BL techniques were introduced to carry out FDI task. However, in the present study, the single classification schemes are also exploited along with multiple classifiers methods for comparison purposes. That is, 8 single classifiers namely WKNN, NB, SVM, PLS, FRBS, DFA, RSIMCA, and PNN are taken into consideration (Reader is referred to Appendices A and B for descriptions and values of parameters used to develop the classification schemes). The efficiencies of the proposed single and multiple classifiers-based FD modules used in series diagnostic framework are given in Table 2. For performance analysis of the proposed FD modules in series FDI form, three performance indices including sensitivity (*sens.*), specificity (*spec.*), and accuracy (*accu.*) [1] are calculated on the basis of the confusion matrix which was generated after the classifier validation phase for all proposed FD schemes in series form as follows:

$$Sens = TP \big/ (TP + FN) \tag{17}$$

$$Spec = TN \big/ (TN + FP) \tag{18}$$

$$Accu = (TN + TP) \big/ (TN + TP + FN + FP) \tag{19}$$

Where true positive (*TP*) means the proportion of actual faulty cases which are predicted as faults, true negative (*TN*) implies proportion of actual normal case which are predicted as normal operating condition, false positive (*FP*) denotes the proportion of actual faulty cases which are predicted as normal operating situation, and false positive (*FN*) refers to the proportion of actual normal scenario which is predicted as faulty operating situation. It is also noted that this measure can be directly compared with false and true detection/isolation rates proposed in the DAMADICS simulator documentation [9]. Table 3-4 also demonstrate the results of FI in terms of single and multiple classifiers realized by series FDI scheme presented in Fig. 2. Here, the same classification schemes as those employed in FD module are also employed and the related parameters of their structures are also given in Appendices A and B. The last columns of Table 3 and Table 4 indicated as “overall” include values of the general efficiency of a particular classification scheme against all classes of system faulty operations and other columns contain effectiveness of classification schemes for each individual class of system’s faulty operation.

Table 2 Performance results of the proposed FD systems using series form.

Method	Performance indices		
	Accu.	**Sens.**	**Spec.**
SVM	0.9745	0.9673	0.9745
RSIMCA	0.7885	0.7383	0.7885
PLS	0.8325	0.8222	0.8325
PNN	0.9688	0.9673	0.9688
GP-COACH-FRBCS	0.9958	0.9943	0.9958
WKNN	0.9683	0.9388	0.9683
DDA	0.8998	0.8899	0.8998
NB	0.8998	0.9001	0.8998
LDA	0.8878	0.8698	0.8878
CIT	0.9818	0.9713	0.9818
RF	0.9962	0.9953	0.9962
TN-CART	0.9825	0.9787	0.9825
CART-OR	0.9350	0.9254	0.9446
C.50	0.9977	0.9954	0.9977
BL	**0.9980**	**0.9970**	**0.9980**

As discussed in Section 3, with reference to Fig. 2, FDI tasks can be carried out also on the basis of the proposed parallel form where both detection and isolation tasks can be realized in parallel through a single module. In this respect, the FDI module includes all the available process variables as the inputs and 20 output channels where one output is associated for normal operating condition (f0) and the remainder of the outputs are related to faulty cases fi (i=1~19). In other words, If the output associated with f0 is triggered, it means that the process is in nominal operating condition and no fault has been occurred, while, an output related to a particular fault is activated, it shows that a fault has been taken place and it related type and location is also determined. That is, it this case, FDI tasks are performed simultaneously using a solitary module. The employed decision schemes in this framework can be also the same single or multiple classification structures presented in Sections 3-1 and 3-2. The performance results of the proposed parallel FDI framework are also reported in Table 5. The last column of this table indicated as "overall" includes values of the general efficiency of a particular classification scheme against all classes of system operations and other columns contain effectiveness of classification schemes for each individual class of system operation (including normal and faulty conditions).

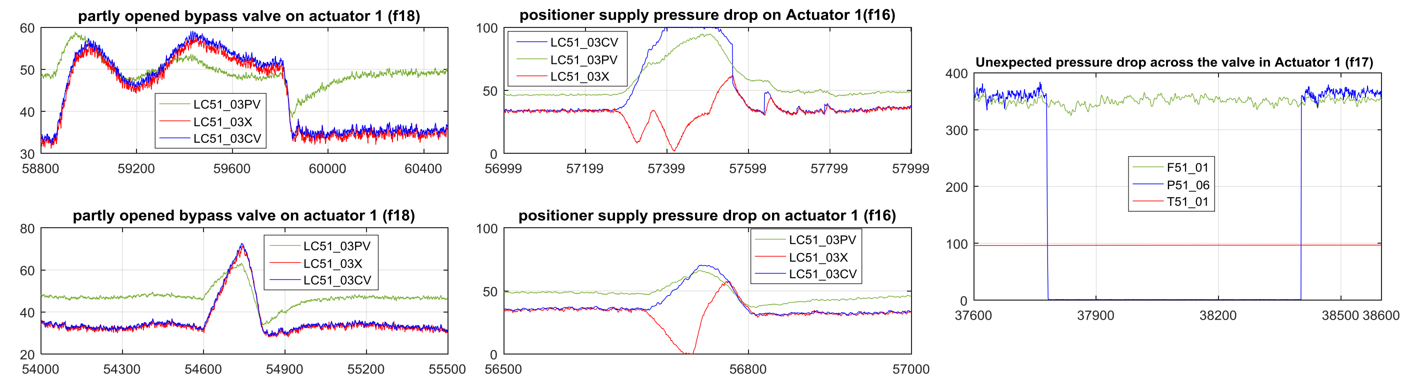


Fig. 8. Signal variations of process variables for abrupt faulty cases on actuator1

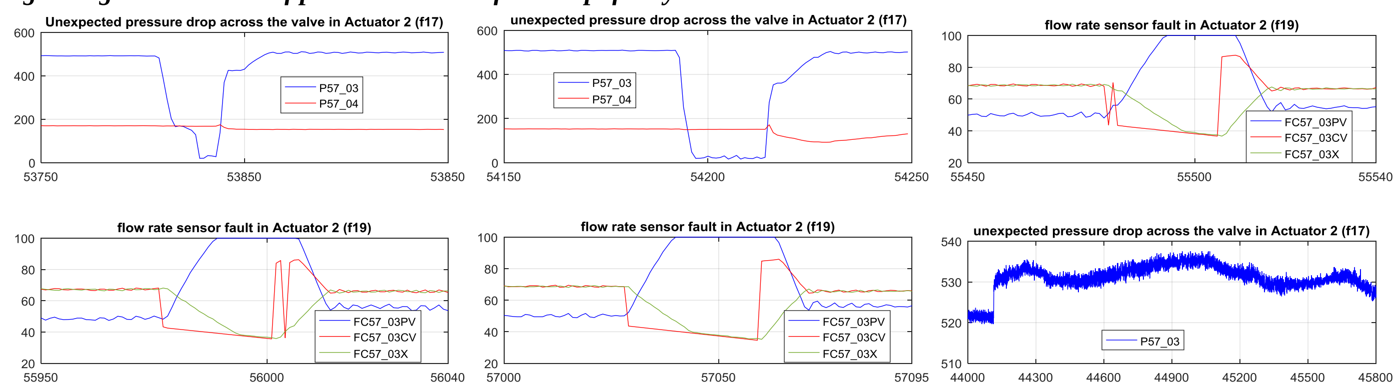


Fig. 9. Signal variations of process variables for abrupt faulty cases on actuator2

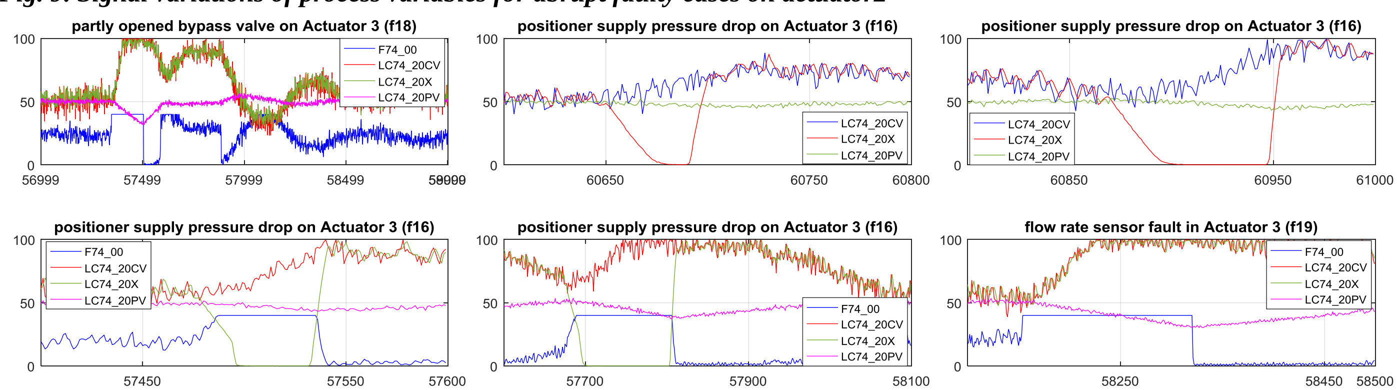


Fig. 10. Signal variations of process variables for abrupt faulty cases on actuator3

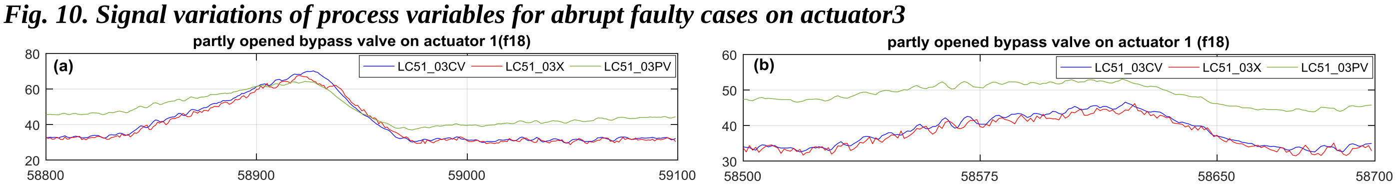


Fig. 11. Signal variations of process variables for incipient faulty cases on actuator1

Comparing the performance results of two series and parallel the FD frameworks presented in Table 2 and Table 5 as well as different types of classification schemes show that results obtained using these schemes are very similar to each other in some cases, whereas, in other cases some schemes outperform their counterparts. For instance, in regards to the proposed BL based classification scheme, parallel FD form represents better accuracy with a difference about 0.002 compared to the series detection form. Amongst the boosting classification schemes, irrespective of TN-CART, CART-OR, and CIT based series FD forms that mildly overtakes their corresponding schemes in parallel form, the other boosting scheme namely RF and C5.0 used in parallel form provides full accuracy values compared to their series counterparts. In the proposed single classifier based FD methods, series FD module where SVM, PLS, DDA, LDA, and PNN are used show better accuracies compared those exploited in the single classifier based parallel FD schemes, while, other single classifiers which were used in parallel FD module such as GP-COACH-FRBCS, WKNN, RSIMC, and NB were much more effective tools (represent near-full accuracies) to detect faults in contrast to those used in series form. On the other hand, it is observed that in both series and parallel forms, proposed BL based classification scheme outperforms almost other classification schemes in FD phase. However, there are cases of comparable results for fault detection, for example, those results provided by GP-COACH-FRBCS (single classifier), RF, and C5.0 (boosted ensemble classifiers) used in both series and parallel FD module.

Table 3 Accuracy results of proposed single classifier based FI method used in series framework.

Methods	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	Overall
SVM	1	1	1	1	1	1	1	0.50	0.999	1	1	1	0.999	1	1	1	1	1	1	0.974
GP-COACH-FRBCS	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1.000
WKNN	0.998	0.944	0.959	0.968	1	0.913	1	0.571	0.999	0.875	0.678	0.884	0.997	0.999	0.999	0.954	1	1	1	0.930
DDA	0.999	0.999	0.994	0.50	1	0.50	1	0.50	0.50	0.50	0.50	0.50			1	1	0.50	1	1	0.787
NB	0.999	1	1	1	1	1	1	1	1	1	1	1	1	0.991	1	1	1	1	1	1.000
LDA	1	0.902	0.986	0.997	1	0.914	1	0.928	0.999	1	1	0.999	0.999	1	0.999	1	1	1	1	0.986
RSIMCA	0.932	0.869	0.960	0.727	0.857	0.931	1	0.50	0.50	0.541	0.50	0.576	0.999	0.994	1	1	0.977	0.999	1	0.783
PLS	0.495	0.809	0.50	0.50	0.50	0.50	1	0.50	0.50	0.50	0.50	0.50	0.653	0.50	0.50	0.50	0.50	0.50	0.50	0.550
PNN	1	0.993	1	0.984	1	0.954	1	0.50	0.500	0.50	0.50	0.999	0.997	1	1	0.999	0.50	0.994	1	0.864

Table 4 Accuracy results of proposed multiple classifier based FI method used in series framework.

Methods	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	Overall
RF	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1
TN- CART	0.997	0.934	0.50	0.978	0.571	0.995	1	0.50	0.499	0.916	0.50	0.50	0.962	0.858	0.5	0.997	0.727	0.984	0.999	0.79
C.5.0	1	1	1	1	0.985	1	1	0.785	0.999	1	1	1	1	1	1	1	1	0.999	1	0.99
CART-OR	0.925	0.50	0.50	0.50	0.50	0.50	0.983	0.50	0.50	0.50	0.50	0.50	0.849	0.50	0.50	0.50	0.50	0.50	0.50	0.57
CIT	0.999	0.999	0.960	1	1	1	1	0.785	0.999	0.958	0.99	1	0.999	0.994	1	1	0.977	0.999	1	0.98
BL	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	1.00

Table 5 Accuracy results of proposed FDI methods used in parallel framework.

	Methods	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	Overall
Single Classifiers	SVM	0.97	1.0	0.97	0.99	0.91	1.0	0.95	1.0	0.50	0.50	0.50	0.50	0.50	0.99	1.0	0.50	1.0	1.0	1.0	1.0	0.84
	PLS	0.72	0.50	0.50	0.50	0.50	0.50	0.50	1.0	0.50	0.50	0.50	0.50	0.50	0.79	0.50	0.50	0.50	0.50	0.50	0.50	0.55
	PNN	0.91	0.96	0.92	0.50	0.50	0.50	0.50	1.0	0.50	0.50	0.50	0.50	0.50	0.99	0.72	0.50	0.50	0.50	0.50	0.50	0.62
	NB	0.97	0.95	0.98	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.99	0.98	1.0	1.0	1.0	1.0	1.0	0.99
	LDA	0.93	0.95	0.93	0.94	0.90	1.0	0.87	1.0	0.50	0.50	0.65	0.50	0.54	0.98	0.99	1.0	1.0	1.0	0.99	0.81	0.85
	GP-COACH-FRBCS	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**0.97**	**1.0**	**1.0**	**1.0**	**0.97**	**1.0**	**1.0**	**1.0**	**1.0**	1.0
	WKNN	0.97	1.0	0.96	0.97	0.87	1.0	0.84	1.0	0.71	0.79	0.95	0.58	0.82	0.99	0.99	0.97	0.98	1.0	0.92	0.96	0.91
	DDA	0.94	0.95	0.99	1.0	0.50	1.0	0.50	1.0	1.0	0.50	0.50	0.50	0.50	0.99	0.98	1.0	1.0	0.50	1.0	1.0	0.82
Ensemble Classifiers	C.5.0	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**0.93**	**1.0**	**1.0**	**0.92**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.92
	CART-OR	0.87	0.75	0.50	0.50	0.50	0.50	0.50	1.0	0.50	0.50	0.50	0.50	0.50	0.95	0.50	0.50	0.50	0.50	0.50	0.50	0.59
	CIT	0.98	0.99	0.97	0.84	0.91	0.93	0.95	1.0	0.50	0.50	1.0	0.83	0.96	1.0	0.98	0.94	0.92	0.84	0.99	0.99	0.90
	RSIMCA	0.84	0.83	0.87	0.88	0.66	0.77	0.82	1.0	0.71	0.64	0.65	0.53	0.61	0.94	0.85	0.89	0.86	0.84	0.68	0.51	0.80
	RF	**1.0**	**1.0**	**0.99**	**1.0**	**0.99**	**1.0**	**1.0**	**1.0**	**0.71**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.98
	TN-CART	0.93	0.88	0.68	0.50	0.50	0.50	0.80	1.0	0.50	0.50	1.0	0.50	0.50	1.0	0.99	0.50	0.96	0.94	1.0	0.50	0.73
	BL	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**0.93**	**1.0**	**1.0**	**0.90**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	0.99

FI task is also carried out using presented series and parallel forms where the same classification schemes as FD phase are employed as well. The performance results of FI for series form are included in Table 3 and Table 4. It is immediately observed that values 1.00 for all faulty cases (that indicate 100% efficiencies for isolation of all faults) are obtained by the proposed BL based series FI module. These results are comparable to those obtained by GP-COACH-FRBCS (single classifier) and RF (boosted ensemble classifiers) in series FI module. The more watchful analysis of the results for the series FI method shows that some classic classification schemes such as PLS (single classifier) and CART-OR (boosted ensemble classifiers) have the worst performances in isolation of most faulty scenarios. Contrariwise, in terms of the overall level of the efficiency of the proposed series FI module, among the single classifiers, SVM presents the best performance, whereas, the PLS exhibits the worst efficiency. Moreover, RF and CART-OR are the best and the worst performing boosting classifier schemes used in series FI framework with corresponding accuracy values 1.00 and 0.566, respectively.

The results of FI for parallel framework are also included in Table 5. It is visible that the proposed BL based classification scheme exploited in parallel FI form provides values 1.000 for all faulty cases excluding f4, f8, f11, and f13. Although these efficiency results are comparable to those obtained by GP-COACH-FRBCS single classification scheme, the BL based single classification scheme represents more full efficiency results (100% accuracy) over more classes of system's faulty operation particularly for incipient faulty scenarios f3 and f4. In addition, GP-COACH-FRBCS represents the best performance, whereas, the PLS exhibits the worst efficiency in single classifier based parallel FI form and RF and CART-OR are the best and the worst performing boosting classifier schemes used in this form with accuracy values 0.995 and 0.541, respectively. Table 6 also displays sensitivity and specificity results for of parallel FDI framework for all classes of system's operating conditions where one against all approach is employed to calculate these measures. With reference to Table 6, it is immediately seen that for most of faulty cases fi, almost all fi instances are truly classified as fi and non-fi instances are not classified as fi by using the BL algorithm. In other words, BL provides the best results in terms of sensitivity and specificity indices for all classes of system behavior compared to other presented methods. However, there are a few cases reported in Table 6 where sensitivity and specificity of BL are a little bit lesser than 1.000 (such as f10 and f13) that can be due to the fact that some classes of faults are closely arranged in the feature space or even slightly overlap each other.

Table 6 Sensitivity and Specificity results of proposed FDI methods used in parallel framework for different faulty cases through Blended learning.

Method	Index	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	AVG
RF	Sens.	0.998	1.000	0.990	1.000	0.976	1.000	1.000	1.000	0.429	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.970
	Spec.	0.998	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
SVM	Sens.	0.976	1.000	0.948	0.971	0.829	1.000	0.893	1.000	0.000	0.000	0.000	0.000	0.000	1.000	0.994	0.000	1.000	1.000	1.000	1.000	0.681
	Spec.	0.970	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.988	0.998	1.000	1.000	1.000	1.000	0.999	0.998
TN-CART	Sens.	0.930	0.866	0.661	0.418	0.440	0.449	0.776	1.000	0.486	0.000	0.992	0.000	0.000	0.993	0.990	0.000	0.950	0.927	0.991	0.000	0.593
	Spec.	0.934	0.885	0.696	0.554	0.550	0.550	0.830	1.000	0.500	1.000	0.999	1.000	1.000	0.999	0.991	1.000	0.966	0.940	0.999	1.000	0.870
C.50	Sens.	0.998	1.000	1.000	1.000	0.999	1.000	1.000	1.000	0.901	1.000	1.000	0.909	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.990
	Spec.	0.999	1.000	1.000	1.000	1.000	0.999	1.000	1.000	0.932	0.996	0.998	0.917	1.000	0.999	1.000	1.000	1.000	1.000	1.000	0.999	0.992
CART-OR	Sens.	0.851	0.738	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.908	0.000	0.000	0.000	0.000	0.000	0.000	0.175
	Spec.	0.881	0.772	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.976	1.000	1.000	1.000	1.000	1.000	1.000	0.981
GP-COACH-FRBCS	Sens.	0.995	0.997	0.999	0.998	0.999	0.999	0.999	0.999	1.000	1.000	1.000	0.958	1.000	1.000	1.000	0.971	1.000	0.999	1.000	1.000	0.996
	Spec.	0.997	1.000	1.000	1.000	1.000	1.000	1.000	0.999	1.000	1.000	1.000	0.981	1.000	1.000	1.000	0.973	1.000	1.000	1.000	1.000	0.997
WKNN	Sens.	0.981	0.998	0.943	0.964	0.856	1.000	0.793	1.000	0.699	0.783	0.942	0.395	0.817	0.999	0.990	0.970	0.970	1.000	0.932	0.942	0.899
	Spec.	0.965	0.999	0.973	0.977	0.875	1.000	0.885	1.000	0.730	0.787	0.957	0.771	0.824	0.986	0.985	0.974	0.989	1.000	0.912	0.970	0.928
LDA	Sens.	0.957	1.000	0.991	0.998	0.000	0.996	0.000	1.000	1.000	0.000	0.000	0.000	0.000	0.979	0.979	1.000	0.999	0.000	1.000	1.000	0.645
	Spec.	0.932	0.902	0.987	0.998	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.995	0.980	1.000	1.000	1.000	1.000	1.000	0.990
NB	Sens.	0.974	0.950	0.981	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.978	0.971	1.000	1.000	1.000	1.000	1.000	0.993
	Spec.	0.959	0.943	0.988	0.999	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.998	0.988	1.000	0.999	1.000	1.000	1.000	0.994
DDA	Sens.	0.931	0.951	0.928	0.940	0.890	0.999	0.859	1.000	0.000	0.000	0.127	0.000	0.118	0.968	0.976	0.999	0.999	0.997	0.970	0.775	0.722
	Spec.	0.921	0.950	0.936	0.942	0.912	1.000	0.891	1.000	1.000	1.000	1.000	1.000	0.953	0.990	1.000	0.999	0.999	1.000	1.000	0.837	0.967
CIT	Sens.	0.986	0.976	0.956	0.819	0.912	0.928	0.945	1.000	0.000	0.000	0.999	0.828	0.934	0.993	0.988	0.931	0.901	0.793	0.981	0.990	0.843
	Spec.	0.974	0.998	0.980	0.855	0.915	0.936	0.946	1.000	1.000	1.000	1.000	0.838	0.995	0.998	0.978	0.957	0.938	0.894	0.997	0.993	0.960
RSIMCA	Sens.	0.852	0.850	0.849	0.892	0.398	0.687	0.808	1.000	0.704	0.560	0.531	0.333	0.403	0.929	0.809	0.889	0.862	0.837	0.182	0.119	0.675
	Spec.	0.834	0.811	0.900	0.873	0.912	0.853	0.822	1.000	0.724	0.726	0.769	0.722	0.811	0.952	0.886	0.888	0.858	0.850	1.000	0.897	0.855
PLS	Sens.	0.718	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.751	0.000	0.000	0.000	0.000	0.000	0.000	0.123
	Spec.	0.740	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.701	1.000	1.000	1.000	1.000	1.000	1.000	0.972
NN	Sens.	0.913	0.956	0.918	0.000	0.000	0.000	0.000	0.999	0.000	0.000	0.000	0.000	0.000	0.974	0.665	0.000	0.000	0.000	0.000	0.000	0.271
	Spec.	0.900	0.961	0.931	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.991	0.768	1.000	1.000	1.000	1.000	1.000	0.978
BL	Sens.	1.000	1.000	1.000	1.000	0.999	1.000	1.000	1.000	0.892	1.000	1.000	0.818	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.985
	Spec.	0.994	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.976	1.000	1.000	0.990	1.000	0.999	1.000	1.000	1.000	1.000	1.000	1.000	0.998

It is also worth highlighting that false alarms are undesirable in FDI systems because they tend to cause the users of the system to lose faith in it. Therefore, an analysis of the ability of the FDI method to test against false alarms is very essential. False alarms represent the time fraction of the benchmark zone during which the FD algorithm has wrongly detected the fault: the fault is not acting on the plant and the FD algorithm is signaling a false alarm. Table 7 represents the false alarm rates for different faulty scenarios given by BL algorithm in FD module. As seen, in most of faulty scenarios the best value false alarm rate is 0 that means that no false alarms at all are raised at any time when the FD algorithm is on. It is also worth observing that although in some cases false alarm rate is not 0, it is still retained almost near 0 which can be acceptable due to the presence of uncontrolled effects such as noise and disturbance acting on industrial actuator system.

Table 7. False alarm analysis for different faulty cases through Blended learning.

Fault	False Alarm	Fault	False Alarm
f1	0.000	f10	0.047
f2	0.027	f11	0.000
f3	0.019	f12	0.000
f4	0.000	f13	0.000
f5	0.000	f14	0.000
f6	0.000	f15	0.000
f7	0.000	f16	0.022
f8	0.000	f17	0.000
f9	0.000	f18	0.057
		f19	0.000

Remarks on Robustness: It is also worth noting that the proposed FDI schemes based on blended leaning show high degrees of robustness in that they are sensitive to faulty conditions (*i.e.,* high sensitivity rates), while, producing very low false alarm rates (*i.e.,* producing high specificity rate and being highly sensitive to normal operating condition). Moreover, since the proposed FDI schemes are developed and tested based on the datasets collected from a high-fidelity simulator of process actuator system which was validated against real measurements and could yield robust and highly accurate results, it can be deduced that the proposed methods can show high performance on the real measurements, as well. It is also worth observing that the suggested FDI algorithm is also immune against uncontrolled effects such as noise and disturbance provided that it is assumed that all actuator block inputs and outputs are disturbed by band-limited white noise superimposed with a 50Hz sine wave simulating measurement noise and electro-magnetic susceptibility of physical sensors. The amplitude of the sine bias of artificial noise was arbitrarily set to 2.5% of disturbed signal nominal range. The maximum amplitude of pseudo-random noise was limited also to 2.5% of signal nominal range. This facilitates the simulation of a relatively noisy environment which is particularly suitable for the investigation of robustness features of the FDI algorithms. [9].

4.2. Comparative study

The results of proposed model-free FDI obtained in the present research are comparable to the proposed advanced model-free methods described in the literature [19], [20], and [10]. These works are considered as the leading studies where distinguished results were obtained on the FDI of the DAMADICS benchmark. Thus, in this section, a brief comparison has been made between the results obtained in the present study and those achieved by aforementioned works. Table 8 provides detailed information in regards to the FDI results of three conducted studies on this benchmark against that one obtained in the present research. As seen, it is only the present study that reports and investigations and results on all potential set of faulty scenarios of the benchmark problem. Moreover, it is worth observing that all of the considered works ignored considering fault cases f3, f4, f5, f6, and f9. However, with reference to Table 1, it is observed that these neglected are of incipient fault-type and their detection and isolation represents the greatest concern in an FDI problem because they gradually develop over a long period of time [1]. Furthermore, among these ignored incipient faults, faulty scenarios f3, f5, f6, and f9 are hardly detectable compared to their ramp-shaped faulty counterparts due to their very slow growth in time. The main superiority of the present study over those presented in the literature is taking all possible faulty scenarios including incipient faulty cases of this benchmark into account and the achieved FDI results on these neglected cases are also high. For a fair comparative study, only results on the considered faulty scenarios presented by the literature are compared. Regarding FD task, it is seen that only the results provided by [19] is comparable with the proposed FDI methods in this study. However, the presented series FDI strategy outperforms the whole FD schemes as it produced the minimum false alarm rate, whereas, representing the maximum true detection rate. Moreover, it is highlighted that the FD method presented in [20] is gently sensitive to the normal operating condition and the proposed FD method in [10] is also mildly sensitive to faulty operating conditions. FI was not totally conducted in [19], and [20] only investigated the isolation task in regards to faulty scenarios f10, f11, f13, and f19.

However, according to FI results, it is evidently seen that proposed parallel isolation strategy in the present study provides the highest accuracy results compared to those achieved by other schemes. In addition, it is noted that the proposed series FI scheme also outperforms its counterparts in other works in all considered faulty cases expect f8 and f11 where it shows mildly lesser accuracy results. The worst performing classification scheme for fault classification is also presented by [20] in which even weak accuracy results was reported for abrupt-type faulty cases f10 and f19. As a result both proposed FDI frameworks in this research can be nominated as the best performing methods. However, if one method has to be solitarily chosen, the proposed series BL classification scheme can be preferred as the leading FDI performer. Furthermore, it is also noted that the proposed series FDI method shows high level of robustness in the sense that it is highly sensitive to faulty conditions while keeping the false alarm rates in the lowest level.

Table 8. Comparative analysis of model-free FDI methods on DAMADICS benchmark

		[19]	**[20]**	**[10]**	**Proposed FDI Methods**	
					Series	Parallels
Fault Detection	True Detection	0.938	0.918	0.788	**0. 997**	**1.000**
	False Alarm	0.000	0.220	0.048	**0.002**	**0.057**
Fault Isolation	f1	-	-	1.0	**1.0**	1.0
	f2	-	-	1.0	**1.0**	1.0
	f7	-	-	1.0	**1.0**	1.0
	f8	-	-	0.972	**1.0**	**0.933**
	f10	-	**0.932**	1.0	**1.0**	1.0
	f11	-	**0.878**	1.0	**1.0**	**0.900**
	f12	-	-	1.0	**1.0**	1.0
	f13	-	**0.950**	1.0	**1.0**	1.0
	f14	-	-	0.086	**1.0**	1.0
	f15	-	-	1.0	**1.0**	1.0
	f16	-	-	0.708	**1.0**	1.0
	f17	-	-	1.0	**1.0**	1.0
	f18	-	-	0.668	**1.0**	1.0
	f19	-	1.0	1.0	**1.0**	1.0

5. Conclusions

The paper proposes a new multiple classifier-based FDI system based on a blended learning algorithm used in model-free series and parallel configurations. That is, the application of proposed model-free FDI schemes on the DAMADICS process control system benchmark was presented. The main purpose of this study was to compare two proposed non-model-based series and parallel FDI system on the basis of single and ensemble classification strategies and to present effective and relatively computationally- efficient FDI systems which can be practically exploited as decision schemes in off-line as well as on-line process diagnostic systems. Moreover, a powerful classification scheme on the basis of ensembling approach and blended learning was presented that outperformed the conventional and well-practiced classification methods. The employed classification schemes were implemented in R standard platform which is a distinguished software package for machine learning.

The proposed FDI strategies have been tested on the DAMADICS benchmark problem. This benchmark is concerned with intensely nonlinear process characteristics thereby 19 artificial faulty cases are considered. However, the obtained results on single abrupt faulty scenarios demonstrate efficiency in both detection and isolation phases. This is strongly due to the proposed blended learning based ensembling of classifiers. On the other hand, it is seen that the proposed FDI system is strongly capable of dealing with incipient faulty scenarios. The incipient faults have been simulated in the control loop according to the DAMADICS benchmark rule which imposes a very low development speed. However, due to feedback control action in steady-state conditions, even if these faults take place as gradual development, their effects on output measurements cannot be observed in a ramp mode. In other cases the fault effects in the system behavior are similar to the noise effects making the faults undistinguishable from the normal system operation. In the present study, confusion matrix is used in order to evaluate the proposed fault diagnosis systems that were created applying different classification schemes. However, this measure can be directly compared with false and true detection/isolation rates proposed in the DAMADICS simulator documentations. The results of FDI using the proposed blended learning based series and parallel strategies that were achieved in this study outperform advanced methods described in the literature. Moreover, all possible faulty cases were taken into consideration, while, in the related literature only selected cases that mostly were picked from abrupt scenarios were taken into consideration.

The proposed FDI frameworks could also be extended to present and integrated computational intelligence-based diagnostic and prognostic frameworks for prediction of the remaining useful life time of the system/components when an incipient fault occurred in the system, Moreover, the proposed FDI scheme can simply deal with the combined simultaneous faults by considering each possible combination of single faults as an independent class of system's behavior, as well.

References

[1]Nozari H. A., Simani S., Shoorehdeli M. A., (2012). Model-based robust fault detection and isolation of an industrial gas turbine prototype using soft computing techniques. Neurocomputing, (Vol.91, pp. 29–47).

[2]Odgaard P. F., (2014). Fault-Tolerant Control of Wind Turbines: A Benchmark Model, IEEE Transactions on Control Systems Technology, (Vol.21 (4), pp. 1168 – 1182).

[3]Castaldi P., Mimmoa N., Simani S., (2014). Differential geometry based active fault tolerant control for aircraft, Control Engineering Practice, (Vol 32, pp.227–235,).

[4]Simani S., Fantuzzi C., (2006). Dynamic system identification and model-based fault diagnosis of an industrial gas turbine prototype, Mechatronics (Vol.16 (6), pp.341–363).

[5]Guglielmi, G., Parisini, T., & Rossi, G. (1995). Fault diagnosis and neural networks: a power plant application, Keynote Paper. Control Engineering Practice, (Vol.3, pp.601–620).

[6]Kuncheva L., (2004). Combining Pattern Classifier: Methods and Algorithms. New Jersey: Wiley-Interscience.

[7]Rokach L (2010). Ensemble-based classifiers. Artificial Intelligence Review. (Vol. 33 pp.1–39).

[8]DAMADICS (2004), Website of the Research Training Network on Development and Application of Methods for Actuator

Diagnosis in Industrial Control Systems http://diag.mchtr.pw.edu.pl/damadics.
[9]Bartys, M., R. Patton, M. Syfert, S. de las Heras, and J. Quevedo (2006). Introduction to the DAMADICS actuator FDI benchmark study. Control Engineering Practice, (Vol. 14(6), 577-596).
[10]Kalisch M., Przystałka P., Timofiejczuk A., (2015). Actuator Fault Diagnosis Using Single and Meta-Classification Strategies. In: Mercier-Laurent E., Owoc M.L., Boulanger D. (eds) Artificial Intelligence for Knowledge Management. IFIP Advances in Information and Communication Technology, (Vol. 469 Springer, Cham).
[11]Isermann R., (1998). On Fuzzy Logic Applications for Automatic Control, Supervision, And Fault Diagnosis, IEEE Transactions in Systems, (Vol.28, pp.221-235).
[12]Witczak M., (2006). Advances in model-based fault diagnosis with evolutionary algorithms and neural networks International Journal of Applied Mathematics and Computer Science, (Vol.16 (1), pp.85-99).
[13]Puig, V., Stancu, A., Escobet, T., Nejjari, F., Quevedo, J., Patton, R.J. (2006). Passive robust fault detection using interval observers: Application to the DAMADICS benchmark problem. Control Engineering Practice, (Vol.14 pp.621–633).
[14]Stancu, A., V. Puig, J. Quevedo and RJ. Patton, (2003). Passive robust fault detection using nonlinear interval observers: Application to the DAMADICS Benchmark Problem, in proceedings of 5-th IFAC Symposium SAFEPROCESS, Washington, USA.
[15]L. F. Mendonça, J. M. C. Sousa, J. M. G. Sá da Costa, (2009). An architecture for fault detection and isolation based on fuzzy methods, Expert Systems with Applications, (Vol. 36 (2), pp. 1092-1104).
[16]Kourd Y, Lefebvre D., Guersi N., (2013). fault diagnosis based on neural networks and decision trees: application to damadics, International Journal of Innovative Computing, (Vol. 9, (8), pp. 1–12).
[17]Sundarmahesh. R and Kannapiran. B., (2013). Fault Diagnosis of Pneumatic Valve with DAMADICS Simulator using ANN based Classifier Approach. IJCA Proceedings on International Conference on Innovations in Intelligent Instrumentation, Optimization and Electrical Sciences ICIIIOES, (Vol.9, pp.18-17).
[18]Andrzej Katunin , Marcin Amarowicz , Paweł Chrzanow (2015). Faults diagnosis using self-organizing maps: A case study on the DAMADICS Benchmark problem. IEEE Conference on Computer Science and Information Systems FedCSIS, (Vol. 5, pp. 1637-1681).
[19]J. Calado, J. Sá da Costa, M. Barty´s, and J. Korbicz, (2006). FDI approach to the damadics benchmark problem based on qualitative reasoning coupled with fuzzy neural networks, Control Engineering Practice, (Vol. 14, pp. 685–698).
[20]Previdi F. and Parisini T., (2006). Model-free actuator fault detection using a spectral estimation approach: the case of the damadics benchmark problem, Control Engineering Practice, (Vol. 14, pp. 635–644).
[21]Oliveira A. R. C., Sá da Costa J. M. G., Hierarchic Fault Diagnosis by Pattern-Recognition Approaches Applied to DAMADICS Benchmark, 18th IFAC World Congress Milano , Italy , August 28 - September 2, 2011
[22]Chopra T and Vajpai J, fault diagnosis in benchmark process control system using stochastic gradient boosted decision trees, international journal of soft computing and bioinformatics, Volume 2, Number 1, January-June 2011, pp. 37-41
[23]Y. Freund and R. E. Schapire (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences, (Vol.55(1), pp. 119–139).
[24]Roe, B. P.; Yang, H.-J.; Zhu, J. (2006). Boosted Decision Trees, A Powerful Event Classifier, and Statistical Problems in Particle Physics, Astrophysics and Cosmology: proceedings of PHYSTAT05, London, UK, (p.139).
[25]Freund Y. and. Schapire R E, (1999). Large margin classication using the perceptron algorithm. Machine Learning, (Vol.37, (3) pp.277-296).
[26]M. Kuhn and K. Johnson (2013). Applied predictive modeling. Springer.
[27]Pandya R, Pandya J, (2015). C5.0 Algorithm to Improved Decision Tree with Feature Selection and Reduced Error Pruning, International Journal of Computer Applications, (Vol. 117, (16)).
[28]S. Torsten Hothorn, Kurt Hornik, and Achim Zeileis. (2006). unbiased recursive partitioning: A conditional inference framework. Journal of Computational and Graphical Statistics, (Vol.15, (3), pp.651-674).
[29]Sardá-Espinosaa A, Subbiaha S, Bartz-Beielsteinb T., (2017), Conditional inference trees for knowledge extraction from motor health condition data, Engineering Applications of Artificial Intelligence, (Vol. 62, pp. 26–37).
[30]Pitman E. J. G. (1937), Significance Tests Which May be Applied to Samples from any Populations. Supplement to the Journal of the Royal Statistical Society, (Vol. 4, (1), pp. 119-130).
[31]Breiman, L., J. H. Friedman, R. A. Olshen, and C. G. Stone, (1984). Classification and Regression Trees. Wadsworth International Group, Belmont, California, USA.
[32]Therneau TM, Atkinson B. RPART (2008). Recursive partitioning. R port by B. Ripley. R package version, (Vol.3, pp.1-41).
[33]Galimberti G., Soffritti G, Di Maso M. (2012). Classification Trees for Ordinal Responses in R: The rpartScore Package. Journal of statistical software, (Vol. 47(10), pp.1-25).
[34]Breiman L (2001). Random forests. Mach Learn, (Vol.45, pp. 5–32).
[35]Wolpert, D. H, (1992). Stacked generalization. Neural Networks (Vol.5, (2), pp.241–260).
[36]Yin-Wen Chang, Cho-Jui Hsieh, Kai-Wei Chang, (2010). Training and Testing Low-degree Polynomial Data Mappings via Linear SVM, Journal of Machine Learning Research, (Vol. 11, pp.1471-1490).
[37]Gou J, Du L, Zhang Y, Xiong T (2012). A New Distance-weighted k-nearest Neighbor Classier. Journal of Information & Computational Science, (Vol. 9 (6), pp. 1429–1436).
[38]Dudoit, S., J. Fridlyand, and T. P. Speed, (2002) Comparison of Discimination Methods for the Classification of Tumors Using

Gene Expression Data, Journal of American Statistical Association, (pp. 77–87).

[39]Silva A. (2011), Two-group classification with high-dimensional correlated data: A factor model approach. Computational Statistics & Data Analysis, (Vol. 55 (11), pp. 2975–2990).

[40]Ahdesm, A., and K. Strimmer, (2010). Feature selection in omics prediction problems using cat scores and false non-discovery rate control. Ann. Appl. Stat. (Vol.4, pp.503-519).

[41]Zuber, V., and K. Strimmer. (2009). Gene ranking and biomarker discovery under correlation. Bioinformatics, (Vol.25, pp. 2700-2707).

[42]T. M. Mitchell, Machine Learning, McGraw-Hill.

[43]C. M. Bishop, Pattern Recognition and Machine Learning, Springer

[44]Magdalena L. (2015). Fuzzy Rule-Based Systems Chapter in Springer Handbook of Computational Intelligence. Springer-Verlag Berlin Heidelberg. (pp. 203-218).

[45]Ishibuchi H., Nakashima T. and Nii M., (2004) Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining, Springer Verlag.

[46]Berlanga F. J., (2008). del Jesus M J and Herrera F, A Novel Genetic Cooperative-Competitive Fuzzy Rule Based Learning Method using Genetic Programming for High Dimensional Problems, In Proc. Of 3rd International Workshop on Genetic and Evolving Fuzzy Systems Witten-Bommerholz, (pp. 101-106).

[47]Mehmood T, Ahmed B. (2016). The diversity in the applications of partial least squares: an overview. Journal of Chemometrics, (Vol. 30 (1), pp. 4–17).

[48]Wold S, and Sjostrom, M (1977). SIMCA: A method for analyzing chemical data in terms of similarity and analogy. Chapter in Chemometrics: Theory and Application, ACS Symposium Series, (Vol. 52, pp 243–282).

[49]O. Nelles, (2001). Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models, Springer Press.

Appendix A.: Simulation parameters for the employed classification schemes

Classifier	Parameters in Parallel FDI Form	Parameters in Series Fault detection Form	Parameters in Series Fault Isolation Form
SVM	Kernel mode= radial, kpar = "automatic", C = 1, nu = 0.2, epsilon = 0.1	Kernel mode= radial, kpar = "automatic", C = 1.3, nu = 0.2, epsilon = 0.12	Kernel mode= radial, kpar = "automatic", C = 1.41 nu = 0.2, epsilon = 0.12
RF	mtry= sqrt(p), ntree: 150, importanceSD=p/nclass, cutoff=1/nclass	mtry= sqrt(p), ntree: 100, importanceSD=p/nclass cutoff=1/nclass	mtry= sqrt(p), ntree: 170, importanceSD=p/nclass cutoff=1/nclass
TN- CART	Automatically find best cp through cross validation method	Automatically find best cp through cross validation method	Automatically find best cp through cross validation method
C.50	metric= usage, CF=0.6	metric= usage, CF=0.5	metric= usage, CF=0.6
CART-OR	Split=abs, Prune= mc	Split=quad, Prune=mr	Split=abs, Prune=mc
G-COACH-FRBCS	max.iter= 200, max.num.rule=100, persen_cross= 0.6, persen_mutant=0.5	max.iter= 200, max.num.rule=200, persen_cross=0.55, persen_mutant=0.7	max.iter= 200, max.num.rule=120, persen_cross=0.55, persen_mutant=0.5
W KNN	Kernel= biweight, K=6	Kernel= triweight, K=23	Kernel= biweight, K=8
LDA and DDA	num.iterations= 200, alpha= 0.6, lambda=0.3, diagonal=FALSE	num.iterations= 300, alpha=0.5, lambda=0.35, diagonal=FALSE	num.iterations= 200, alpha=0.6, lambda=0.39, diagonal=FALSE
	num.iterations= 200, alpha=0.5, lambda=0.41, diagonal=TRUE	num.iterations= 300, alpha=0.4, lambda=0.3, diagonal=TRUE	num.iterations= 200, alpha=0.5, lambda=0.41, diagonal=TRUE
NB	Laplace=0.01, Kernel=density	Laplace=0.015, Kernel= density	Laplace=0.01, Kernel= density
CIT	Xtrafoa= ptrafo, Ytrafoa= ptrafo, Weights= vectors 2, Ntree=300, Minsplit: 20, Mincriterion=0.25	Xtrafoa= ptrafo, Ytrafoa= ptrafo, Weights= vectors 3, Ntree=500, Minsplit:10, Mincriterion=0.95	Xtrafoa= ptrafo, Ytrafoa= ptraf, Weights= vectors 2 Ntree=350, Minsplit:15, Mincriterion=3.5
R- SIMCA	Grouping=fault number, Alpha=0.55, Kmax=20	Grouping= fault number, Alpha=0.5, Kmax=16	Grouping= fault number, Alpha=0.57, Kmax=17
PLS	ncomp = 9, lower=0.4	Ncomp= 5, Lower=0.5	Ncomp=8, Lower=0.5
PNN	Alpha= 0.3, Epoch= 200, Nlayer= 3	Alpha= 0.5, Epoch= 100, Nlayer= 4	Alpha= 0.3, Epoch= 150, Nlayer= 3

Appendix B.: Descriptions of parameters used in the structures of classifiers

SVM:

- kpar: the list of hyper-parameters (kernel parameters). This is a list, which contains the parameters to be used with the kernel function.
- C: the cost regularization parameter. This parameter controls the smoothness of the fitted function, essentially higher values for C lead to less smooth functions.
- Epsilon: in the insensitive-loss function.
-

RF:

- mtry: Number of variables randomly sampled as candidates at each split. Note that the value for classification is chosen sqrt(p) where p is number of variables.
- ntree: number of trees grown.
- importanceSD: The “standard errors” of the permutation-based importance measure. For classification, a p (number of variables) by (number of classes)nclass + 1matrix corresponding to the first nclass + 1 columns of the importance matrix.
- Cutoff= (Classification only) A vector of length equal to number of classes. The ‘win-ning’ class for an observation is the one with the maximum ratio of proportion of votes to cutoff. It is set 1/nclass where nclass is the number of classes (i.e., majority vote wins).
- Cart: Tuned parameter is CP or complexity parameter. We establish range of numbers of 0.001 to 1 to find best match through cross validation.

C5.0 tree:

By setting "usage" as metric parameter, C5.0 measures predictor importance by determining the percentage of training set samples that fall into all the terminal nodes after the split.

- CF: confidence factor, which should be between 0 to 1.

CART-OR:

The argument split controls the splitting function used to grow the classification tree, by setting the misclassification costs in the generalized Gini impurity function equal to the absolute ("abs") or to the squared ("quad") differences in scores.

The argument prune allows the user to select the prediction performance measure used to prune the classification tree, and can take two values: "mr" (total misclassification rate) or "mc" (total misclassification cost).

G-COACH- FRBCS:

- max.iter: maximum number of iteration.
- max.num.rule: the maximum number of rules.
- persen_cross: determining the probability of crossover.
- persen_mutant: determining the probability of mutation.

WKNN:

- Kernel= kernel functions to weight the neighbors according to their distances
- K= number of the nearest neighbors

LDA and DDA:

- num.iterations: number of iterations
- Alpha: learning rate
- Lambda: Shrinkage intensity for the correlation matrix
- Diagonal: Chooses between LDA (diagonal=FALSE) and DDA (diagonal=TRUE).

NB:

- Laplace: value used for Laplace smoothing.
- Kernel: to estimate the densities of metric predictors.

CIT:

- Weights: a vector weights to be used in the fitting process.
- Xtrafoa: function to be applied to all input variables.
- Ytrafoa: function to be applied to all response variables.
- Minsplit: the minimum sum of weights in a node in order to be considered for splitting.
- Ntree: number of trees to grow
- Mincriterion: depths of trees

R- SIMCA:

- Grouping: (grouping variable) a factor specifying the class for each observation or vector of classes.
- Alpha: this parameter measures the fraction of outliers the algorithm should resist. Alpha controls the size of the subsets over which the determinant is minimized.
- Kmax: maximal number of principal components to compute.

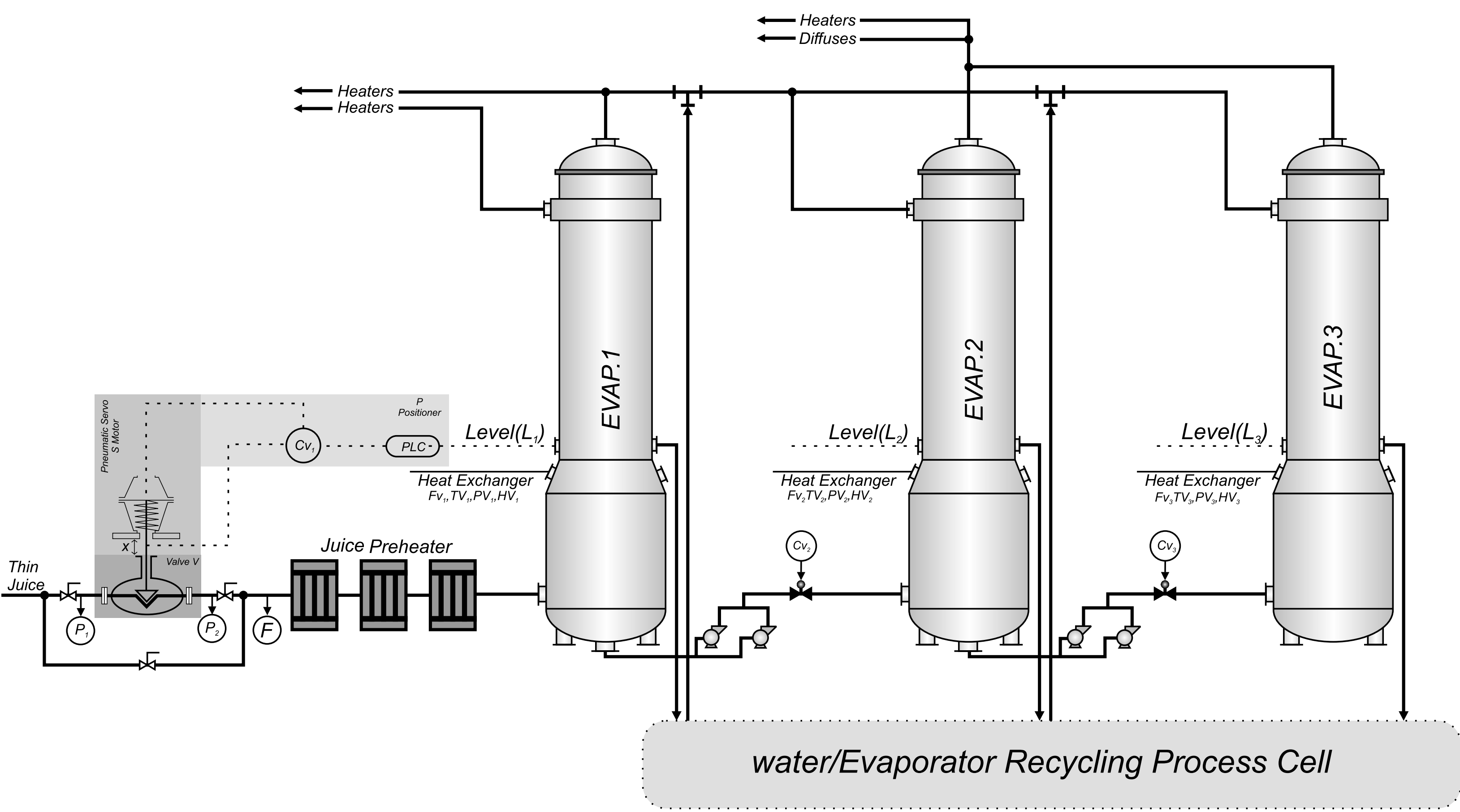
PLS:

- ncomp = number of component or projections
- Lower: lower is limit for power optimization

Multilayer Neural Network:

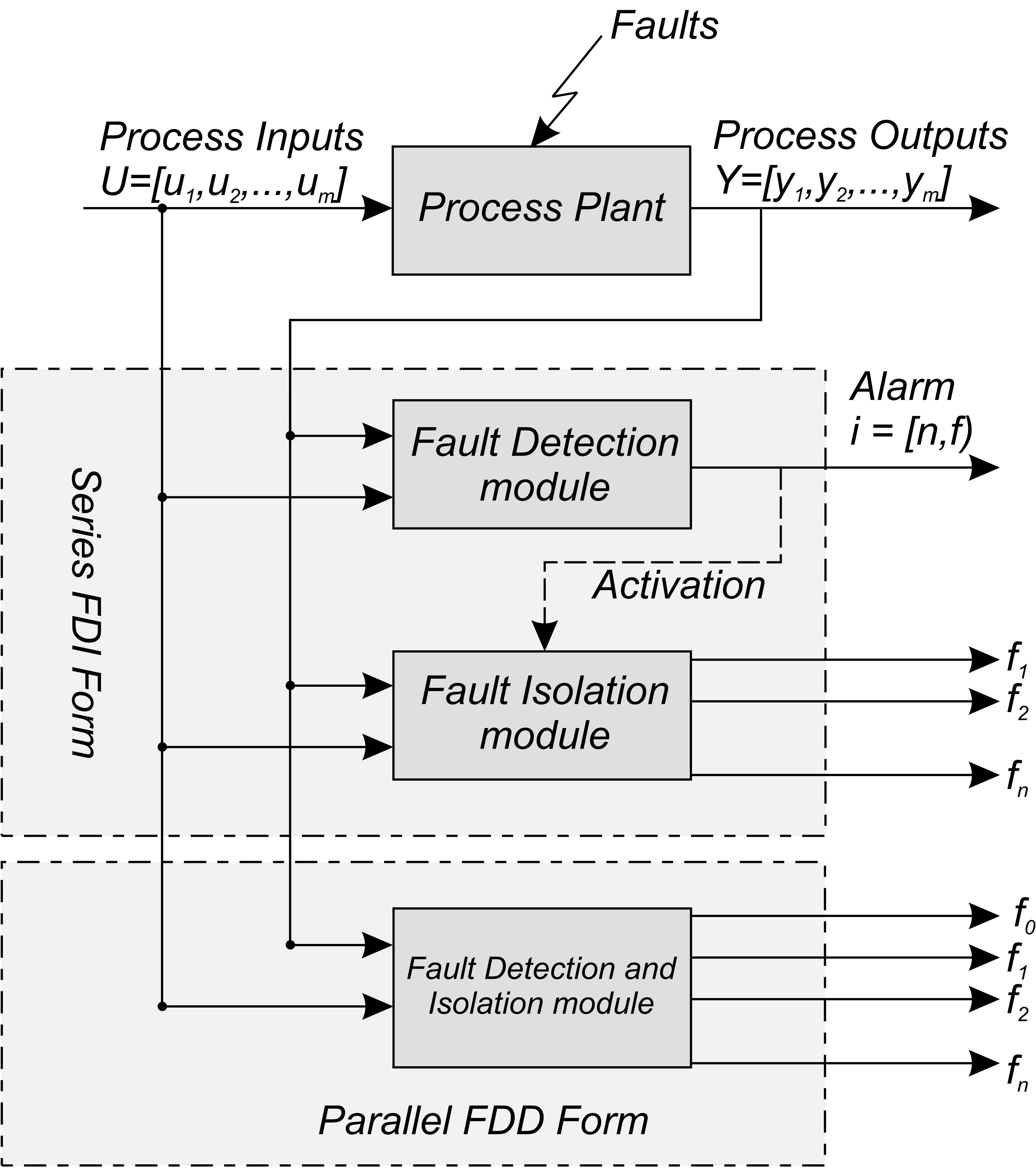
- Alpha= Learning rate
- Epoch= number of epochs
- Nlayer= number of hidden layers

Figure



Heaters
Diffuses
Heaters
Heaters
EVAP.1
EVAP.2
EVAP.3
Pneumatic Servo S Motor
P Positioner
Cv1
PLC
Level(L1)
Level(L2)
Level(L3)
Heat Exchanger
Fv1,TV1,PV1,HV1
Fv2TV2,PV2,HV2
Fv3TV3,PV3,HV3
Juice Preheater
Thin Juice
Valve V
x
P1
P2
F
Cv2
Cv3
water/Evaporator Recycling Process Cell

Figure



Faults
Process Inputs
U=[u1,u2,...,um]
Process Plant
Process Outputs
Y=[y1,y2,...,ym]
Alarm
i = [n,f)
Fault Detection module
Activation
Fault Isolation module
f1
f2
fn
Series FDI Form
Fault Detection and Isolation module
f0
f1
f2
fn
Parallel FDD Form

Figure

Algorithm1. (AdaBoost)

C: Base Classifier/Learner

N: Number of Iterations

D: Data Set $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$

$D_1(\mathrm{i})=1/\mathrm{n}$

For j=1 to N

Construct classifier Ci with distribution Di

Calculate: $\varepsilon = \sum_{j:C \neq yi} D_j(i)$

If (ε>0.5) Then

Break

End If

Calculate: $\alpha_j = \frac{1}{2} \ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$

Update: $D_{j+1}(i) = D_j(i), \exp\left(-\alpha_j y_i C_j(x_i)\right)$

Normalize $D_{j+1}(i)$ so that it will be a distribution.

End For

Figure

Algorithm2. (Bagging)

D: Data Set $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$

N: Number of Iterations

S: Subset Size

D_1(i)=1/n

For i=1 to N

D_i=choose S samples from D with replacement

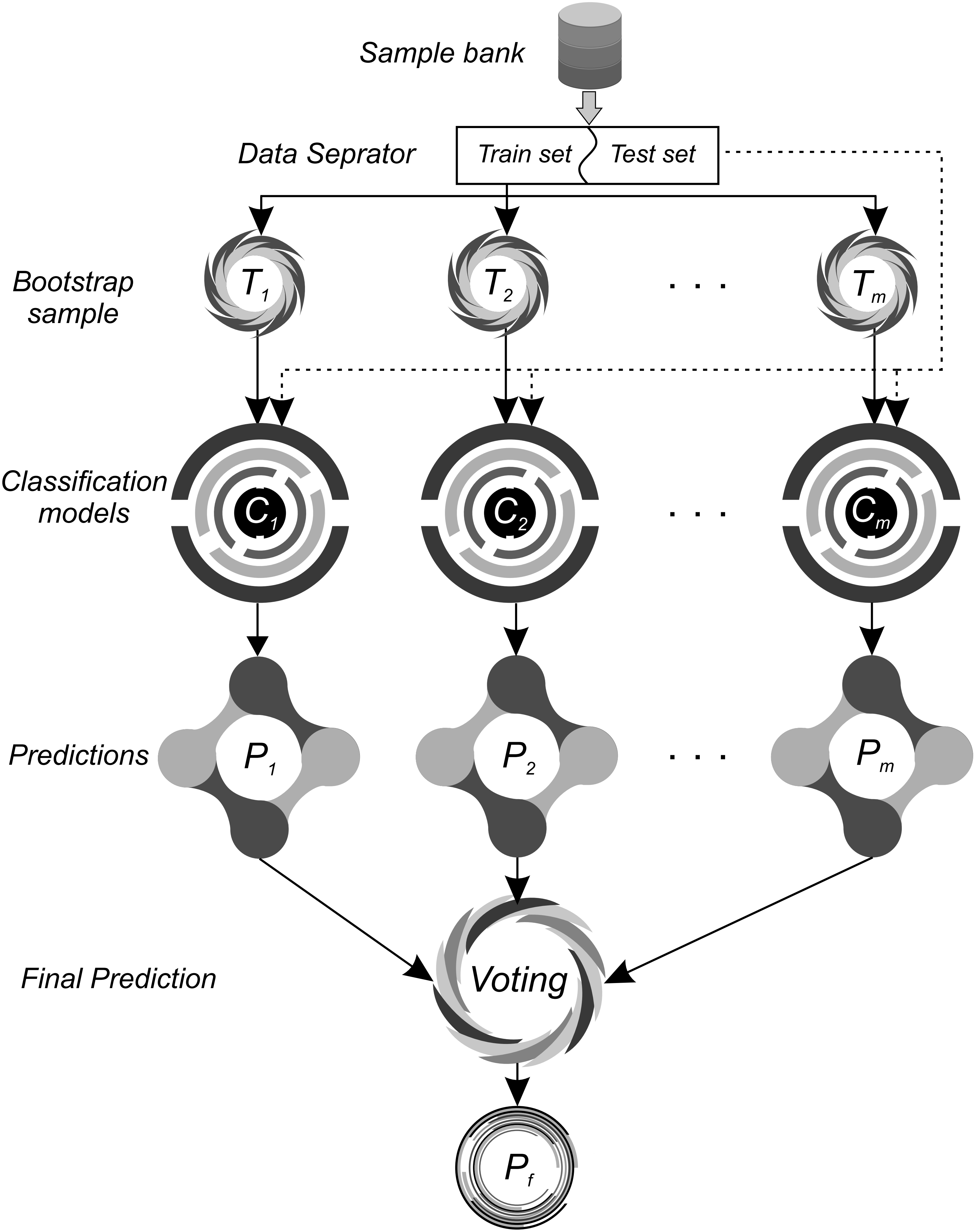
Construct classifier C using D_i

End For

Majority voting to calculate the total prediction:

$$pt = \arg\left(\max_{y \in Y} \sum_{1:pt(x)=y} (1)\right)$$

Figure


Sample bank
Data Seprator
Train set
Test set
Bootstrap sample
T1
T2
Tm
. . .
Classification models
C1
C2
Cm
Predictions
P1
P2
Pm
Final Prediction
Voting
Pf

Figure

Algorithm3. (Random Forest)

DTC: Base decision tree classifier

N: Number of iterations

D:Data set $\{(x_1,x_1),(x_2,x_2),\ldots,(x_n,x_n)\}$

S: Subset size

M: Attribute number in each node

For i=1 to N

D_i=choose S samples from D with replacement

Construct classifier C using *DTC (M)* on D

End For

Figure

Algorithm4. (Blended Learning)

D: Data Set $\{(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)\}$

C: Individual learner / base classifier

Ct: Meta classifier/voting scheme

First-level learning algorithms $LA_1, \ldots, LA_m$

Second-level learning algorithm LA

First-level predictions $P_1, \ldots, P_m$

Second-level prediction P_t

$D_{new}=\emptyset$

For i=1 to m

$C_i=LA_i(D)$

End

For i=1 to n

For j=1 to m

$z_{ij}=Ct(x_i)$

End

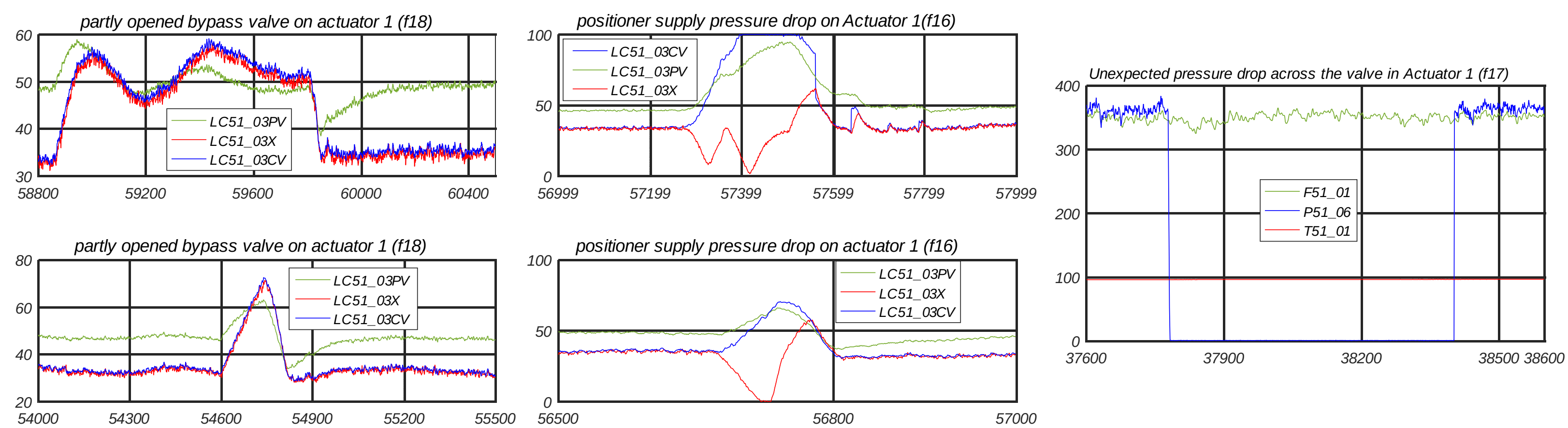
$D_{new} = D_{new} \cup \{(z_{i1},z_{i2}, \ldots, z_{im}),y_i)\}$

End

Ct=$LA(D_{new})$

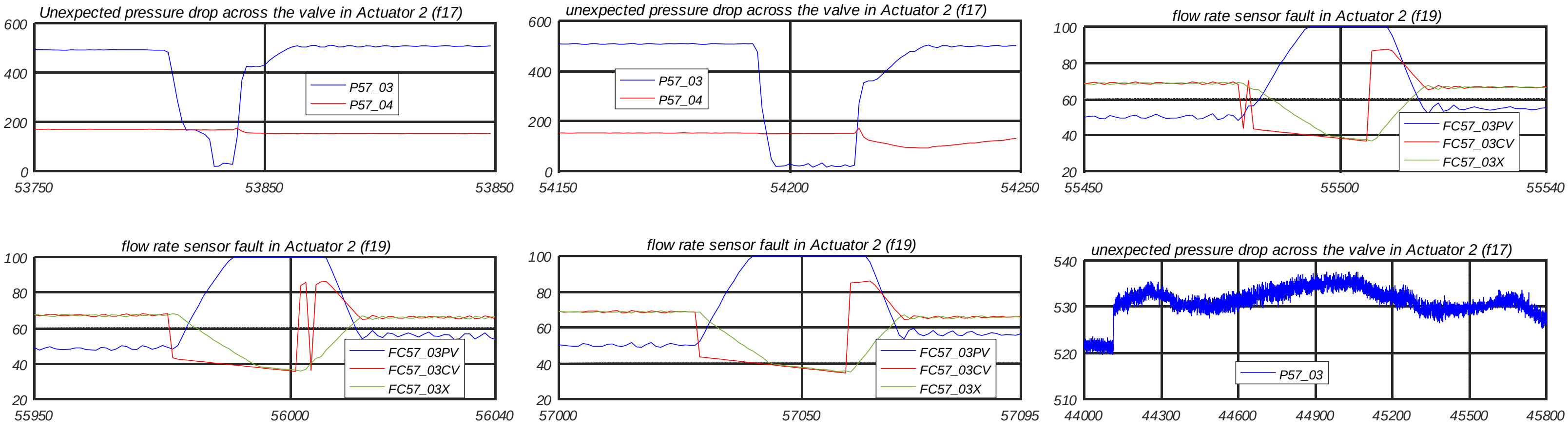
$Pt=Ct(P_1, P_2, \ldots, P_m)$

Figure



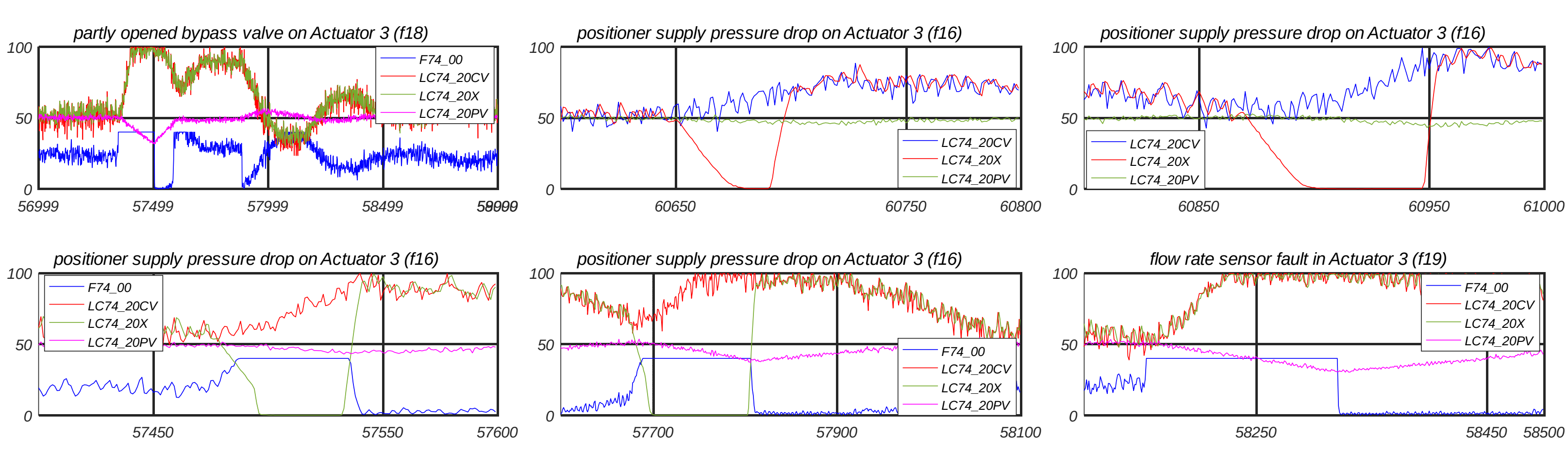
partly opened bypass valve on actuator 1 (f18)
LC51_03PV
LC51_03X
LC51_03CV
positioner supply pressure drop on Actuator 1(f16)
LC51_03CV
LC51_03PV
LC51_03X
Unexpected pressure drop across the valve in Actuator 1 (f17)
F51_01
P51_06
T51_01
partly opened bypass valve on actuator 1 (f18)
LC51_03PV
LC51_03X
LC51_03CV
positioner supply pressure drop on actuator 1 (f16)
LC51_03PV
LC51_03X
LC51_03CV

Figure

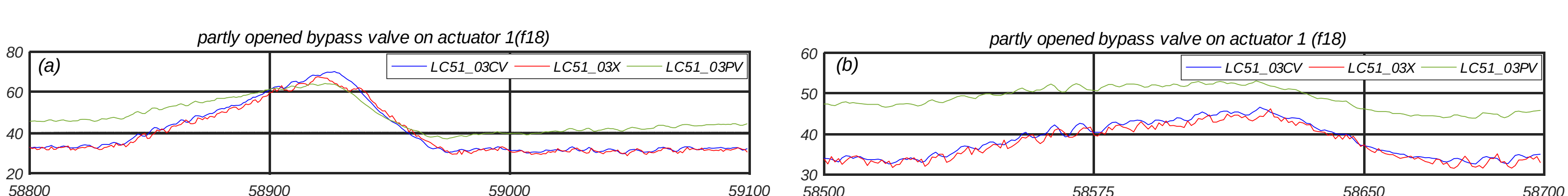


Unexpected pressure drop across the valve in Actuator 2 (f17)
600
400
200
0
53750
53850
53850
P57_03
P57_04
unexpected pressure drop across the valve in Actuator 2 (f17)
600
400
200
0
54150
54200
54250
P57_03
P57_04
flow rate sensor fault in Actuator 2 (f19)
100
80
60
40
20
55450
55500
55540
FC57_03PV
FC57_03CV
FC57_03X
flow rate sensor fault in Actuator 2 (f19)
100
80
60
40
20
55950
56000
56040
FC57_03PV
FC57_03CV
FC57_03X
flow rate sensor fault in Actuator 2 (f19)
100
80
60
40
20
57000
57050
57095
FC57_03PV
FC57_03CV
FC57_03X
unexpected pressure drop across the valve in Actuator 2 (f17)
540
530
520
510
44000
44300
44600
44900
45200
45500
45800
P57_03

Figure


partly opened bypass valve on Actuator 3 (f18)
F74_00
LC74_20CV
LC74_20X
LC74_20PV
positioner supply pressure drop on Actuator 3 (f16)
LC74_20CV
LC74_20X
LC74_20PV
positioner supply pressure drop on Actuator 3 (f16)
LC74_20CV
LC74_20X
LC74_20PV
positioner supply pressure drop on Actuator 3 (f16)
F74_00
LC74_20CV
LC74_20X
LC74_20PV
positioner supply pressure drop on Actuator 3 (f16)
F74_00
LC74_20CV
LC74_20X
LC74_20PV
flow rate sensor fault in Actuator 3 (f19)
F74_00
LC74_20CV
LC74_20X
LC74_20PV

Figure


partly opened bypass valve on actuator 1(f18)
(a)
LC51_03CV
LC51_03X
LC51_03PV
80
60
40
20
58800
58900
59000
59100
partly opened bypass valve on actuator 1 (f18)
(b)
LC51_03CV
LC51_03X
LC51_03PV
60
50
40
30
58500
58575
58650
58700

Response Letter to Reviewers

The authors would like to express their gratitude to anonymous reviewers whose valuable comments and suggestions made a major contribution to improve the paper.

Reviewer #1: The paper is well written and structured. The novelty of the idea proposed by the authors needs to be reasserted by a more in-depth discussion revolving around the following areas:

- CM11. Dataset gathering for faulty operation in real-life industrial plants, especially for the purpose of faults isolation.
  As a platform for technology transfer the benchmark is based on the complete working electro-mechanical details of an electro-pneumatic valve actuator used in virtually all industrial processes. The benchmark system includes a study of complex system behavior: static and dynamic nonlinearity, multi-signals, closed-loop operation, a complete set of simulated faults in all the possible subsystems of electro-pneumatic valve actuators and an interesting set of real faults tested in the chosen sugar production process [Ref1]. The actuator Simulink model was tuned and validated by means of data acquired from real industrial process (three different actuators installed in the Lublin Sugar Factory). The model can be used for generating the data from normal and abnormal (faulty) states, thus allowing learning or tuning FDI algorithms. Moreover, it is also assumed that all measurements in the benchmark steps based on the actuator simulation model are disturbed by artificial noise [Ref1].

  As far as we understand from this comment, there is not a novelty in dataset gathering phase because we use a benchmark and the datasets were gathered and proposed by the benchmark developers and were the same datasets as exploited by other researchers in the literature.

  Besides, due to safety critical issues, collecting much data from faulty operating conditions of the system for fault isolation task is not possible. Therefore, existing of a high-fidelity simulator model of the system where faulty scenarios are simulated and validated against real-world applications can be very useful.

- CM12. Faults discernibility analysis based on the sensitivity of the classifiers.
  This comment is taken into consideration and the paper have been improved in accordance to the reviewer suggestion: A thorough sensitivity analysis in regards to all possible classes of system behavior has been carried out and added to the paper in Table 6 and last paragraph of Page 15. It includes sensitivity and specificity results of parallel FDI form for all classes of system's operating conditions where *one against all* approach is employed to calculate these measures based on the Equations 17-18 [Ref9].

With reference to Table 6, it is immediately seen that for most of faulty cases fi (i=1~19), almost all fi instances are truly classified as fi and non-fi instances are not classified as fi by using BL algorithm. In other words, BL provides the best results in terms of sensitivity and specificity measures for all classes of system's behavior compared to other presented methods. However, there are some cases reported in Table 6 where sensitivity of blended learning is a little bit lesser than 1 (such as f10 and f13) that can be due to the fact that some classes of faults are closely arranged in the feature/variable space or even slightly overlap each other.

- CM13. Analytical approach to generalize the results and performance improvements claimed in the DAMADICS benchmark

  This comment is taken into consideration. With reference to the remarks given in response to [CM15], it is worth noting that some methods, *e.g.* structural analysis FDI methods will not be suitable as system model dynamics are not provided in the DAMADICS benchmark. FDI methods based on the use of identification, together with unknown input observers, Kalman filters, neural networks or neuro-fuzzy methods are most suitable as the models derived do not depend upon analytical knowledge of the dynamics of the overall actuator system. Pattern recognition-based methods (as proposed in this paper) can also be exploited here as there is no need of analytical model dynamics [Ref1]. Same justification is also added to the last paragraph of the Section 2 on Page 5.
- CM14. On page 17, please correct small type: DAMADICS instead of DAMADIC

  This comment is taken into consideration. The paper was refined in terms of grammar and typos.
- CM15. Implementation constraints in real-life plant (this example is limited to an actuator. How can this be generalized to a more complex process unit (delayed Coker unit for example) with more actuators and process variables?) Availability of data for fault isolation is usually very scarce in those cases.

  The authors thank the anonymous reviewer for the opportunity to clarify these important aspects. A number of powerful methods have been developed for early detection and isolation of actuator faults [Refs 5, 6]. It seems that there is a lack [Ref 7] of FDI benchmarks when considering the widespread use of industrial actuators. The well-known ''three tank system'' [Ref6] benchmark, extensively used for FDI methods evaluation seems to be inadequate when considering industrial implementations. Some classes of FDI approaches require data acquired from faulty process states, these approaches must be ruled out as this requirement is very difficult to meet in practice. Therefore, the main aim of actuator benchmark is to set the evaluation frames of industrial applicability of FDI methods under the assumption of unknown actuator behavior in the faulty state, poor or unavailable analytical description of the process and the actuator itself. In [Ref1], it was claimed that the DAMADICS benchmarking meets the specification of a well-defined benchmark.

However, in order to tackle the FDI/FTC problem of the Coker process through the proposed FDI method in this paper, good development of the benchmark of the process actuators has to be performed provided that it must [Ref1]:

1) be FDI method independent,
2) be focused on the industrial actuator
3) use both a Simulink model of the actuator as well as real process data,
4) not contain any analytical description of the actuator,
5) not describe the analytical model of the controlled system

Therefore, the suggested FDI method is a data-driven approach and can be useful when an analytical (physics-based) model of the system is not in hand, and, if we want to cope with complex process plants with a large number of actuators and process variables (*e.g.*, Coker process unit), we must have a complex high-fidelity simulator. That is, since the proposed FDI method is powerful and can extract all the information on the data (due to Blended learning), therefore, it is not depended on the complexity of the process, while, it relies on the complexity and fidelity of the available simulator which simulates a wide range of operating conditions. Furthermore, feature extraction and selection methods can be also used in the proposed method to select the most important variables and reduce the dimension of the feature space.

- CM16. Impact of multiple simultaneous faults and their consequences for faults isolation. The authors thank the anonymous reviewer for the opportunity to clarify this aspect. The main objective of this research is to tackle detection and isolation of singe faults scenarios given by the developers of the DAMADICS. However, the proposed FDI scheme can simply deal with the combined simultaneous faults by considering each possible combination of single faults as an independent class of system's behavior as $f_{xx}+f_{yy}$. This can also yield to a supervised classification problem to be solved by BL. Same clarification has been added to the manuscript in the last paragraph of the Conclusion section on Page 17. It is also worth highlighting that due to the accurate results achieved under single incipient (*e.g.*, f2, f3, f4, f5) faulty scenarios, double simultaneous incipient faulty cases can be also considered in this research as it is the case that is ignored in most double-faults classification schemes.

Reviewer #4: The manuscript provides details of a multiple classifier technique and applies it to a benchmark simulation of a control valve. The application and results are interesting but not altogether unexpected. Overall I believe that the manuscript provides a reasonable contribution but there are modifications that should be made to it to clarify certain aspects. In particular:

- CM21.The introductory section should make the contribution of the work much clearer. There are many multiple classifier techniques that have been developed and it should be made clear what limitations the proposed approach addresses.

The authors thank the anonymous reviewer for this important consideration which is carefully taken into consideration in the following answer. Same justification is also given in the last 10 lines of the fourth paragraph of the Section 1 on Page 3.

The main contributions of the suggested FDI method can be as follows: it was shown that it outperforms the other related data-driven methods applied for FDI of the DAMADICS application and it is also the first time that ensemble classification based on the Blended learning is used for FDI of such a process actuators system. Moreover, the proposed FDI method can cope with complex process plants with a large number of actuators and process variables such as DAMADICS because it is powerful method and can extract all the information on the data (due to Blended learning). Therefore, the proposed FDI method is not depended on the complexity of the process, whereas, it relies on the complexity and fidelity of the available simulator which simulates a wide range of system's operating conditions.

However, it can be shown that a large group of diverse collaborative models leads to an accurate BL-based prediction system. A large ensemble of supervised learning techniques can also improves the accuracy of the whole ensemble by clever blending. Accordingly, the number of input attributes in the second-level dataset is multiplied by the number of system's health classes. These can be seen as the main limitations of the proposed BL method, and therefore, a compromise has to be made between the accuracy of the multiple-classifiers scheme and the number of combined models which are used in the ensemble.

- CM22. It is stated several times that the classifier technique is model free. The definition of model free needs to be clarified as I would consider at least some of the classifiers as models. In particular, PLS is a model based approach.

Many thanks to the anonymous reviewer who allowed us to clarify this important point: From a general standpoint, PCA and PLS can be regarded as tools/algorithms exploited in Model Based or Model Free FDI methods depending on the application. In particular, PCA and PLS can be used jointly with dynamic model structures such as ARX, ARMAX, NARX in a model-based FDI method based on identification approach [Ref8]. In this case, they are used to estimate the parameters of the models. Anyway, other optimization algorithms such as Genetic algorithms, Particle Swarm Optimization can be also used to identify ARX, ARMAX, NARX models in identification-based model-based FDI approach. On the other hand, PCA or PLS (and also other algorithms) can be exploited as classifiers within a data-driven model-free approach as proposed in our work [Ref10, 11].

Additionally, model-free methods are considered those that do not require the direct estimation of a time domain model of a plant to generate symptoms [Ref11]. So, they could provide powerful solutions in the industrial oriented setup defined by the DAMADICS benchmark. Roughly speaking, a model-free method mainly consists in addressing the FDI problem from the control-board operator's point of view. The basic idea is to monitor on-line the measurements of the control system variables without the need for defining explicit dependence laws in time-domain among them. By analysis of the measured variables the operator can decide about the operating mode of the plant and raise alarms. In this context, sometimes the problem can be assimilated to a pattern-recognition problem (see, for example, [Ref10]).

From a modeling perspective, there are methods that require accurate process models, semi-quantitative models, or qualitative models (Model-based FDI). At the other end of the spectrum, there are methods that do not assume any form of model information and rely only on historic process data. In addition, given the process knowledge, there are different search techniques that can be applied to perform diagnosis [Refs 2, 3, and 4]. The latter is the topic covered by the present research. Moreover, Fault diagnosis can be a classification problem and hence can be cast in a pattern recognition framework. Methods that extract quantitative information can be broadly classified as non-statistical or statistical methods. Neural networks are an important class of non-statistical classifiers. Principal component analysis (PCA) or partial least squares (PLS) and other statistical pattern classifiers form major statistical classification methods that are widely used in process analysis and fault diagnosis. Furthermore, a model-based FDI method uses an input-output model of the system which is created based on physics-based rules or dynamic system identification, and thereby, the indicative residual signals which are the difference between systems output and the model output are generated and are evaluated to make a decision on the occurrence and location and size of the faults [Ref9].

Consequently, the proposed FDI method in this paper can be considered as a model-free technique, since it is not necessary to resort to any explicit I/O model of the plant (at least in time domain) [Ref11].

- CM23. The issue of false alarms is very important. Are there any data sets available that really excite the valve dynamics but contain no faults? A thorough analysis of the ability to test against false alarms is very important.

This comment is taken into consideration and the paper have been improved in accordance to the reviewer suggestion: A thorough false alarm analysis in regards to all possible classes of system behavior has been carried out and added to the paper in Table 7 and first paragraph of Page 16. With reference to Table 7, it is obvious that by using the BL algorithm in most of faulty scenarios the best value for false alarm rate is 0 that means that no false alarms at all are raised at any time when the fault detection algorithm is on. It is also worth noticing that although in some cases false alarm rate is not 0, it is still retained almost near 0 which can be acceptable due to the presence of un-controlled effects such as noise and disturbance acting on industrial actuator system [Ref1].

- CM24. The results in this manuscript are compared with those obtained in 10, 19 and 20. Do these studies describe 'bagging' approaches? If not are there other bagging techniques which the proposed technique could be compared against?

  The authors thank the anonymous reviewer for this valuable comment. References [19 and 10] are referred to as the data-driven works on this benchmark challenge where high accuracy results obtained so far. However, both these works do not present any ensemble-based classification scheme and rely on single classifiers. According to our best knowledge, reference [Ref 10] is the only work where ensemble of the classifiers (on the basis of bagging and/or stacking techniques) was presented for FDI of the DAMADICS benchmark. However, it does not present any results in regards to incipient faults which are hardly detectable since they develop very slowly over the time and may be also masked by the action of the closed-loop controllers [Ref 1].

  It is also worth stressing that the basic ways of aggregating classification methods is bagging that creates solutions to be combined. In contrast, the suggested ensemble classifier in this paper combines the existing solutions by Blending technique in the sense that in order to determine the value of each new hypothesis (diverse classifier model), the algorithm re-evaluated the aggregation set while stacking each new model, sequentially.

Some minor issues:

- CM25. In the first paragraph, I believe that there are many other approaches to improving availability and reliability. In fact FDI probably contributes very little to this as the currently applied techniques are quite crude.

  The authors thank the anonymous reviewer for the opportunity to clarify this very important point involving the whole FDI community. As industrial systems have become more highly integrated and complex, the faults occurring in modern processes present monitoring challenges that are not readily addressed using univariable charts (e.g., Shewhart charts). The weaknesses of univariate control charts for detecting faults in multivariate processes as well as the insufficiency of PID and robust control in dealing with faults have led to a surge of research literature concentrated on developing better methods for FDI. This growth of research activity can be explained by the fact that industrial systems are becoming more heavily instrumented, resulting in larger quantities of data available for use in FDI, and that modern computers are becoming more powerful. The availability of data collected during various operating and fault conditions is essential to FDI. The storage capacity and computational speed of modern computers enables FDI algorithms to be computed when applied to large quantities of data [Ref12].

- CM26. On page 8. Section 3.1 is incorrectly numbered.

  This comment is taken into consideration.

- CM27. Also found the opening paragraph to be a little confusing and it would benefit with re-wording.

  This comment is taken into consideration. The opening paragraph has been re-written more carefully.

References:

[Ref1] Bartys, M., R. Patton, M. Syfert, S. de las Heras, and J. Quevedo (2006). Introduction to the DAMADICS actuator FDI benchmark study. Control Engineering Practice, (Vol. 14(6), 577-596).

[Ref2] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri. A review of process fault detection and diagnosis part i: Quantitative modelbased methods. Computers and Chem. Eng. 27, 293—311 (2003).

[Ref3] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri. A review of process fault detection and diagnosis part II: Qualitative models and search strategies. Computers and Chem. Eng. 27, 313—326 (2003).

[Ref4] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri. A review of process fault detection and diagnosis part III: Process history based methods. Computers and Chem. Eng. 27, 327—346 (2003).

[Ref5] Blanke, M., & Patton, R. J. (1995). Industrial actuator benchmark for fault detection and isolation. Control Engineering Practice, 3(12).

[Ref6] Patton, R. J., Frank, P. M., & Clark, R. N. (2000). Issues in fault diagnosis for dynamic systems. London: Springer.

[Ref7] Mediavilla, M., de Miguel, L. J., & Vega, P. (1997). Isolation of multiplicative faults in the industrial actuator benchmark. In Proceedings of the IFAC symposium SAFEPROCESS 97, Kingston Upon Hull, UK (pp. 855–860).

[Ref8] Lennart Ljung, System Identification;Theory for the User, 2nd Ed., Prentice-Hall,. 1999

[Ref9] Nozari HA, Simani S, Shoorehdeli MA, et al. Model-based robust fault detection and isolation of an industrial gas turbine prototype using soft computing techniques. Neurocomputing 2012; 91: 29–47.

[Ref10] Guglielmi, G., Parisini, T., & Rossi, G. (1995). Fault diagnosis and neural networks: a power plant application, Keynote Paper. Control Engineering Practice, 3, 601–620.

[Ref11] Previdi F. and Parisini T., (2006). Model-free actuator fault detection using a spectral estimation approach: the case of the damadics benchmark problem, Control Engineering Practice, (Vol. 14, pp. 635–644).

[Ref12] Chiang L.H.,. Russell E.L, R.D. Braatz, Fault Detection and Diagnosis in IndustrialSystems, Springer Verlag, London, UK, 2001.