Hindawi
Journal of Sensors
Volume 2018, Article ID 7467418, 21 pages
https://doi.org/10.1155/2018/7467418

Research Article

A Novel Feature Selection Scheme and a Diversified-Input SVM-Based Classifier for Sensor Fault Classification

Sana Ullah Jan and Insoo Koo

School of Electrical Engineering, University of Ulsan, Ulsan 44610, Republic of Korea

Correspondence should be addressed to Insoo Koo; iskoo@ulsan.ac.kr

Received 26 May 2018; Accepted 29 July 2018; Published 5 September 2018

Academic Editor: Giuseppe Maruccio

The efficiency of a binary support vector machine- (SVM-) based classifier depends on the combination and the number of input features extracted from raw signals. Sometimes, a combination of individual good features does not perform well in discriminating a class due to a high level of relevance to a second class also. Moreover, an increase in the dimensions of an input vector also degrades the performance of a classifier in most cases. To get efficient results, it is needed to input a combination of the lowest possible number of discriminating features to a classifier. In this paper, we propose a framework to improve the performance of an SVM-based classifier for sensor fault classification in two ways: firstly, by selecting the best combination of features for a target class from a feature pool and, secondly, by minimizing the dimensionality of input vectors. To obtain the best combination of features, we propose a novel feature selection algorithm that selects m out of M features having the maximum mutual information (or relevance) with a target class and the minimum mutual information with nontarget classes. This technique ensures to select the features sensitive to the target class exclusively. Furthermore, we propose a diversified-input SVM (DI-SVM) model for multiclass classification problems to achieve our second objective which is to reduce the dimensions of the input vector. In this model, the number of SVM-based classifiers is the same as the number of classes in the dataset. However, each classifier is fed with a unique combination of features selected by a feature selection scheme for a target class. The efficiency of the proposed feature selection algorithm is shown by comparing the results obtained from experiments performed with and without feature selection. Furthermore, the experimental results in terms of accuracy, receiver operating characteristics (ROC), and the area under the ROC curve (AUC-ROC) show that the proposed DI-SVM model outperforms the conventional model of SVM, the neural network, and the k-nearest neighbor algorithm for sensor fault detection and classification.

1. Introduction

The sensors in industrial systems are listed as a second major source of faults after the rolling elements (e.g., bearings) on top [1–3]. These faults lead to intolerable consequences including an increase in maintenance costs, compromising the reliability of products and, even more critical, safety [3, 4]. These issues can be avoided significantly by detecting fault appearance instantly. Therefore, the output of a sensor is monitored to promptly identify the anomaly. After detecting, it is mandatory to find out the primary reason of the fault occurrence in order to implement safety measures. For this purpose, faults are categorized into predefined classes obtained from historical data, a process referred to as classification.

Recently, machine learning (ML) techniques, such as neural networks (NN), support vector machines (SVM), and k-nearest neighbors (KNN), are favored for classification problems due to an efficient performance [5–12]. However, they require a feature extraction method to overcome the curse of high dimensionality of input signals. This method characterizes the hundred- or even thousand-dimensional input signal by extracting a few features. Then, these features are used as inputs to the classifiers instead of the raw signal. Although this technique may increase the efficiency in some cases, mostly there is no or very little improvement in the performance of the classifier. A major reason is the use of features with low discriminating power between the samples of different classes. The good features are the ones that characterize the signals from all classes in a dataset such that a signal of one class is easily discriminated from others. To pick such good features from a feature pool, a feature selection (FS) algorithm is used. Therefore, the feature selection step has a dominant role in reliable performance of classifiers in pattern recognition and classification applications. Nevertheless, a combination of individually good features does not necessarily lead to a good classification performance [13]. Therefore, a good FS algorithm tries to find the best combination of features and not a combination of individually good features. Moreover, as mentioned earlier, the complexity of the system is dependent on the dimensions of input vectors in direct relations. Therefore, selecting the fewest possible features when selecting good features can lead to a classifier performance enhancement.

FS algorithms are classified as wrapper-based, embedded-based, and filter-based [14–16]. The wrapper-based methods select features by analyzing the performance of a classifier after each selection. The features that optimize the performance of the classifier are selected. These techniques require high computational resources and a long time to get the best features; even so, optimality is not ensured. Embedded-based feature selection optimizes the classifier and feature selection simultaneously. The problem with these approaches is that the features are selected for the classifier under consideration and may not be able to merge with any other classifier. In contrast, the filter-based feature selection approach selects features irrespective of classifier optimization. In this approach, mutual information (MI) is a widely used measure to select the features most relevant to the target class.

The objective of the current work is to develop an efficient detection and classification algorithm for sensor faults using a supervised ML-based classifier. For this purpose, we first propose a novel MI-based FS scheme to select the exclusive relevant features with high discriminating power from a feature pool. Then, we propose a diversified-input SVM (DI-SVM) model to reduce the dimensions of input vectors to the classifiers ultimately improving the performance.

1.1. Contributions. The major contributions of this work are summarized as follows.

(i) A filter-based FS algorithm is proposed to select a combination of features exclusively discriminating one class from others. For a target class, the measure of MI of the appointed feature on the nontarget class is subtracted from the measure of MI of the same feature on the target class. This technique selects the features having high capability of discriminating target class and, at the same time, having less sensitivity to nontarget classes. The results show the efficiency of the proposed FS scheme over other schemes which utilize the redundancy between the features in addition to relevance to measure the goodness of features.

(ii) Furthermore, we propose a DI-SVM model to reduce the dimensions of input vectors to classifiers. This model inputs a diversified combination of features to different SVMs utilized for multiclass classification. These combinations of features input to any classifier are selected by the feature selection algorithm. For example, the c^{th} classifier is fed with a combination that is selected for the c^{th} class. On the other hand, in a conventional way of using SVMs for multiclass classification, all classifiers are trained with the exact similar number and set of features selected for all classes. The experimental analysis shows that the proposed DI-SVM model further improves the classification performance of the conventional SVM.

(iii) The performance of the proposed methodology is analyzed using two datasets. In the first dataset, the faulty signals are obtained by keeping the index of fault insertion point fixed at 500 during simulation, whereas the second dataset is obtained using faulty signals with fault insertion point varying from 0 to 1000 in each sample. The latter case is used to replicate a practical scenario in industrial systems. The results show that it is more challenging to detect and classify faults in the second case. However, the framework of the proposed FS scheme and DI-SVM achieves satisfactory results in this case also.

(iv) A series of five experiments are performed to compare the performance of the proposed DI-SVM model with those of the conventional SVM, NN, and KNN classifiers. The first two experiments are performed using the conventional SVM, NN, and KNN classifiers without applying a feature selection. In the next experiment, we prove the efficiency of the proposed FS algorithm using measures of accuracy, receiver operating characteristics (ROC), area under the ROC (AUC-ROC), and scatter plots in selected feature spaces. In the last two experiments, we deploy the proposed FS and DI-SVM model along with the abovementioned classifier and compare the performances. The results show that DI-SVM outperforms all the counter three classifiers.

This paper is organized as follows. A review of related works and the proposed feature selection algorithm are presented in Section 2. The theory of the classification model and proposed DI-SVM model is presented in Section 3. The experimental results are illustrated in Section 4. Section 5 presents the discussion about the experimental results, and finally, the study is concluded in Section 6.

2. Mutual Information-Based Feature Selection

2.1. Preliminaries. Before presenting a review of the related works, we present fundamental background knowledge about the MI-based feature selection schemes. For two random variables, x and y, the MI is defined as follows [17]:

$$I(x;y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy, \quad (1)$$

where $p(x)$ and $p(y)$ are probability density functions of continuous random variables x and y, respectively, and $p(x,y)$ is their joint probability density function. For discrete random variables, the integration is replaced with summation as follows:

$$I(x;y) = \sum_{x\in X}\sum_{y\in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}, \quad (2)$$

where $p(x)$ and $p(y)$ are probability mass functions of discrete random variables x and y, respectively, and $p(x,y)$ is their joint probability mass function. The MI can be expressed in terms of entropy as follows:

$$I(x;y) = H(x) - H(x \mid y) = H(y) - H(y|x). \quad (3)$$

$H(x)$ and $H(x \mid y)$ represent the entropy and conditional entropy, defined as follows:

$$H(x) = -\sum_{r\in x} p(r) \log p(r), \quad (4)$$

$$H(x|y) = -\sum_{s\in y}\sum_{r\in x} p(r,s) \log p(r \mid s). \quad (5)$$

2.2. Related Works. In this paper, we focus on the selection criteria of features based on the measure of MI. Different FS algorithms have been proposed in the past using different criteria to measure the goodness of features using MI. Battiti [18] selected features by calculating the MI $I(x_i;c)$ of individual feature x_i with class c. To avoid the selection of redundant features, the MI measure $I(x_i;x_j)$ between two features x_i and x_j is calculated. The MI-based feature selection (MIFS) is given as [18]

$$I(x_i;c) - \beta \sum_{x_j\in S} I(x_i;x_j), \quad (6)$$

where β is the regularization parameter to weight the redundancy $I(x_i;x_j)$ between a candidate feature x_i and the already-selected features $x_j \in S$. Kwak and Choi [19] proved that a large value of β can lead to the selection of suboptimal features. They improved the MIFS scheme to propose MIFS-U by making modifications as follows

$$I(x_i;c) - \beta \sum_{x_j\in S} \frac{I(x_j;c)}{H(x_j)} I(x_j;x_i). \quad (7)$$

In both of these algorithms, the selection of a feature is dependent on the user-defined parameter β weighting the importance of redundancy between features. If β is selected too large, the feature selection algorithm will be dominated by the redundancy factor. The authors of [13] proposed parameter-free criteria of feature selection, named as minimal-redundancy-maximal-relevance (mRMR), given by

$$I(x_i;c) - \frac{1}{|S_{m-1}|} \sum_{x_j\in S_{m-1}} I(x_j;x_i), \quad (8)$$

where m is the number of features to be selected.

Estevez et al. [20] pointed out that the right-hand sides of MIFS and MIFS-U, in (6) and (7), will increase with the increase in the cardinality of the selected feature subset. This will result in the dominating right-hand side, thus forcing the FS to select nonredundant features. This may lead to the selection of irrelevant features before relevant features. Another problem is that there is no technique to optimize β and its value depends highly on the problem under consideration. Although the mRMR partly solves the first problem in MIFS and MIFS-U, the performance is still comparable with those of MIFS and MIFS-U [21].

A normalized MIFS (NMIFS) is proposed to address the above problems and is given as [20]

$$I(x_i;c) - \frac{1}{|S_{m-1}|} \sum_{x_j\in S_{m-1}} \frac{I(x_j;x_i)}{\min\{H(x_j), H(x_i)\}}. \quad (9)$$

An improved version of NMIFS (I-NMIFS) is given in [21] as follows:

$$\frac{I(x_i;c)}{\log_2(|\Omega_c|)} - \frac{1}{|S_{m-1}|} \sum_{x_j\in S_{m-1}} \frac{I(x_j;x_i)}{\min\{H(x_j), H(x_i)\}}. \quad (10)$$

The problem with NMIFS and I-NMIFS is that they both rely on the measure of entropy of both the selected and under-observation features. The entropy of a feature is totally dependent on the number of samples from each class in the dataset. If one class has a high number of samples than the other, this situation will have an effect on the value of entropy of a feature leading to degradation of performance of the classifiers.

Furthermore, all these algorithms rely on the measure of mutual information of a to-be-selected feature with a target class (relevance) and with an already-selected feature (redundancy). In this paper, we focus only on the relevance of a feature with a target class and nontarget class to assess the goodness of a candidate feature. Indeed, a feature having high MI (i.e., relevance) with a target class might not be a good choice due to a high relevance to at least one nontarget class as well. This may lead to degrading rather than improving the performance of the classifier. Therefore, we propose a feature selection scheme that relies only on the measures of relative relevance of a feature with different classes in the dataset. The measure of redundancy is eliminated making the scheme simplified yet achieving satisfactory performance compared to that of the FS schemes utilizing the redundancy factor also in the selection criteria.

2.3. The Proposed Feature Selection Scheme. Given a training dataset composed of N samples and M features, the aim is to select m out of M features for each class c, individually, where $c \in \{1, 2, \dots, C\}$ and C is the number of total classes. The features having maximum relevance to class c and minimum relevance to remaining $c-1$ classes are most suitable for discriminating class c. Relevance is usually described by MI or correlation, of which the MI is a widely adopted measure to indicate the dependence between two random variables. Generally, the mutually independent variables have zero MI between them. The higher the dependency of two variables, the higher the MI.

The proposed feature selection algorithm selects features by considering their measures of MI with both target class and nontarget class. The m features having maximum relevance with a target class are fit for target class, but it may also have high relevance with a nontarget class, making this feature less discriminating. Therefore, the features having maximum relevance with a target class and minimum relevance with nontarget classes are selected. This approach selects the subtlest features among the feature sets for the target class exclusively. The relevance of features exclusive for the target class is obtained by calculating the difference of the MI of the selected feature vector $\mathbf{x}_i$, where $i = 1, 2, \dots, M$, on target class c and the MI of $\mathbf{x}_i$ on nontarget class c'. Mathematically,

$$d = I(x_i\,; c) - I\left(x_i\,; c'\right), \tag{11}$$

where $c' = C/c$ is the set of nontarget classes. The m among M features with maximum values of d is selected for target class c. The MI of x_i on class c is calculated in terms of entropy as follows:

$$I(x_i\,; c) = H(c) - H(c|x_i), \tag{12}$$

where $H(c)$ is the entropy of class c and $H(c \mid x_i)$ is the conditional entropy of c given x_i. Assuming $0.\log 0 = 0$ from continuity, $H(c)$ and $H(c \mid x_i)$ are defined as

$$\begin{aligned} H(c) &= -p(c)\log p(c) - p\left(c'\right)\log p\left(c'\right), \\ H(c|x_i) &= -\sum_{x\in x_i} p(x)\sum_{c=1}^{C} p(c|x)\log p(c|x), \end{aligned} \tag{13}$$

where $p(c)$ and $p(c')$ are the probabilities of the target class and nontarget class, respectively, and can be calculated as $p(c) = n_c/N$ and $p(c') = (N - n_c)/N$, respectively, where n_c is the number of samples corresponding to class c in the training dataset. The conditional probability $p(c \mid x)$ is given as

$$p(c|x) = \frac{p(c)p(x|c)}{\sum_{s=1}^{C} p(s)p(x|s)}, \tag{14}$$

where $s = 1, 2, \dots, C$ correspond to the set of all C classes. The probability mass function $p(x \mid c)$ can be estimated using the Parzen window method as follows [13]:

$$\widehat{p}(x|c) = \frac{1}{N_c}\sum_{i\in I_c}\Phi(x - x_i, h), \tag{15}$$

where I_c represents the set of indices of training samples corresponding to class c and $\Phi(\cdot)$ is the window function. The commonly used Gaussian window function is expressed as

$$\Phi(z, h) = \frac{\exp\left(-\left(z^T\Sigma^{-1}z/2h^2\right)\right)}{(2\pi)^{M/2}h^M|\Sigma|^{1/2}}, \tag{16}$$

where Σ is the matrix covariance of the M-dimensional vectors of random variables, $z = x - x_i$, and the width parameter h is given by $h = \beta/\log N$ for positive constant β. The appropriate selection of $\Phi(\cdot)$ and a large h can converge (16) to a true density [13]. Using (5), (6), and (7), the estimated conditional probability mass function can be obtained as

$$\widehat{p}(c \mid x) = \frac{\sum_{i\in I_c}\exp\left(-\left((x - x_i)^T\Sigma^{-1}(x - x_i)/2h^2\right)\right)}{\sum_{k=1}^{C}\sum_{i\in I_k}\exp\left(-\left((x - x_i)^T\Sigma^{-1}(x - x_i)/2h^2\right)\right)}. \tag{17}$$

A pseudocode of steps in the proposed FS algorithm is given in Algorithm 1. A training dataset $\{x_{ij}, c_i\}_{i=1}^{N}$ with N samples and M features, where x_{ij} is the i^{th} element of the j^{th} feature vector, is given input to the selection scheme. A matrix $\mathbf{X}$ of size $C \times m$ is the output of the algorithm where each c^{th} row contains the m features selected for targeting class c. In the first step, a target class c is selected among the sets of all classes $1, 2, \dots, C$. Then, the respective set of nontarget classes is obtained. In the next step, each feature vector is selected simultaneously. For each j^{th} feature vector, the elements from class c and class c' are stored in x_j^c and $x_j^{\bar{c}}$, respectively. The mean values of these vectors are stored in x_j^c and $x_j^{\bar{c}}$. After a series of calculation of the estimated conditional probability, entropy, and conditional entropy, d is calculated for given class c using (2). Then, the m features having maximum values of d in the final vector are selected and stored in the c^{th} row of the matrix $\mathbf{X}$. Similarly, m features are selected for each class and stored in the final matrix $\mathbf{X}$.

3. Classification Models

A generic framework of classification methodology using supervised machine learning-based classifiers is given in Figure 1. This methodology is performed in two phases: a training phase and a testing phase. In the training phase, a classifier is trained using a set of historical (or training) data.

```
INPUT: ${x_{ij}, c_i}_{i=1}^{N}$, Training dataset; $j = \{1, 2, \dots, M\}$, Number of features; $c \in \{1, 2, \dots, C\}$, Number of classes;
$m$, Number of features to select by algorithm.
OUTPUT: $X_{C\times m}$, Matrix of $C$ vectors of $m$ features selected for each class $c$.
1:  for $c$ in $C$ do
2:    $c' \leftarrow C/c$
3:    for $j$ in $M$ do
4:      $x_j^c \leftarrow$ values of $j^{th}$ feature from class $c$
5:      $x_j^t \leftarrow$ values of $j^{th}$ feature from class $c'$
6:      $x_j^c \leftarrow$ mean of $x_j$ from class $c$
7:      $x_j^t \leftarrow$ mean of $x_j$ from class $c'$
8:      Obtain $\hat{p}(c \mid x_j^c)$ and $\hat{p}(c' \mid x_j^t)$ using Eq(8)
9:      Obtain $H(c)$, $H(c')$ using Eq(4)
10:     Obtain $H(c \mid x_j)$ and $H(c' \mid x_j)$ using Eq(4)
11:     Obtain $I(x_j ; c)$ and $I(x_j ; c')$ using Eq(3)
12:     Obtain $d_j^c \leftarrow I(x_i ; c) - I(x_i ; c')$ using Eq(2)
13:   end for
14:   Indices of $m$ maximum values of $d^c \rightarrow X_{c\times m}$
15: end for
```

Algorithm 1: Proposed feature selection algorithm.

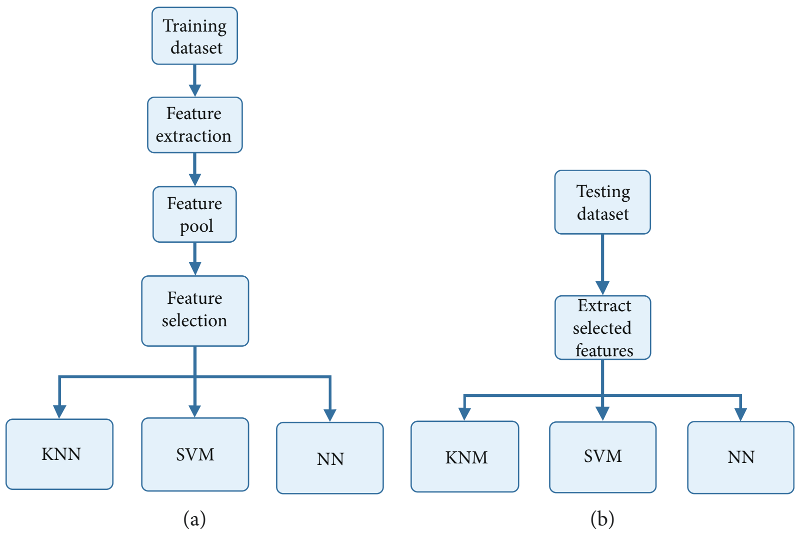

Figure 1: (a) Training phase and (b) testing phase of the generic fault classification framework for machine learning-based classifiers.

The steps include feature extraction from raw signals to obtain a feature pool. Then, an FS algorithm selects the best features in terms of discriminating power between data from different classes. Finally, the sets of the selected features are used to train the classifier, as illustrated in Figure 1(a). The performance of the trained classifier is evaluated using a set of unobserved test signals in the testing phase. The classifier is loaded with the particular features, selected in the training phase and extracted from the test signals as shown in Figure 1(b).

The commonly adopted supervised machine learning-based classifiers include NN, KNN, and SVM-based classifier. A short introduction to each one of these classifiers is given here for completeness. Then, the proposed DI-SVM-based classifier model is presented.

3.1. Neural Network. A neural network (NN), also known as feedforward NN or multilayer perceptron (MLP), is a machine learning tool for classification and regression problems. The structure of NN is inspired from the biological nervous system. It consists of an input layer, an output layer, and at least one hidden layer. An input signal is propagated forward from the input layer to the output layer to obtain the weight vectors of each layer. During this stage, the neurons in hidden and output layers learn about the input patterns from different classes. The neurons are triggered with a differentiable nonlinear activation function. A commonly used sigmoid activation function is given as

$$y_i = \frac{1}{1 + \exp(-v_i)}, \tag{18}$$

where v_i is the weighted sum of all synaptic inputs and the bias to the ith neuron and y_i is the output of the same neuron.

A backpropagation algorithm is used to estimate the difference between the predicted output of the network and the true result. An estimate of the gradient vector containing the gradients of the error surface with respect to the input weights of neurons in hidden and output layers is computed. The gradient vector is then passed backward from the output layer to the input layer to update the weight vectors of each layer. In this paper, a Levenberg-Marquardt (LM) algorithm is used to update the weights of neurons in backpropagation [22].

3.2. k-Nearest Neighbor. A k-nearest neighbor (KNN) is a simple classifier that takes a decision about the test input by simply looking at the k-nearest points in the training set [23]. The classifier counts the number of members from each class in this set of k-nearest neighbors to test the input and classifies the test signal in the class having the highest number of members. The Euclidean distance is commonly used as a distance metric to obtain the set of the nearest neighbors.

3.3. Support Vector Machine. The SVM is a well-known classifier primarily addressing two-class classification problems. A linear hyperplane is drawn as a decision boundary between the data points of the two classes. The optimal decision boundary is the one that maximizes the weight vector $\|\mathbf{w}\|$ [3, 24].

For a given set of training samples, x_i, and discrete class labels, c_i, the SVM tries to obtain the optimal weight vector, $\mathbf{w}$, using the following optimization problem:

$$\begin{aligned} &\min \quad \Phi(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + \alpha \sum_{i=1}^{N} \xi_i, \\ &\text{such that} \\ &c_i\left(\mathbf{w}^T x_i + b\right) \geq 1 - \xi_i, \quad \text{for } i = 1, 2, \dots, N, \xi_i \geq 0 \,\forall\, i, \end{aligned} \tag{19}$$

where α is the cost parameter and ξ_i represent the slack variables. A test observation, $\mathbf{x}_j$, is classified by the decision function, given as follows:

$$f(\mathbf{x}_j) = \operatorname{sign}\left(\sum_i \alpha_i K(\mathbf{x}_i, \mathbf{x}_j) + b\right), \tag{20}$$

where $\mathbf{x}_i$ represents the set of support vectors obtained in the training phase and $K(\mathbf{x}_i, \mathbf{x}_j)$ is the kernel function explained in the next section.

For a multiclass (more than two classes) classification problem, an equal number of binary SVMs to the number of classes in a dataset are utilized, where each SVM is used in a one-versus-rest approach [10, 25–27]. The cth classifier is trained with the samples from the cth class as a positive class and the samples from the remaining classes as a negative class.

3.3.1. Kernel Functions. To reduce the complexity of the machine in linearly nonseparable patterns, the data points are mapped on a higher-dimensional feature space where a linear decision surface can be drawn to classify the datasets of different classes. Inner-product kernels are used to perform this job. This is the key technique of SVM to solve the optimization problems of the input feature space in a higher-dimensional feature space. In such a way, the SVM deals with linearly nonseparable problems. For mapping function $\varphi(\cdot)$, the kernel function of two vectors $\mathbf{x}$ and $\mathbf{y}$ is given as $K(\mathbf{x}, \mathbf{y}) = \varphi(x)^T \varphi(y)$. A radial-basis function is a commonly used kernel function, defined as

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right). \tag{21}$$

3.4. Proposed Diversified-Input SVM Model. The training and testing phases of the proposed diversified-input SVM (DI-SVM) model for sensor fault detection and classification are illustrated in Figures 2(a) and 2(b), respectively. Given a training dataset composed of N labelled raw signals from a sensor, the first step is to extract M features to obtain the feature pool of size $N \times M$. The term raw signal refers to the signal obtained from a sensor, which is not preprocessed (i.e., any preprocessing technique, such as feature extraction and standardization, is not applied). In other words, the signals from the sensor in the earliest form are the raw signals, whereas the M-dimensional row vector in the feature pool, which is composed of the M features and extracted from a raw signal, is interchangeably referred to as a sample or an observation. After feature extraction to obtain a feature pool, the FS algorithm selects m out of M features for each class, separately. To perform this, the FS algorithm first selects a target class, c, from the given set of classes $1, 2, \dots, C$, where C is the total number of classes. Then, the goodness of each feature is assessed using a selection criterion to take a decision about selecting the features for the cth class. Subsequently, the next class is selected as the target class, and the feature selection steps are repeated. In this way, m features are selected for each class in the dataset. Finally, the column vectors in the feature pool of training data corresponding to the features selected for each single class are given inputs to C different SVMs, separately, for training. For example, if the two features x and y are selected targeting class c, then the vectors $\mathbf{x}$ and $\mathbf{y}$ containing the values of x and y from the feature pool are input to the SVM to specifically identify class c. Furthermore, the cth classifier is trained with observations of the cth class labelled as a positive class and the remaining $c - 1$ classes labelled as a negative class.

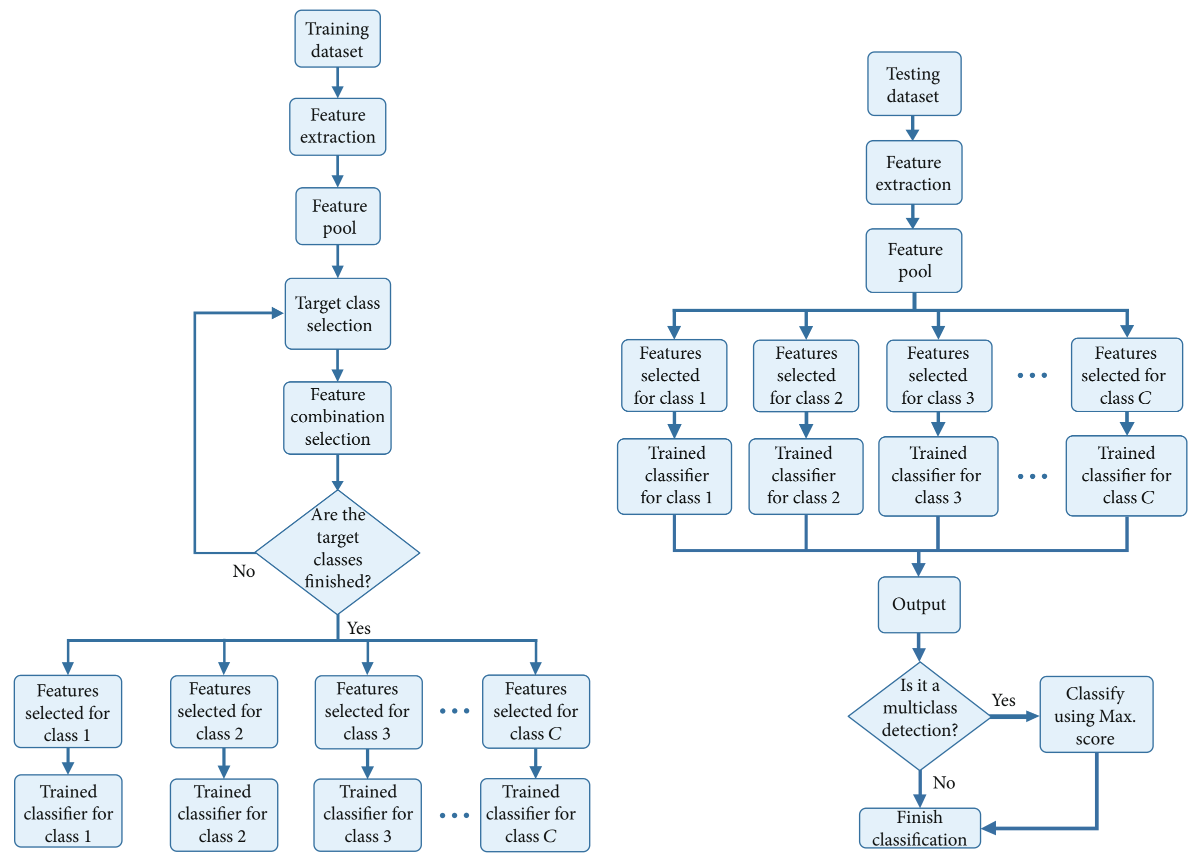
Training dataset
Feature extraction
Feature pool
Target class selection
Feature combination selection
Are the target classes finished?
No
Yes
Features selected for class 1
Features selected for class 2
Features selected for class 3
Features selected for class C
Trained classifier for class 1
Trained classifier for class 2
Trained classifier for class 3
Trained classifier for class C
(a)
Testing dataset
Feature extraction
Feature pool
Features selected for class 1
Features selected for class 2
Features selected for class 3
Features selected for class C
Trained classifier for class 1
Trained classifier for class 2
Trained classifier for class 3
Trained classifier for class C
Output
Is it a multiclass detection?
Yes
No
Classify using Max. score
Finish classification
(b)

Figure 2: Framework of (a) the training phase and (b) testing phase of the proposed diversified-input support vector machine (DI-SVM) model.

In such a way, C different SVMs are trained for C-class classification problems, where the observations input to each SVM are composed of diversified-feature combinations selected by the FS algorithm.

To classify a test raw signal, first, the features selected in the training phase are extracted from the raw signal to obtain the feature pool. The trained C SVMs are used to classify the test observation. Input to the SVMs is given with the same feature combinations as in the training phase. The output class labels and respective scores of all the SVMs are stored in the output vector. It is checked for any observation classified into multiclasses, that is, if any observation is classified positive by more than one SVM in the classifier. In this case, the decision of the SVM with the maximum output score is taken into account, as shown in Figure 2(b).

3.5. Classifier Parameters

3.5.1. Cross-Validation. The cross-validation (CV) techniques assess generalization and overcome the overfitting problem of the classifier in the training phase. The k-fold CV technique partitions the training data into k complementary subsets to train and validate SVM k times. In each round, a new subset is used for training, and the remaining samples are used for validation. This cycle is repeated until each sample is used for training at least once. Consequently, each subset is used once for training while being used $k - 1$ times for validating the classifier.

3.5.2. Standardization. Standardizing the training data prior to training the SVM may improve the classification performance [28]. This preprocessing technique is applied to prevent large valued features from dominating the small valued features in the dataset. The two commonly used ways of standardizing a dataset include the min–max (also known as normalization) and the z-score formula. In the min–max data standardization, the values of the data vector are scaled from 0 to 1. In contrast, the z-score formula scales the data vector to a standardized vector having zero mean and unit variance. Some other standardization techniques derived from these two basic methods are proposed in the literature [28, 29].

In fact, there is no obvious answer to which technique is the best. However, a basic difference between the two approaches is that the min–max method bounds the standardized data between 0 and 1. In this approach, a single outlier in the data may compress the remaining feature values toward zero. On the other hand, there is no limit on the range of standardized data using the z-score formula. This makes z-score standardization more robust

Table 1: Time-domain features.

Name	Definition	Name	Definition
Mean	$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$	Standard deviation	$Y_{\text{STD}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$
Root mean square	$Y_{\text{RMS}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$	Variance	$Y_{\text{VAR}} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$
Sign function	$Y_{\text{Sign}} = \text{sign}\,(\varepsilon - \mu)$	Square root of amplitude	$Y_{\text{SRA}} = \left(\frac{1}{N}\sum_{i=1}^{N}\sqrt{\lvert x_i\rvert}\right)^2$
Kurtosis value	$Y_{\text{KV}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i - \mu}{Y_{\text{STD}}}\right)^4$	Skewness value	$Y_{\text{SV}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i - \mu}{Y_{\text{STD}}}\right)^3$
Crest factor	$Y_{\text{CF}} = \max \frac{\lvert x_i\rvert}{Y_{\text{RMS}}}$	Impulse factor	$Y_{\text{IF}} = \frac{\max \lvert x_i\rvert}{(1/N)\sum_{i=1}^{N}\lvert x_i\rvert}$
Margin factor	$Y_{\text{MF}} = \max \frac{\lvert x_i\rvert}{Y_{\text{SRA}}}$	Form factor	$Y_{\text{FF}} = \frac{Y_{\text{RMS}}}{\mu}$
Kurtosis factor	$Y_{\text{KF}} = \frac{Y_{\text{KV}}}{\left((1/N)\sum_{i=1}^{N} x_i^2\right)^2}$	Peak-to-peak value	$Y_{\text{PPV}} = \max\,(x_i) - \min\,(x_i)$

Table 2: Frequency-domain features.

Name	Definition	Name	Definition
Center frequency	$Y_{\text{FC}} = \frac{1}{N}\sum_{i=1}^{N} X_i$	Sum FFT	$Y_{\text{SFFT}} = \sum_{i=1}^{r} X_i$
RMS frequency	$Y_{\text{RMSF}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} X_i^2}$	Root variance frequency	$Y_{\text{RVF}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - Y_{\text{CF}})^2}$

to the outliers in the data as compared to the min–max standardization method.

In z-score standardization, the input feature vectors are scaled to a zero mean by subtracting the statistical mean from each element of the vector and normalized to have a standard deviation of 1 by dividing by the standard deviation of the input vector. Given a vector, $\mathbf{x}_i$, the formula of z-score standardization is given as

$$\mathbf{x}_i^* = \frac{x_i - \mu_{x_i}}{\sigma_{x_i}}, \tag{22}$$

where $\mathbf{x}_i^*$ is the vector of standardized values and μ_{x_i} is the mean of the feature vector $\mathbf{x}_i$ with standard deviation σ_{x_i}.

3.6. Feature Pool. A feature pool was obtained by extracting 14 time-domain and 4 frequency-domain features. The respective names and mathematical definition of these features are illustrated in Tables 1 and 2.

4. Experimental Analysis

4.1. Dataset Acquisition. The types of sensor faults considered in this work include drift, erratic, hardover, spike, and stuck faults [3]. A plot of a raw signal from each of these classes and normal classes is given in Figure 3. The fault is considered a drift fault if the output of the sensor increases linearly. The hardover fault occurred when the output of the sensor is increased from a normal value, represented by a red line in Figure 3. If the erratic fault occurs, the variance of the output signal of the sensor is increased from a routine value. In the case of spike fault occurrence, spikes are observed in the output signal of the sensor. When the output of a sensor sticks to a fixed value, then this fault type is named as stuck fault. These fault types are divided based on the data measurements and can be considered generic in different types of sensors such as the pressure sensor. These faults are referred to by many researchers, with some using different nomenclature. For instance, Yu et al. [4] and Kullaa [30] referred to erratic, stuck, and hardover faults as precision degradation failure, complete failure, and bias failure, respectively.

The training and testing datasets utilized in this work are obtained using a healthy temperature-to-voltage converter TC1047/TC1047A as follows. A set of 100 raw time signals is acquired from a sensor using an Arduino Uno microcontroller board with a serial communication between the Arduino and personal computer. Each raw signal is

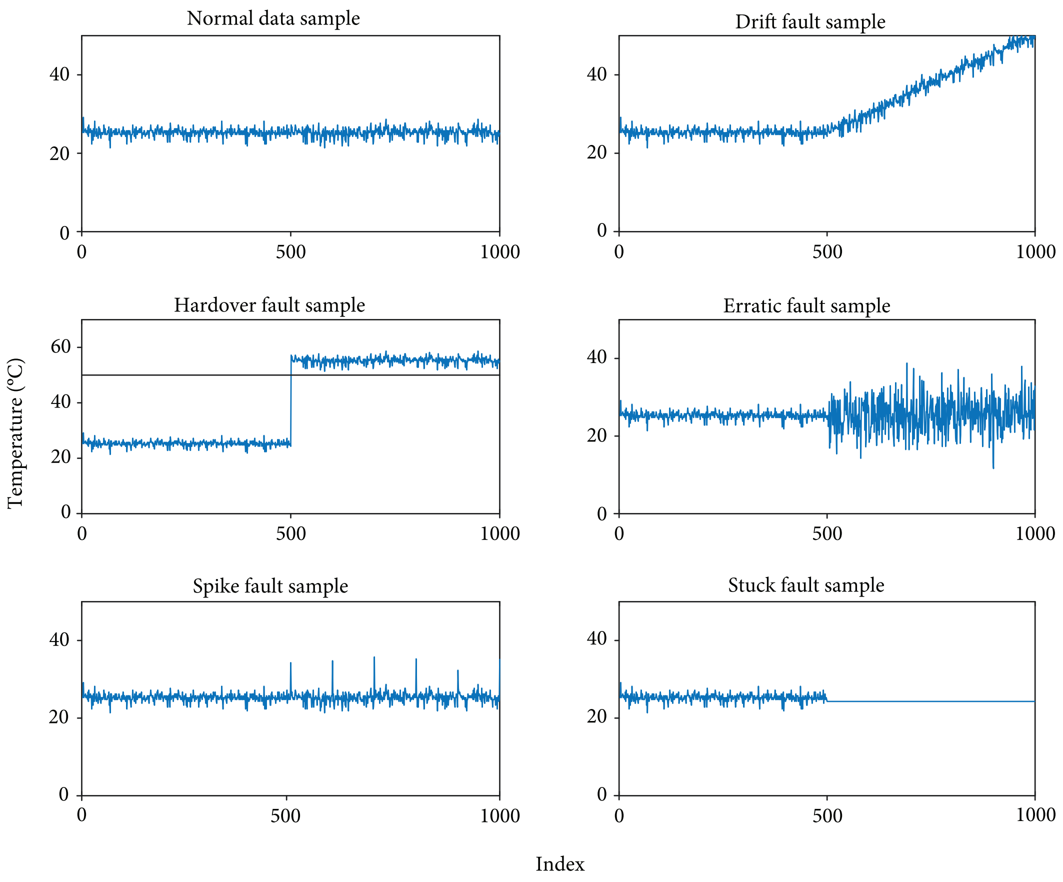

FIGURE 3: Plots of samples of all six classes from dataset 1 (with fixed fault insertion point).

composed of 1000 raw data elements from the sensor. Then, each fault type, that is, drift, hardover, erratic, spike, and stuck faults, is simulated in each stored signal obtained from a healthy sensor. In such a way, a set of 100 signals from each class, including normal and faulty classes, is obtained. The details of fault simulation in a normal data signal are as follows: a drift fault signal s_n^{Drift} is obtained by adding a linearly increasing bias term in a normal signal s_n^{Normal}, where the bias added to the n^{th} element is n times the constant initial bias b_0. A hardover fault signal s_n^{Hardover} is obtained by adding a large constant bias value b to all elements of the normal signal. To obtain an erratic fault signal s_n^{Erratic}, a signal $\dot{s}_n$ of mean 0 and high variance, $\dot{\delta}^2 \gg \delta^{2^{\text{Normal}}}$, where $\delta^{2^{\text{Normal}}}$ is the variance of the normal signal, is added to the raw normal signal. To obtain spike fault signals s_n^{Spike}, a constant bias is added periodically to the u^{th} elements of the normal signal, where $u = v \times \eta$ is the index of elements in the signal with $v = \{1, 2, \dots ,\}$ as a set of natural numbers and $\eta \geq 2$ as a positive integer. Finally, the stuck fault signals s_n^{Stuck} are obtained by keeping a fixed value at all indices of the normal signal. The statistical representations of these faults are given in Table 3, where $s_n^{\text{Normal}}(n = 1, 2, \dots , 1000)$ is the n^{th} element of the raw signal corresponding to the normal class. In conclusion, the final dataset is composed of 100 raw signals of each of the six classes. This means that 100 raw time signals of each class are included resulting in a total of 6 ∗ 100 raw time signals in the final dataset. As mentioned earlier, a raw signal is comprised of 1000 data elements. Therefore, the final dataset of raw signals has a size of 600 ∗ 1000 data points.

TABLE 3: Statistical representation of fault simulations in the normal signal.

Fault	Statistical Representation
Drift	$s_n^{\text{Drift}} = s_n^{\text{Normal}} + b_n, b_n = nb_0, b_0 = \text{constt}$
Hardover	$s_n^{\text{Hardover}} = s_n^{\text{Normal}} + b, b = \text{constt}$
Erratic	$s_n^{\text{Erratic}} = s_n^{\text{Normal}} + \dot{s}_n, \dot{s}_n \sim N(0, \dot{\delta}^2), \dot{\delta}^2 \gg \delta^{2^{\text{Normal}}}$
Spike	$s_n^{\text{Spike}} = s_n^{\text{Normal}} + b_n,$ $b_n = \begin{cases} b, n = v \times \eta, v = \{1, 2, \dots ,\}, \eta = \text{constt} \\ 0, \text{otherwise} \end{cases}$
Stuck	$s_n^{\text{Stuck}} = \alpha, \alpha = \text{constt}$

In dataset 1, the fault insertion point is kept fixed at index = 500 of each fault sample, illustrated in Figure 3. The

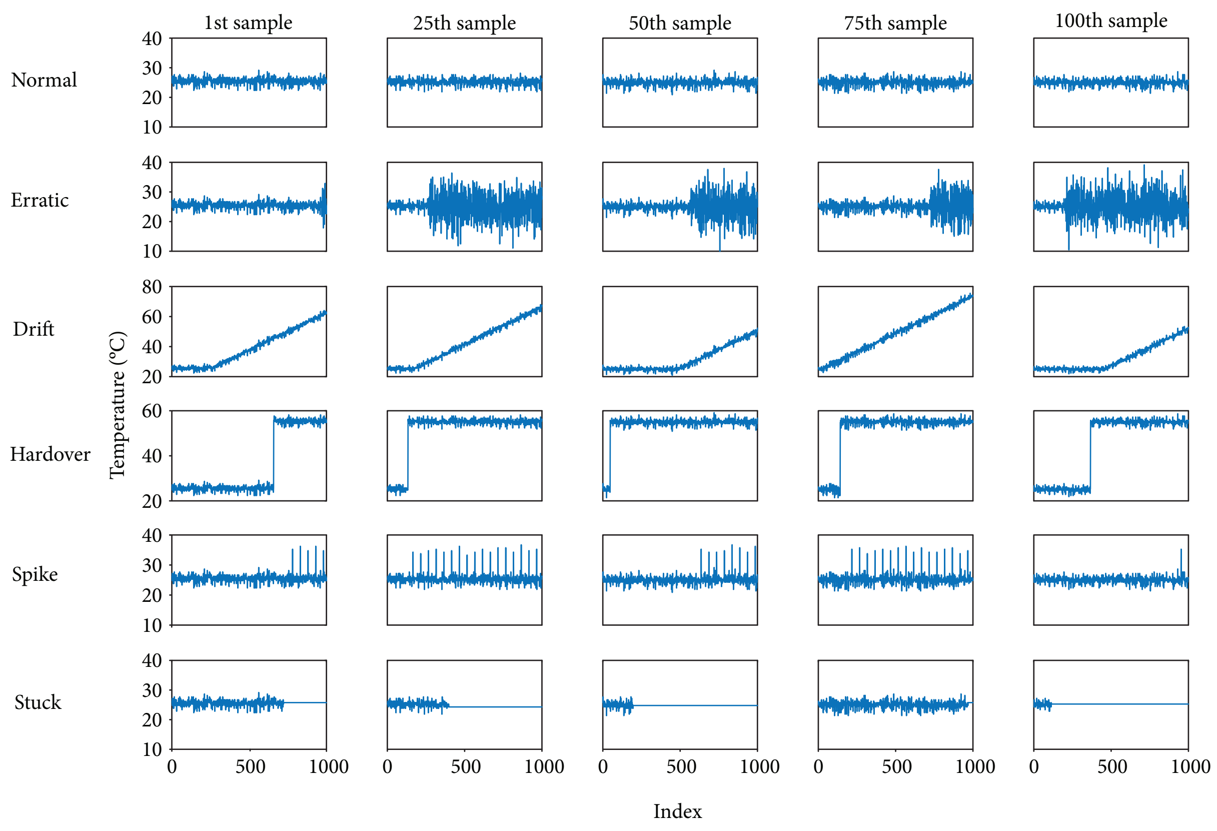

Figure 4: Plots of the 1st, 25th, 50th, 75th, and 100th sample of all six classes from dataset 2 (with a variable fault insertion point).

fault insertion point is the index of the element of the signal from where the fault occurrence starts. A similar way of fault simulation is adopted to obtain dataset 2 with the same size as the first dataset; however, the fault insertion point is a variable index, chosen randomly, of the signal. This case is considered to replicate a more practical scenario of the fault occurrence time in sensors. The time-domain plots of the 1st, 25th, 50th, 75th, and 100th sample from each class from the second dataset with a random fault insertion point are given in Figure 4.

4.2. Performance Evaluation Metrics. The performance of the classification models is analyzed and compared using the following metrics.

4.2.1. Accuracy. The accuracy of a classifier is the ratio of the number of true predictions to the total number of observations in the test set. Mathematically,

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{N}, \tag{23}$$

where TP and TN denote the number of true positive and true negative predictions and N is the total number of observations under the test.

4.2.2. Receiver Operating Characteristics. A receiver operating characteristic (ROC) graph is used to visualize, analyze, and select classifiers based on the performance. A two-dimensional graph represents a tradeoff between the true positive rate plotted on the y-axis and the false positive rate on the x-axis.

4.2.3. Area under the ROC. A comparison of two-dimensional ROC curves of two classifiers becomes difficult if the difference in the performances is too small. It may be needed to represent the performance with a scalar value. Calculating the area under the ROC curve (AUC-ROC) is a common technique used for this purpose [31].

4.3. Classifier Implementation. The SVM-based classifiers in both conventional and proposed models are trained and tested using MATLAB in a one-versus-rest manner. The cost and box constraints of misclassifications in the SVM are set to 1. The solution to the optimization problem is obtained using a sequential minimal optimization (SMO) algorithm.

The NN is composed of an input layer, three hidden layers, and an output layer. Unfortunately, there is no way to compute the optimized number of layers or the number of neurons per layer in the NN. However, the number of nodes in the input layer is equal to the number of input variables. We used 12, 6, and 2 nodes in the hidden layers with the hidden layer of 12 nodes being the nearest to the input layer. The output layer is composed of one node, giving output in the form of an integer number ranging from 1 to 6 corresponding to the normal, erratic, drift, hardover, spike, and stuck faults, respectively. A Levenberg-Marquardt (LM) algorithm is used in the training phase to optimize the weights of the neurons. The mean squared error (MSE) is utilized to update the weights of the network in backpropagation. A sigmoid function is utilized to activate the nodes in the hidden layers.

In the KNN classifier, the nearest 3 neighbors are weighted using the Euclidean distance parameter to classify a test observation. The cost of wrongly identifying an observation of a class in any other class is set to 1.

For all experiments, a 10-fold CV technique is applied in the training phase to improve the generalization of each classifier. Furthermore, a z-score standardization is applied in preprocessing of the test dataset to normalize the input vectors.

4.4. Experimental Results. To analyze the respective efficiency of the proposed feature selection scheme and the proposed DI-SVM model, a series of five experiments are performed. The first and second experiments are performed without applying a feature selection scheme in the fault detection and diagnosis procedure, using datasets 1 and 2, respectively. In these experiments, the performance comparison of the conventional SVM, NN, and KNN classifiers is presented. In experiment 3, the performance of the proposed feature selection scheme is analyzed and compared to the existing schemes. Lastly, the first two experiments are repeated after adding the proposed feature selection algorithm to the system to observe the effect on the performance of classifiers. The performance of the proposed DI-SVM model is also added in comparison with the conventional SVM, NN, and KNN in last two experiments.

4.4.1. Experiment 1: Fault Diagnosis without Feature Selection Using Dataset 1. The first experiment is performed without applying feature selection in the fault detection and classification methodology. The dataset contained 1000-dimensional normal and fault samples having a constant fault insertion point indexed at 500. All the features given in Tables 2 and 3 are extracted and input to the classifiers.

The accuracy results obtained from this experiment using the conventional SVM, NN, and KNN classifiers are given in Table 4. An individual accuracy of each class is obtained by assuming the given class as a positive class and the remaining classes as a set of negative classes. The total accuracy is the percent ratio of the sum of individual accuracies to the sum of the highest possible accuracies of all classes.

As given in the table, an SVM attains higher accuracies than NN and KNN for classifying all classes except for the normal and the stuck faults. The reason is the small difference between the range of features in normal and stuck fault classes. However, the SVM outperforms the NN and the KNN classifiers achieving a total accuracy of 81%. Furthermore, the NN performs worst among the three classifiers achieving a total accuracy of 78%, a difference of almost 1% from 79% achieved by the KNN classifier. However, the NN, as compared to SVM and KNN, distinctly outperforms in classifying the normal class. Moreover, a KNN classifier achieves slightly higher accuracy when classifying the stuck fault class than do NN and SVM.

Table 4: Accuracies (%) of classifiers obtained in experiment 1.

Classifier	Normal	Erratic	Drift	Hardover	Spike	Stuck	Total
SVM	43.05	85.55	92.50	95.83	85.83	83.33	81.01
NN	83.33	63.05	75.55	86.11	81.38	83.33	78.79
KNN	70.83	63.05	82.77	83.61	92.50	83.88	79.44

Table 5: Accuracies (%) of classifiers obtained in experiment 2.

Classifier	Normal	Erratic	Drift	Hardover	Spike	Stuck	Total
SVM	40.83	90.27	84.44	86.94	93.33	85.83	80.27
NN	83.33	60.00	71.11	82.22	77.22	83.33	76.20
KNN	53.33	81.66	65.00	83.05	80.27	83.33	74.44

4.4.2. Experiment 2: Fault Diagnosis without Feature Selection Using Dataset 2. In this experiment, the fault detection and classification methodology is similar to that of experiment 1; however, the dataset contains samples with variable fault insertion between 0 and 1000 index, as shown in Figure 4.

The results in Table 5 show that the samples considered in this case put a higher challenge to classifiers than those considered in the previous case. The total accuracy of all classifiers is degraded as compared to experiment 1. Nevertheless, the SVM still outperforms the NN and KNN classifiers. However, the total accuracy is degraded from 81% to around 80%. Similarly, the total accuracies of the NN and KNN classifier also reduced from 78% and 79% to 76% and 74%, respectively. The NN is successful in achieving an equal accuracy for normal and stuck fault classes, although a reduction is observed in the accuracies of other classes. The performance of the KNN classifier in classifying most of the classes is reduced with a considerable difference. On the other hand, an increased accuracy is reported for the erratic fault class for both SVM and KNN classifiers.

4.4.3. Experiment 3: Feature Selection. The mean values of 14 time-domain and 4 frequency-domain features extracted from the training samples are given in Table 6. The proposed FS scheme is supposed to select the exclusive most relevant features for each class among the given sets of features in the pool.

To obtain an optimal value of m for the FS scheme, the ROCs along with the respective AUC-ROC values are given in Figures 5, 6, 7, and 8. A conventional SVM classifier was used to obtain the discriminating efficiency of the classifier for the normal class in these figures. The results of Figure 5 are obtained using dataset 1. The normal class samples are trained as a positive class whereas the remaining class samples are trained as a single negative class. Figure 6 shows the results obtained using dataset 2 for a similar positive class, that is, normal class, and negative class. In Figures 7 and 8, datasets 1 and 2 are utilized, respectively; however, the set of faulty classes is trained as a positive class and the normal class as a negative class.

As illustrated in the figures, the classifier performs better when dataset 1 is used as compared to the counter case of

Table 6: Mean values of features of all samples.

Features	Normal		Erratic		Drift		Hardover		Spike		Stuck	
Fault insertion point	Dataset 1	Dataset 2	Dataset 1	Dataset 2	Dataset 1	Dataset 2	Dataset 1	Dataset 2	Dataset 1	Dataset 2	Dataset 1	Dataset 2
Mean	25.14	25.14	25.14	25.13	26.39	33.67	40.14	39.42	25.20	25.24	25.05	25.14
STD	0.97	0.97	2.98	2.88	1.88	8.02	15.03	11.97	1.24	1.38	0.81	0.76
VAR	0.95	0.95	8.93	9.00	3.57	86.64	226.08	151.99	1.55	1.96	0.71	0.67
RMS	795.07	795.07	795.16	794.95	834.68	1064.80	1269.41	1246.57	796.97	798.28	792.25	795.11
PPV	7.45	7.45	24.91	24.45	10.86	30.23	36.93	36.68	14.85	15.04	6.97	6.63
Sign	1	1	1	1	1	1	−1	−1	1	1	1	1
SRA	25.13	25.13	25.05	25.04	26.36	33.12	38.68	38.43	25.18	25.22	25.04	25.13
KV	4.82	4.82	5.40	6.82	2.77	3.19	1.01	5.99	27.67	25.67	7.37	23.48
SV	−0.3226	−0.3226	−0.0041	−0.0698	0.5293	0.6979	−0.0001	0.0942	2.9003	3.0904	−0.2370	−0.4364
CF	114	1.14	1.48	1.46	1.22	1.47	1.36	1.47	1.43	1.44	1.14	1.12
IF	1.14	1.14	1.49	1.47	1.23	1.52	1.45	1.55	1.44	1.44	1.14	1.12
MF	1.14	1.14	1.49	1.48	1.23	1.54	1.51	1.59	1.44	1.44	1.14	1.12
FF	1.0007	1.0007	1.0070	1.0070	1.0025	1.0286	1.06778	1.0513	1.0012	1.0015	1.0006	1.0004
KF	$2.62e-48$	$2.62e-48$	$2.10e-53$	$2.11e-50$	$1.14e-52$	$2.12e-49$	$8.24e-55$	$8.76e-53$	$9.53e-49$	$7.63e-49$	$1.11e-46$	$5.48e-43$
FC	52.50	52.50	108.91	106.05	59.67	104.61	130.43	135.85	59.35	60.47	45.20	43.87
SFFT	126.00	126.00	126.00	126.00	126.00	126.00	126.00	126.00	126.00	126.00	126.00	126.00
RMSF	795.67	795.67	800.76	800.58	836.81	1099.13	1355.46	1307.12	797.94	799.51	792.70	795.54
RVF	793.93	793.93	793.31	793.20	834.68	1094.06	1349.17	1299.89	795.73	797.21	791.41	794.30

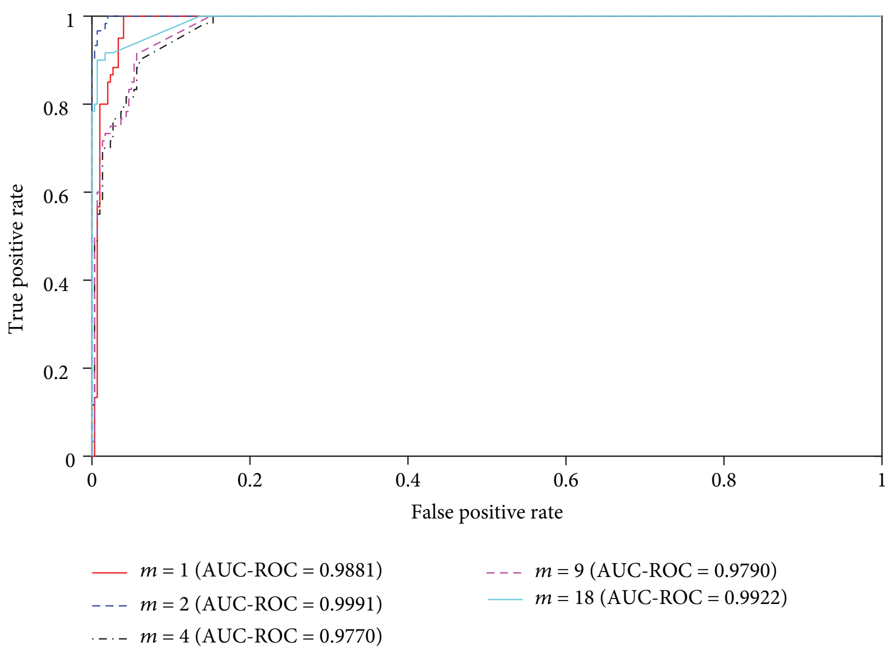

FIGURE 5: ROC and AUC-ROC comparison of the conventional SVM model for various numbers of features (m) selected by the feature selection scheme using dataset 1 when the normal class is selected as a positive class.

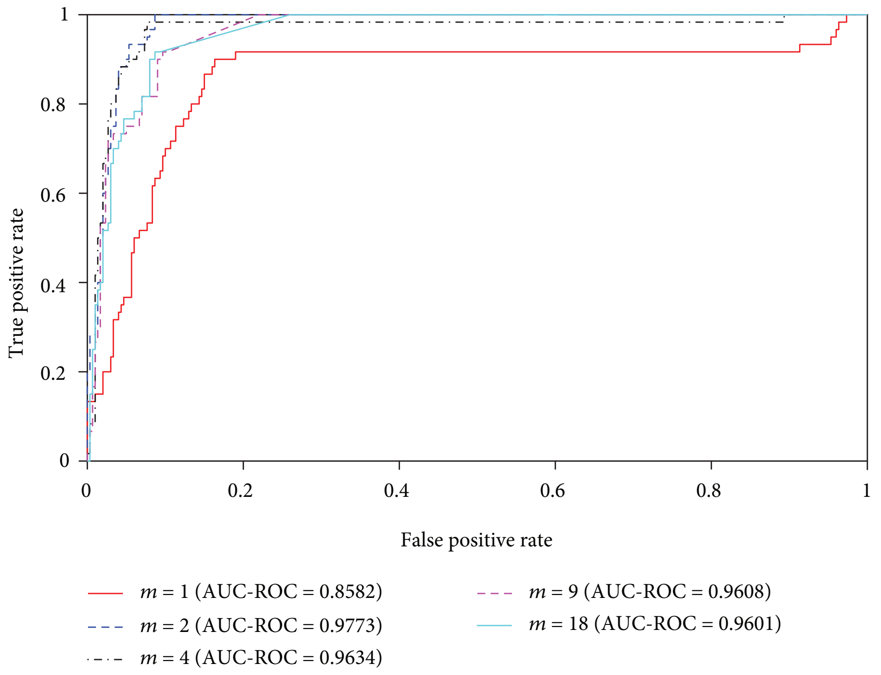

FIGURE 6: ROC and AUC-ROC comparison of the conventional SVM model for various numbers of features (m) selected by the feature selection scheme using dataset 2 when the normal class is selected as a positive class.

using dataset 2. These results show that it is more challenging for a classifier to perform classification when the dataset with a variable fault insertion point is used. Furthermore, it can be observed that $m = 2$ feature selection for the normal class gives the optimal results in all cases as compared to $m = 1$, 4, 9, and 18.

The respective AUC-ROCs for each class are given in Figure 9 using dataset 1 and Figure 10 using dataset 2. The AUC-ROC measures in these figures also illustrate that $m = 2$ is the optimal value for all classes with a given set of features and fault types. The hardover fault type has an almost similar value of AUC-ROC for $m = 1$ and $m = 2$.

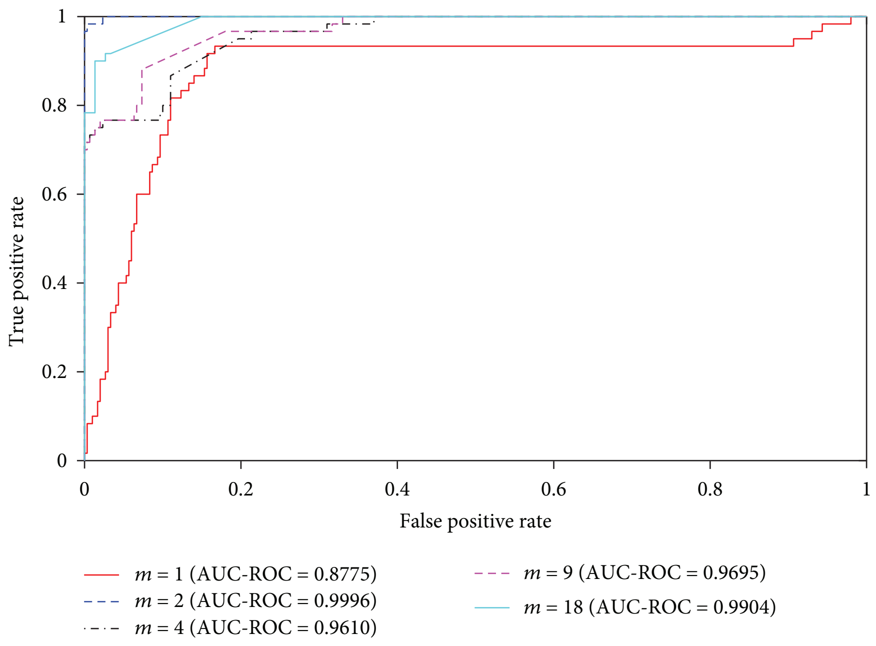

Figure 7: ROC and AUC-ROC comparison of the conventional SVM model for various numbers of features (m) selected by the feature selection scheme using dataset 1 when all fault classes are selected as a positive class.

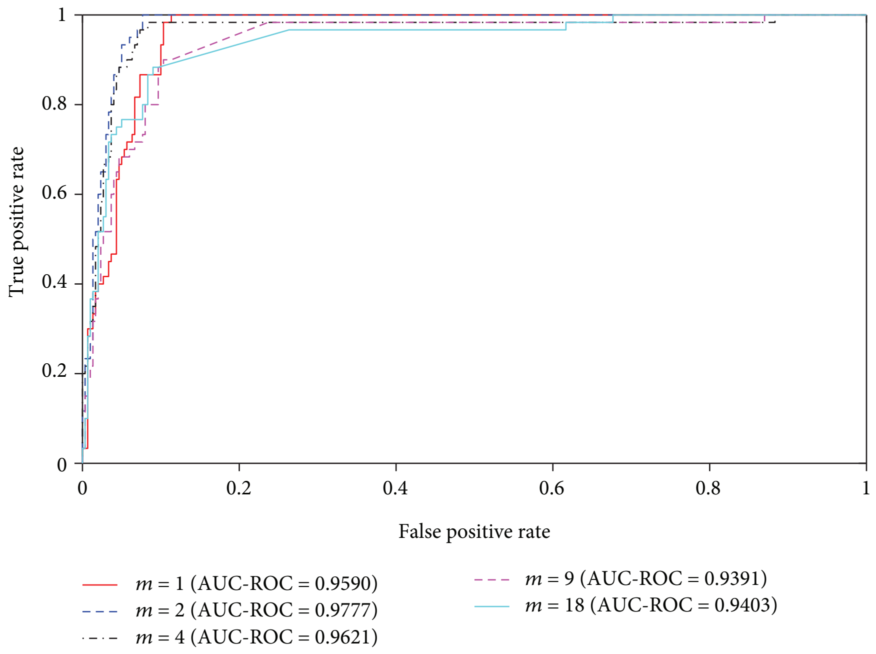

Figure 8: ROC and AUC-ROC comparison of the conventional SVM model for various numbers of features (m) selected by the feature selection scheme using dataset 2 when all fault classes are selected as a positive class.

The reason is that only a sign function is a good enough feature to discriminate the hardover fault class. A further increase in the number of feature selection m from 2 shows a degrading AUC-ROC measure for this class as well.

The efficiency of the proposed FS scheme is shown in Figures 11 and 12. The performance is compared with those of the MIFS, MIFS-U, mRMR, NMIFS, I-NMIFS, and exhaustive search algorithm. The exhaustive search algorithm uses all combinations of features in the feature pool to assess the performance of the classifier and then selects the best combination among all. The total accuracy obtained from the conventional SVM classifier is used as evaluation criteria. Datasets 1 and 2 are utilized in Figures 11 and 12, respectively.

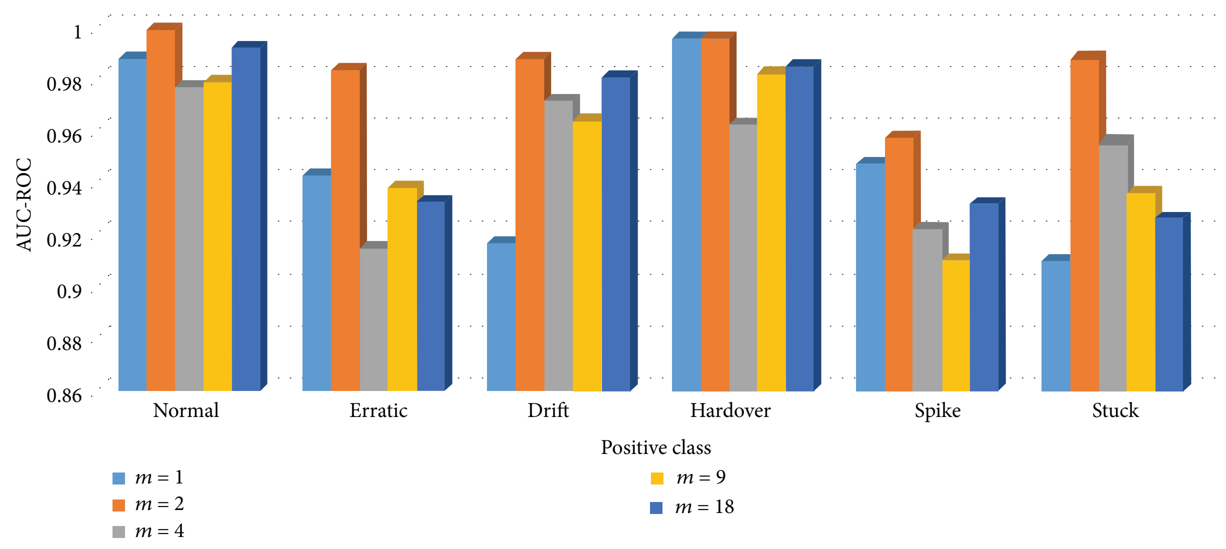

Figure 9: AUC-ROCs obtained for different values of m with respective positive classes using dataset 1.

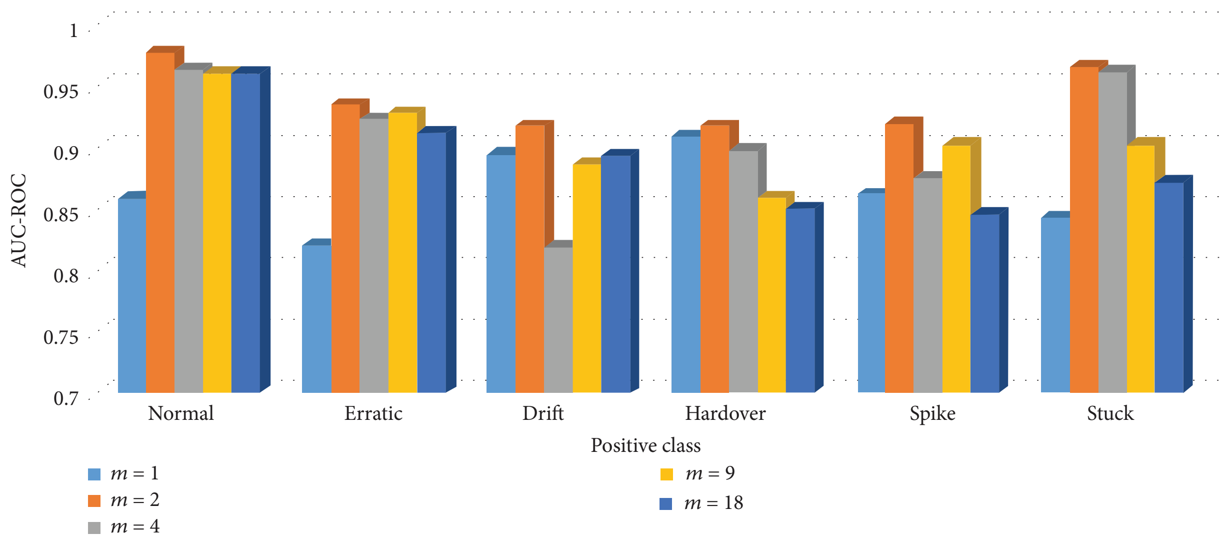

Figure 10: AUC-ROCs obtained for different values of m with respective positive classes using dataset 2.

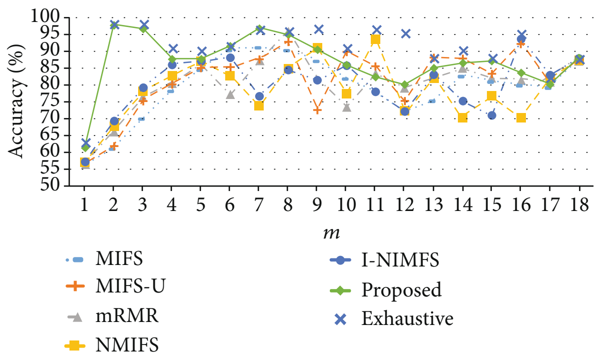

Figure 11: Accuracy comparison of feature selection schemes with different values of m and dataset 1.

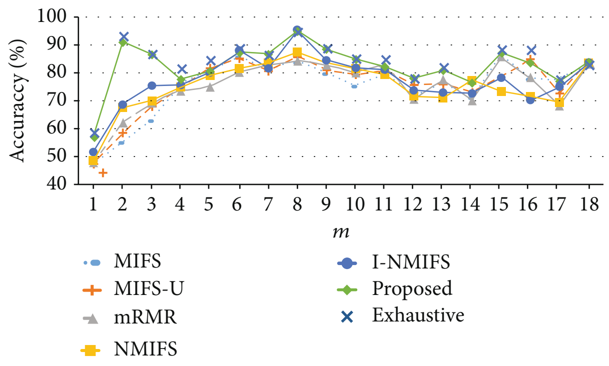

Figure 12: Accuracy comparison of feature selection schemes with different values of m and dataset 2.

Table 7: Maximum total accuracies (%) of respective FS with the number of features selected for all classes.

FS scheme	Dataset 1		Dataset 2	
	Number of selected features	Accuracy (%)	Number of selected features	Accuracy (%)
MIFS	12	95.34	13	89.98
MIFS-U	11	95.88	12	90.23
mRMR	10	96.00	11	90.02
NMIFS	10	96.55	10	90.92
I-NMIFS	9	97.01	10	91.00
Proposed	9	97.77	10	91.10
Exhaustive	8	98.02	10	91.98

Table 8: Features selected using the proposed feature selection algorithm for each class.

Class	Normal	Erratic	Drift	Hardover	Spike	Stuck
Selected features	CF	FC	FF	Sign	KV	FC
	FC	STD	Mean	RMS	SFFT	Mean

While selecting the first feature, all the abovementioned schemes have similar criteria of selecting a feature, that is, the measure of mutual information of the candidate feature with the target class. This results in a selection of the same feature by all FS schemes, hence giving comparable performance, as shown in both figures. To select the next features, the criteria of all FS schemes are almost the same, except for the proposed FS scheme. These features rely on both mutual information and redundancy measures, that is, assessing the mutual information of the candidate feature with the target class as well as the redundancy with the already-selected feature. This technique may sometime result in losing a good discriminating feature due to a high redundancy with the already-selected feature. Therefore, these schemes follow a similar trend of increase in the accuracy of the classifier with the increase in the number of the selected features. Furthermore, the selection of the next feature depends highly on the nature of the currently selected feature. It is possible that one of the already-selected features is not a good choice for the target class and it may result in losing a good feature in hand due to a high-redundancy factor. However, the figures show that the proposed FS scheme is able to achieve a good performance by selecting only the 2 most discriminating features per class from the feature pool. Furthermore, including all 18 features in the input to the classifier results in a similar performance irrespective of the FS scheme.

Furthermore, Table 7 shows the optimal accuracy obtained by the classifier with a given total number of features selected by different FS schemes. The results show that the proposed scheme can achieve the results almost comparable to the exhaustive search algorithm with the need of the lowest number of features compared to other techniques. For dataset 1, the proposed scheme can achieve optimal results by selecting only 9 features from the given feature pool. Similarly, 10 features are selected by the proposed FS scheme to obtain optimal results in the case of dataset 2.

The two features selected for each class by the proposed FS scheme are given in Table 8. To show the efficiency of the proposed feature selection algorithm, the scatter plots of all samples are shown in Figure 13 in the selected feature spaces. The red squares represent the data elements of the class for which the respective features are selected. The black circles illustrate the data points of the remaining classes in the same feature space. For instance, the two features selected for the normal class are crest factor (CF) and center frequency (FC). Therefore, the red squares represent the data samples of the normal class in Figure 13(a) in the CF-versus-FC feature space, whereas the black circles represent the data elements of all the remaining classes in this subfigure. A similar trend is followed in the all subfigures.

To observe the feature spaces CF-versus-FC and FC-versus-mean, the scatter plots are redrawn in Figures 14 and 15 with the data elements from normal, stuck, and spike fault classes only. The CF and FC features are selected targeting the normal class. Therefore, most of data elements of the normal class can be distinguished from the nearest classes, that is, spike and stuck fault classes, as shown in Figure 14. Similarly, the FC-versus-mean feature space targets the stuck fault class, and thus the data from the stuck fault class are easily discriminated from the normal and spike fault classes as compared to the CF-versus-FC space, as illustrated in Figure 15. This figure shows the efficiency of the proposed feature space graphically.

A value of $m = 2$ is used to obtain the results of experiments 4 and 5.

4.4.4. Experiment 4: Fault Diagnosis with Feature Selection Using Dataset 1.

This experiment is performed applying the proposed FS scheme in the classification methodology and using dataset 1. The performance of the proposed diversified-input (DI-) SVM model is compared with those of the conventional SVM, NN, and KNN classifiers.

The accuracy results of this experiment for all four classifiers are given in Table 9. The proposed DI-SVM achieves a 100% accuracy for erratic, drift, and hardover fault classes. The accuracy of classifying normal, spike, and stuck classes is slightly lower with 98.33%, 98.88%, and 99.44%,

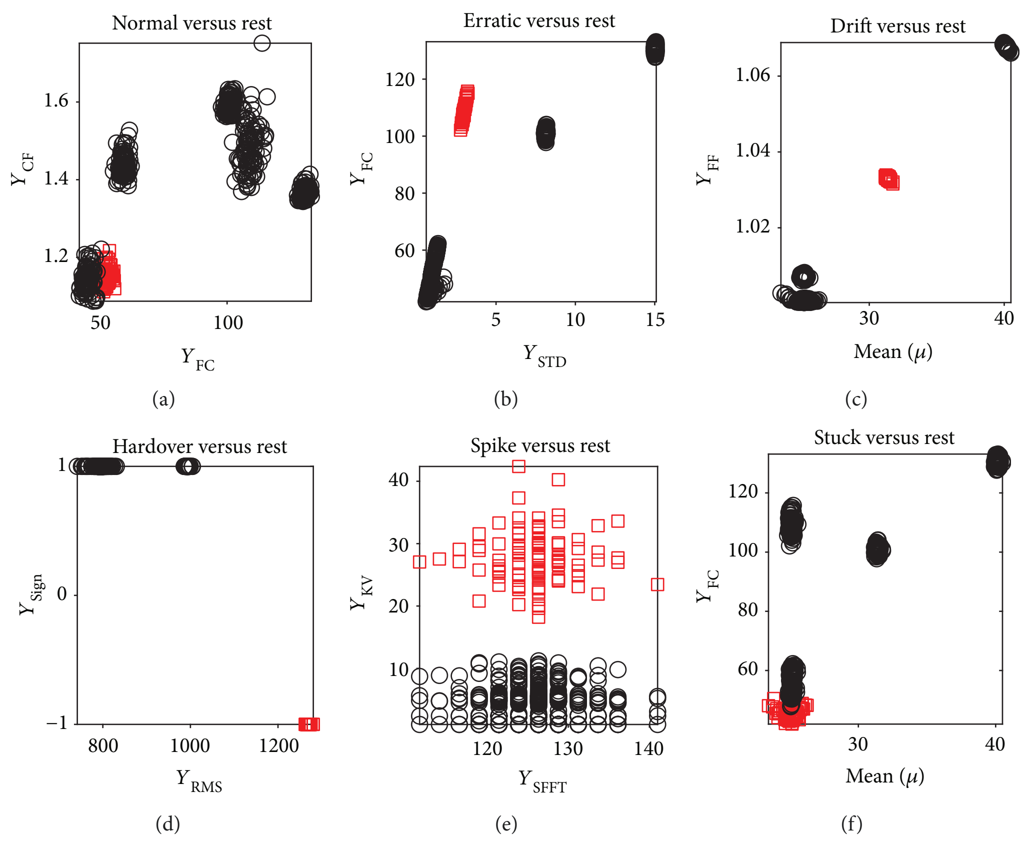

FIGURE 13: Scatter plot data from all classes in the respective feature spaces selected for each class by the feature selection algorithm.

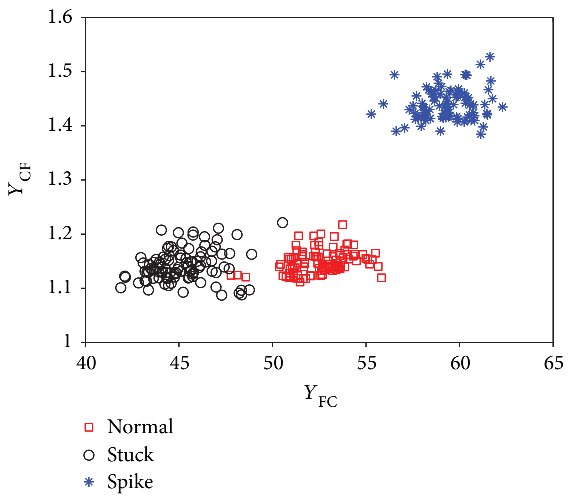

FIGURE 14: Scatter plots of normal, spike, and stuck fault samples in the CF-versus-FC feature space.

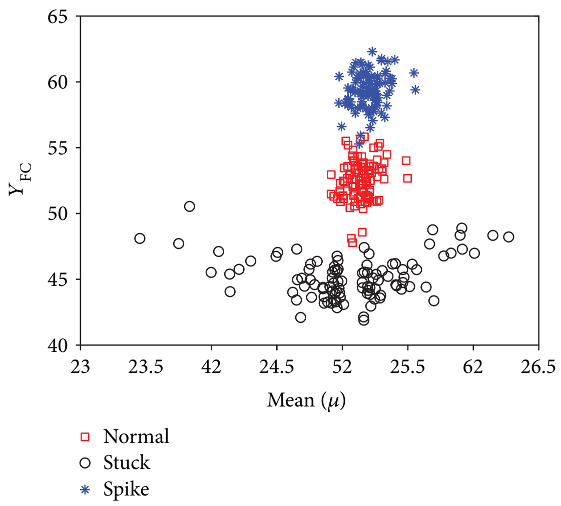

FIGURE 15: Scatter plots of normal, spike, and stuck fault samples in the FC-versus-mean feature space.

resulting in a total accuracy of 99.44% for DI-SVM. The conventional model of SVM is next on the list to perform efficiently achieving a total accuracy of 97.87%. The NN and KNN classifiers perform comparatively worse with a total accuracy of 94% and 85%, respectively.

Table 10 shows the AUC-ROC measures of the four classifiers for the same dataset. The individual AUC-ROC of each class is obtained from the ROC curve attained with the respective class trained as a positive class and the remaining classes trained as a set of negative classes. These results also show that the proposed DI-SVM model outperforms the conventional SVM, NN, and KNN classifiers.

To analyze the effect of the proposed FS scheme and the proposed DI-SVM model, the accuracy of the conventional SVM without applying the FS scheme, the conventional

Table 9: Accuracies (%) of classifiers obtained in experiment 4.

Classifier	Normal	Erratic	Drift	Hardover	Spike	Stuck	Total
DI-SVM	98.33	100.00	100.00	100.00	98.88	99.44	99.44
SVM	93.61	99.16	99.72	99.72	97.22	97.77	97.87
NN	98.61	98.33	98.05	93.33	88.61	90.27	94.53
KNN	83.33	92.77	74.72	83.33	98.05	83.33	85.92

Table 10: AUC-ROC of classifiers obtained in experiment 4.

Classifier	Normal	Erratic	Drift	Hardover	Spike	Stuck
DI-SVM	0.9995	1.0000	1.0000	1.0000	1.0000	0.9997
SVM	0.9994	1.0000	1.0000	1.0000	1.0000	0.9990
NN	0.9697	0.8100	0.6067	0.5966	0.8000	0.9897
KNN	0.9967	1.0000	1.0000	1.0000	1.0000	0.9832

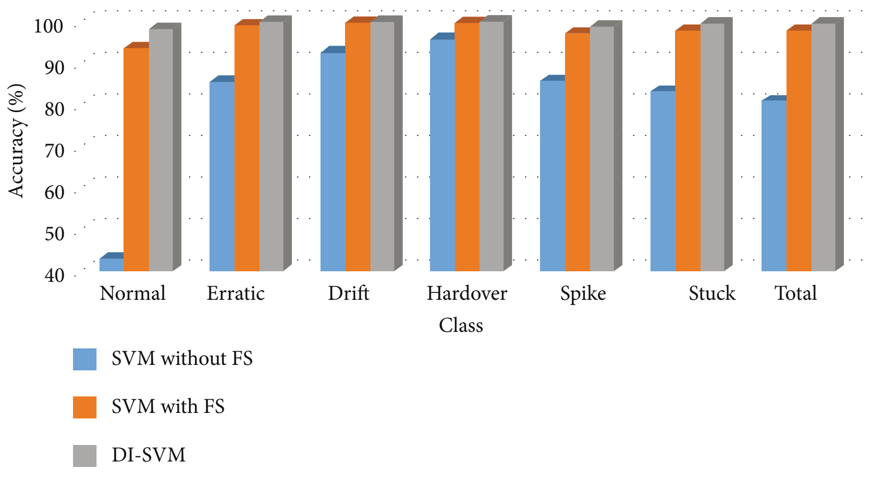

Figure 16: Accuracy comparison of the conventional SVM without FS, SVM with FS, and proposed DI-SVM schemes with the dataset of fixed fault insertion points.

SVM with applying the FS scheme, and the proposed DI-SVM model with FS applied using $m = 2$ is shown in Figure 16. The figure shows that the FS scheme successfully improves the performance of the conventional SVM classifier in classifying all classes. Furthermore, utilizing SVM in the proposed DI-SVM model is able to further improve the classifier's performance.

4.4.5. Experiment 5: Fault Diagnosis with Feature Selection Using Dataset 2. Experiment 5 is a repetition of experiment 4 with a difference in the dataset. In this experiment, dataset 2 is utilized to assess the performance of the proposed FS scheme and the classifiers. As shown in Table 11, the DI-SVM is successful in outperforming the counter three classifiers in classifying the classes in dataset 2 as well. However, the total accuracy is reduced to 93% in this case from 99% in the previous experiment. A similar trend of reduction in accuracy is observed for all classifiers. The intuitive reason is that dataset 2 with a variable fault

Table 11: Accuracies (%) of classifiers obtained in experiment 5.

Classifier	Normal	Erratic	Drift	Hardover	Spike	Stuck	Total
DI-SVM	82.22	99.16	93.32	98.05	99.17	92.21	93.89
SVM	73.89	95.83	92.21	95.82	98.32	89.71	91.10
NN	88.32	88.60	94.71	89.71	91.10	94.72	91.19
KNN	61.38	82.21	76.39	79.17	81.39	83.32	77.30

Table 12: AUC-ROC of classifiers obtained in experiment 5.

Classifier	Normal	Erratic	Drift	Hardover	Spike	Stuck
DI-SVM	0.9767	0.9925	0.9383	1.0000	0.9996	0.8820
SVM	0.9766	0.9829	0.8997	0.8948	0.9663	0.8676
NN	0.9223	0.7933	0.5845	0.6077	0.8103	0.8820
KNN	0.9422	0.9565	0.9913	1.0000	0.9983	0.9269

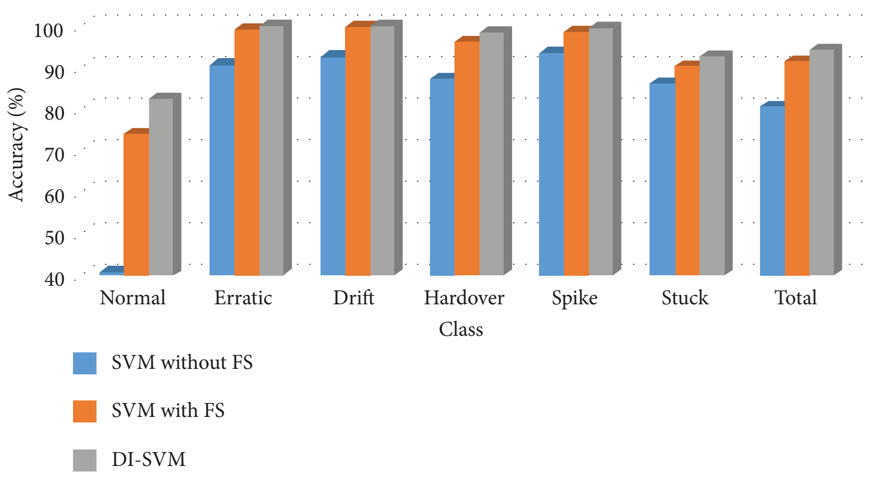

FIGURE 17: Accuracy comparison of the conventional SVM without FS, SVM with FS and proposed DI-SVM schemes with the dataset of variable fault insertion points.

insertion point put a higher challenge in the classification of samples as compared to the classification of samples in dataset 1 with a fixed fault insertion point.

The AUC-ROCs of this experiment are given in Table 12. Once again, the DI-SVM is successful in achieving the optimal AUC-ROC measures for all classes.

For experiment 5, the accuracy comparison of the conventional SVM without applying the FS scheme, the conventional SVM with the FS scheme, and the proposed DI-SVM model with FS applied using $m = 2$ is shown in Figure 17. The figure also strengthens our claim that the proposed DI-SVM model outperforms the conventional SVM model.

5. Discussion

The MI-based FS schemes select features for a given target class based on the measure of MI of the candidate feature with a target class and the mutual redundancy with already-selected features, irrespective of nontarget classes. The proposed FS algorithm considers the MI of a feature on both target and nontarget classes to select the best combination of features discriminating the target class from nontarget classes. As this feature selection scheme is regarded as a filter-based feature selection, the features are selected independently of the classifier performance and the nature of raw signals. Therefore, it is safe to claim that the proposed FS algorithm will efficiently select exclusive most relevant features for a target class among the pooled features in any application. However, it should be noted that the efficiency of the classifier highly depends on the nature of features in the feature pool used to characterize the signal. Using the most characterizing set of features will increase the efficiency of the system and vice versa. From the results of our work, it can be concluded that taking into account the measurement of relevance of the candidate feature with a target class as well as with a nontarget class can help in selecting exclusive discriminating features.

Furthermore, in a conventional model of applying SVMs for multiclass classification, the features selected for each class are concatenated to train all SVMs. Intuitively, reducing the dimension of input observations leads to improved performance of SVMs given that the features are selected properly. Keeping this in mind, we reduced the dimension of the input vector of all SVMs by inputting diverse feature combinations selected for different classes. This means that the feature combination chosen for targeting the c^{th} class is input to the c^{th} SVM, particularly, trained with the data in which the observations of the c^{th} class are labeled as a positive class and vice versa. In this way, the dimension of the input vectors is reduced to simplify the optimization of the classifier. Moreover, similar feature combinations might be selected for more than one class if this combination is the most discriminating feature combination for each class.

Our previous work [3] proved from experimental results that increasing the size of a raw signal, that is, increasing the number of data elements per one raw signal of the sensor output, and increasing the size of the training dataset (increasing the number of training samples) increase the accuracy of the classifier. Therefore, the purpose of selecting a small size for training and testing sets is to observe the lower bound performance of the proposed fault diagnosis methodology.

6. Conclusions

To select the discriminating feature combinations targeting each class, we propose a feature selection scheme that takes into account the measure of relevance of the feature with both target and nontarget classes. The relevance of two variables is characterized by the measure of mutual information between them. The set of m features having the maximum mutual information on the target class and the minimum mutual information on the nontarget classes is chosen for a target class. The results show that this technique can achieve good results as compared to other feature selection schemes that take mutual redundancy measure between the selected features in addition to relevance with a target class.

Furthermore, a DI-SVM model is proposed to improve the performance of the classifier. Each SVM is trained with a single feature combination selected, specifically, for a single target class, unlike the conventional SVM model of delivering all combinations to all SVMs. This approach reduces the complexity of classifier optimization by minimizing the dimensions of the input vectors. The performance of the classifier was analyzed using two different datasets: (1) with a constant fault insertion point and (2) with a variable fault insertion point. A series of 5 experiments were performed to analyze the performance of the proposed fault diagnosis methodology. The evaluation metrics including the accuracy, the ROC, and the AUC-ROC show the efficiency of the proposed FS scheme and proposed DI-SVM model. A further comparison of the performances shows the efficiency of the proposed DI-SVM over the conventional model of SVM-, NN-, and KNN-based classifiers in multiclass sensor fault classification problems.

Data Availability

The simulated sensor fault data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Business for Cooperative R&D between Industry, Academy, and Research Institute funded by the Korean Small and Medium Business Technology Administration in 2016 under Grant C0398156.

References

[1] B. P. Duong and J.-M. Kim, "Non-mutually exclusive deep neural network classifier for combined modes of bearing fault diagnosis," *Sensors*, vol. 18, no. 4, p. 1129, 2018.

[2] M. Sohaib, C.-H. Kim, and J.-M. Kim, "A hybrid feature model and deep-learning-based bearing fault diagnosis," *Sensors*, vol. 17, no. 12, p. 2876, 2017.

[3] S. U. Jan, Y.-D. Lee, J. Shin, and I. Koo, "Sensor fault classification based on support vector machine and statistical time-domain features," *IEEE Access*, vol. 5, pp. 8682–8690, 2017.

[4] Y. Yu, W. Li, D. Sheng, and J. Chen, "A novel sensor fault diagnosis method based on modified ensemble empirical mode decomposition and probabilistic neural network," *Measurement*, vol. 68, pp. 328–336, 2015.

[5] S. U. Jan and I. S. Koo, "Sensor faults detection and classification using SVM with diverse features," in *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 576–578, Jeju, South Korea, October 2017.

[6] P. Huang, Y. Jin, D. Hou et al., "Online classification of contaminants based on multi-classification support vector machine using conventional water quality sensors," *Sensors*, vol. 17, no. 3, p. 581, 2017.

[7] B. Scholkopf, "Support vector machines: a practical consequence of learning theory," *IEEE Intelligent Systems*, vol. 13, 1998.

[8] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.

[9] L. Ren, W. Lv, S. Jiang, and Y. Xiao, "Fault diagnosis using a joint model based on sparse representation and SVM," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 10, pp. 2313–2320, 2016.

[10] S. S. Y. Ng, P. W. Tse, and K. L. Tsui, "A one-*versus*-all class binarization strategy for bearing diagnostics of concurrent defects," *Sensors*, vol. 14, no. 1, pp. 1295–1321, 2014.

[11] P. Thanh Noi and M. Kappas, "Comparison of random forest, k-nearest neighbor, and support vector machine classifiers for land cover classification using sentinel-2 imagery," *Sensors*, vol. 18, no. 2, p. 18, 2018.

[12] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2392–2431, 2017.

[13] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[14] G. Ditzler, R. Polikar, and G. Rosen, "A sequential learning approach for scaling up filter-based feature subset selection," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2530–2544, 2018.

[15] M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," *Pattern Recognition*, vol. 33, no. 1, pp. 25–41, 2000.

[16] Z. M. Hira and D. F. Gillies, "A review of feature selection and feature extraction methods applied on microarray data," *Advances in Bioinformatics*, vol. 2015, Article ID 198363, 13 pages, 2015.

[17] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 2012.

[18] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 537–550, 1994.

[19] N. Kwak and C.-H. Choi, "Input feature selection for classification problems," *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 143–159, 2002.

[20] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada, "Normalized mutual information feature selection," *IEEE Transactions on Neural Networks*, vol. 20, no. 2, pp. 189–201, 2009.

[21] L. T. Vinh, S. Lee, Y.-T. Park, and B. J. d'Auriol, "A novel feature selection method based on normalized mutual information," *Applied Intelligence*, vol. 37, no. 1, pp. 100–120, 2012.

[22] G. Lera and M. Pinzolas, "Neighborhood based Levenberg-Marquardt algorithm for neural network training," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1200–1203, 2002.

[23] J. Yang, Z. Sun, and Y. Chen, "Fault detection using the clustering-kNN rule for gas sensor arrays," *Sensors*, vol. 16, no. 12, p. 2069, 2016.

[24] S. Haykin, *Neural Networks, a Comprehensive Foundation*, Tech. Rep., Macmilan, 1994.

[25] J. Weston and C. Watkins, “Support vector machines for multi-class pattern recognition,” *Proceedings of European Symposium on Artificial Neural Networks ESANN'99*, 1999, Brugs, Belgium, 1999.

[26] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[27] Z. Wang and X. Xue, “Multi-class support vector machine,” in *Support Vector Machines Applications*, pp. 23–48, Springer, 2014.

[28] D.-C. Luor, “A comparative assessment of data standardization on support vector machine for classification problems,” *Intelligent Data Analysis*, vol. 19, no. 3, pp. 529–546, 2015.

[29] C.-W. Chu, J. D. Holliday, and P. Willett, “Effect of data standardization on chemical clustering and similarity searching,” *Journal of Chemical Information and Modeling*, vol. 49, no. 2, pp. 155–161, 2009.

[30] J. Kullaa, “Detection, identification, and quantification of sensor fault in a sensor network,” *Mechanical Systems and Signal Processing*, vol. 40, no. 1, pp. 208–221, 2013.

[31] T. Fawcett, “An introduction to roc analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.