Learning in the Model Space for Cognitive Fault Diagnosis

Huanhuan Chen, *Member, IEEE*, Peter Tiňo, Ali Rodan, and Xin Yao, *Fellow, IEEE*

***Abstract*— The emergence of large sensor networks has facilitated the collection of large amounts of real-time data to monitor and control complex engineering systems. However, in many cases the collected data may be incomplete or inconsistent, while the underlying environment may be time-varying or unformulated. In this paper, we develop an innovative cognitive fault diagnosis framework that tackles the above challenges. This framework investigates fault diagnosis in the model space instead of the signal space. Learning in the model space is implemented by fitting a series of models using a series of signal segments selected with a sliding window. By investigating the learning techniques in the fitted model space, faulty models can be discriminated from healthy models using a one-class learning algorithm. The framework enables us to construct a fault library when unknown faults occur, which can be regarded as cognitive fault isolation. This paper also theoretically investigates how to measure the pairwise distance between two models in the model space and incorporates the model distance into the learning algorithm in the model space. The results on three benchmark applications and one simulated model for the Barcelona water distribution network confirm the effectiveness of the proposed framework.**

***Index Terms*— Cognitive fault diagnosis, fault detection, learning in the model space, one class learning, reservoir computing (RC).**

I. Introduction

THE smooth operation of complex engineering systems is crucial to modern society. To ensure reliability, safety, and availability of such complex systems, large amounts of real-time data need to be collected to detect and diagnose faults as soon as possible. Therefore, designing intelligent real-time systems for fault diagnosis has been receiving considerable attention from both industry and academia.

The fault diagnosis procedure can be investigated in the following three steps: 1) fault detection, which is aimed at determining whether a fault has occurred or not; 2) fault isolation, which aims to determine the type/location of fault; and 3) fault identification, which estimates the magnitude or severity of the fault. In some cases, the issues of fault isolation and fault identification are interwoven, since they both determine the type of fault that has occurred.

In recent years, there has been a lot of research in the design and analysis of fault diagnosis schemes for different dynamic systems (e.g., [1], [2]). A significant part of the research has focused on linear dynamic systems, where it is possible to obtain rigorous theoretical results. More recently, considerable effort has been devoted to the development of fault diagnosis schemes for nonlinear systems with various kinds of assumptions and fault scenarios [3]–[5].

These traditional fault diagnosis approaches rely, to a large degree, on the mathematical model of the “normal” system. If such a mathematical model is available, then fault diagnosis can be achieved by comparing actual observations with the prediction of the model. Most autonomous fault diagnosis algorithms are based on this methodology. However, for complex engineering systems operating in unformulated or time-varying environments, such mathematical models may not be accurate or even unavailable at all. Therefore, it is necessary to develop cognitive fault diagnosis methods mainly based on the collected real-time data.

In this paper, we present a novel framework for dealing with fault detection to fault isolation if no or very limited knowledge is available about the underlying system. We do not assume that we know the type, number, or functional form of the faults in advance. The core idea is to transform the signal into a higher dimensional “dynamical feature space” via reservoir computation models and then represent varying aspects of the signal through variation in the linear readout models trained in such dynamical feature spaces. In this way, parts of the signal captured in a sliding window will be represented by the reservoir model with the readout mapping fitted in that window.

Dynamic reservoirs of reservoir models have been shown to be “generic” in the sense that they are able to represent a wide variety of dynamical features of the input-driven signals, so that, given a task at hand, only the linear readout on top of reservoir needs to be retrained [6]. Hence in our formulation, the underlying dynamic reservoir will be the same throughout the signal—the differences in the signal characteristics at different times will be captured solely by the linear readout

Manuscript received May 21, 2012; accepted March 17, 2013. This work was supported by the European Union Seventh Framework Programme under Grant Agreement INSFOICT-270428 on “Making Sense of Nonsense (iSense).” The work of H. Chen was supported in part by the National Natural Science Foundation of China under Grant 61203292 and the One Thousand Young Talents Program. The work of P. Tiňo was supported by the Biotechnology and Biological Sciences Research Council under Grant H012508/1. The work of X. Yao was supported by a Royal Society Wolfson Research Merit Award.

H. Chen is with the Centre of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K., and also with the USTC-Birmingham Joint Research Institute in Intelligent Computation and Its Applications, School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: H.Chen@cs.bham.ac.uk).

P. Tiňo and X. Yao are with The Centre of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: P.Tino@cs.bham.ac.uk; X.Yao@cs.bham.ac.uk).

A. Rodan is with the King Abdulla II School for Information Technology, The University of Jordan, Amman 11942, Jordan (e-mail: A.Rodan@ju.edu.jo).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2013.2256797

models and will be quantified in the function space of readout models.

We assume that, for some sufficiently long initial period, the system is in a "normal/healthy" regime so that when a fault occurs the readout models characterizing the fault will be sufficiently "distinct" from the normal ones. A variety of novelty/anomaly detection techniques can be used for the purposes of detection of deviations from the "normal." In this paper, we will use the one-class support vector machine (OCS) [7] methodology in the readout model space. As new faults occur in time, they will be captured by our incremental fault library building algorithm operating in the readout model space.

There have been other learning-based approaches on fault detection and diagnosis, e.g., [8]–[12]. In [10], when a neural network is expanded or the topology of the network is changed to accommodate new faults or unexpected dynamics, the network should be retrained. Later on, Barakat et al. [13] proposed the use of a self-adaptive growing neural network for fault diagnosis. They applied wavelet decomposition and used the variance and Kurtosis of the decomposed signals as features. In 2009, Yélamos *et al.* [14] proposed the use of support vector machines (SVMs) for fault diagnosis in chemical plants. Recently, Zhang *et al.* [15] proposed a data-core-based fuzzy min–max neural network (DCFMN) for pattern classification. A new fuzzy membership has been incorporated in the DCFMN to consider the characteristics of the data and the influence of noise. Barua *et al.* [16] developed a hierarchical fault-diagnosis methodology that decomposes a complex system hierarchically into simpler nodes based on "prior" knowledge about the systems and faults, deploys fault-diagnosis algorithms at these nodes, and fuses the results from different nodes on the basis of a Bayesian-network-based component dependency model. Crucially, most of the current learning-based approaches are formulated in the supervised learning framework, assuming that all fault patterns are known in advance. This can clearly be unrealistic.

The contributions of this paper are as follows: 1) we propose a novel learning framework for cognitive fault diagnosis; 2) the framework is based on learning in the model space (as opposed to the traditional data space) of readout models operating on the dynamic reservoir feature space representing parts of signals; and 3) we propose the use of incremental one-class learning in the readout model space for fault detection/isolation and dynamic fault library building.

The rest of this paper is organized as follows. Section II introduces deterministic reservoir computing and the framework of "learning in the model space," followed by the incremental one-class learning algorithm for cognitive fault diagnosis in Section III. The experimental results and analysis are reported in Section IV. Finally, Section V concludes the paper and presents some future work.

II. Deterministic Reservoir Computing and Learning in the Model Space

This section introduces the deterministic reservoir model to fit multiple-input multiple-output (MIMO) signals. Then, we introduce the framework of learning in the model space for fault diagnosis.

A. Deterministic Reservoir Computing

Reservoir computing (RC) [6] is a recent class of state space models based on a "fixed" randomly constructed state transition mapping, realized through a so-called reservoir and a trainable (usually linear) readout mapping from the reservoir. Popular RC methods include echo state networks (ESNs) [17], [18], liquid state machines [19], and the backpropagation decorrelation neural network [20].

In this paper, we will focus on ESNs. ESNs are one of the simplest yet effective forms of RC. Generally speaking, ESNs are recurrent neural networks with a nontrainable sparse recurrent part (reservoir) and a simple linear readout. Typically, the reservoir connection weights as well as the input weights are randomly generated, subject to the "echo state property" [17].

The traditional randomized RC is largely driven by a series of randomized model building stages, which could be unstable and hard to understand, especially for fault diagnosis. In this paper, we propose the use of the deterministic reservoir algorithm, i.e., a simple cycle topology with regular jumps (CRJs) [21], to fit the signals for fault diagnosis, since CRJs can approach any nonlinear mapping with arbitrary accuracy. Because of the linear training, the CRJ model can be trained fast and run in real time.

We deliberately use reservoir models (especially CRJ)—special cases of recurrent neural networks (RNNs)—for several reasons.

1) Training of RNNs on time-series segments would require long segment lengths. Training itself can be complicated, time consuming, and hampered by local optima in the error surface [22], [23].
2) Reservoir models have been shown to be competitive with RNNs on many datasets [23], except for specialized cases, e.g., linguistic data, where more intricate memory structures (e.g., periodic orbits for certain inputs) have to be induced during training through a series of bifurcations.
3) Reservoir models can be trained extremely fast and without local optima (training is only required for linear readouts).
4) The CRJ reservoir architecture has been shown to be competitive or superior to conventional randomly constructed reservoirs while being simple and transparent [21], [24].

B. Learning in the Model Space

Recently, there is a new trend in the machine learning community to represent "local" data collections through models that capture what is thought to be important in the data and carry out machine learning on those models—this can have the benefit of more robust and more targeted learning on diverse data collections [25].

The idea of learning in the model space is to use models fitted on parts of data as more stable and parsimonious representations of the data. Learning is then performed directly in the model space instead of the original data space. Some aspects of the idea of learning in the model space have

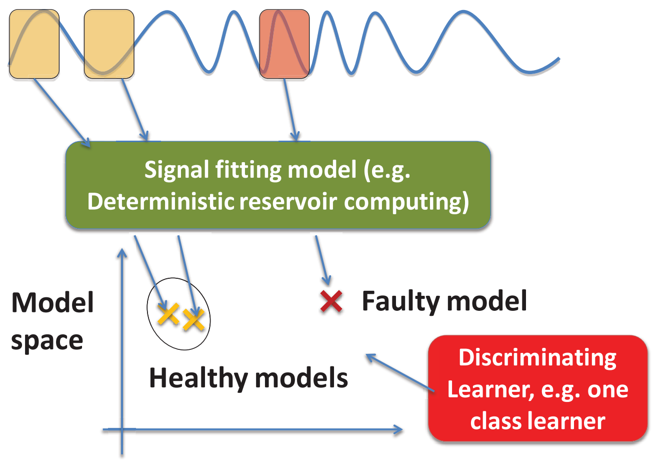

Signal fitting model (e.g. Deterministic reservoir computing)
Model space
Faulty model
Healthy models
Discriminating Learner, e.g. one class learner

Fig. 1. Illustration of "learning in the model space" framework. The first stage is to fit models using the input–output signal, i.e., generate individual points in the model space. The second stage is to discriminate the faulty models from healthy models using discriminating learners.

appeared in different forms in the machine learning community. For example, using generative kernels for classification (e.g., the P-kernel [26] or the Fisher kernel [27]) can be viewed as a form of learning in a model-induced feature space (see [28], [29]). Recently, Brodersen *et al.* [25] used a generative model of brain imaging data to represent functional magnetic resonance imaging (fMRI) measurements of different subjects to build an SVM-type learner to classify these subjects into aphasic patients or healthy controls.

In this paper, we use "learning in the model space" approach to represent chunks of signals by dynamic models (reservoirs models with linear readout) and perform learning in the models space of readouts. The framework is illustrated in Fig. 1.

1) Distance in the Model Space: There are several ways to generate the model space from the original signal space. One possible way is to identify parameterized models with their parameter vectors and work in the parameter space. This, however, will make the learning highly dependent on the particular model parameterization used. A more satisfactory approach is to use parameterization-free notions of distance or similarities between the models.

In the model space, the m-norm distance between models $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ $(f_1, f_2 : \Re^N \to \Re^O)$ is defined as follows:

$$L_m(f_1, f_2) = \left(\int_C D_m\left(f_1(\mathbf{x}), f_2(\mathbf{x})\right) d\mu(\mathbf{x})\right)^{1/m}$$

where $D_m\left(f_1(\mathbf{x}), f_2(\mathbf{x})\right) = \|f_1(\mathbf{x}) - f_2(\mathbf{x})\|^m$ is a function that measures the difference between $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$, $\mu(x)$ is the probability density function of the input domain $\mathbf{x}$, and C is the integral range. In this paper, we adopt $m = 2$ and first assume that x is uniformly distributed. Of course, nonuniform $\mu(\mathbf{x})$ can be adopted either by using samples generated from it or by estimating it directly using, e.g., Gaussian mixture models.

In the following, we demonstrate the application of the distance definition in the model space for linear readout models. The readout model can be represented by the following equation[1]:

$$f(\mathbf{x}) = W\mathbf{x} + \mathbf{a} \tag{1}$$

where $\mathbf{x} = [x_1, \ldots, x_N]^T$ is a state vector or basis function, N is the number of input variables in the model, W is the parameters ($O \times N$ matrix) in the model, O is the output dimensionality, and $\mathbf{a} = [a_1, \ldots, a_o] \in \Re^O$ is the bias vector of output nodes.

Consider two readouts from the same reservoir

$$\begin{aligned} f_1(\mathbf{x}) &= W_1\mathbf{x} + \mathbf{a}_1 \\ f_2(\mathbf{x}) &= W_2\mathbf{x} + \mathbf{a}_2. \end{aligned}$$

Since the sigmoid activation function is employed in the domain of the readout, $C \in [-1, 1]^N$. Then

$$\begin{aligned} L_2(f_1, f_2) &= \left(\int_C \|f_1(\mathbf{x}) - f_2(\mathbf{x})\|^2 d\mathbf{x}\right)^{1/2} \\ &= \left(\int_C \|(W_1 - W_2)\mathbf{x} + (\mathbf{a}_1 - \mathbf{a}_2)\|^2 d\mathbf{x}\right)^{1/2} \\ &= \left(\int_C \|W\mathbf{x}\|^2 + 2\mathbf{a}^T W\mathbf{x} + \|\mathbf{a}\|^2 d\mathbf{x}\right)^{1/2} \end{aligned}$$

where $W = W_1 - W_2$, and $\mathbf{a} = \mathbf{a}_1 - \mathbf{a}_2$.

Note that for any fixed $\mathbf{a}$ and W

$$\int_C \mathbf{a}^T W\mathbf{x}\, d\mathbf{x} = 0$$

in the integral range C.

Therefore

$$\begin{aligned} L_2(f_1, f_2) &= \left(\int_C \|W\mathbf{x}\|^2 + \|\mathbf{a}\|^2 d\mathbf{x}\right)^{1/2} \\ &= \left(\int_C \sum_{i=1}^{O} \left(\mathbf{w}_i^T \mathbf{x}\right)^2 + \|\mathbf{a}\|^2 d\mathbf{x}\right)^{1/2} \\ &= \left(\frac{2^N}{3} \sum_{j=1}^{N} \sum_{i=1}^{O} w_{i,j}^2 + 2^N \|\mathbf{a}\|^2\right)^{1/2} \end{aligned} \tag{2}$$

where $\mathbf{w}_i^T$ is the ith row of W, and $w_{i,j}$ is the (i, j)th element of W.

Scaling of the squared model distance $(L_2^2(f_1, f_2))$ by 2^{-N}, we obtain

$$\frac{1}{3} \sum_{j=1}^{N} \sum_{i=1}^{O} w_{i,j}^2 + \|\mathbf{a}\|^2$$

which differs from the squared Euclidean distance of the readout parameters

$$\sum_{j=1}^{N} \sum_{i=1}^{O} w_{i,j}^2 + \|\mathbf{a}\|^2$$

by the factor $1/3$ applied to the differences in the linear part W of the affine readouts. Hence, more importance is given to the "offset" than "orientation" of the readout mapping.

[1]The dynamic reservoir is fixed throughout the time series—the differences in the signal characteristics at different times will be captured solely by the linear readout model [see (1)] and will be quantified in the function space of readout models. The L_2 distance between linear readouts operates on the reservoir state space $[-1, 1]^N$.

Assume f is the true or the best reservoir readout for the given data, and consider another readout f_1. Consider two situations (see Fig. 2): 1) f and f_1 share the same orientation, but differ in the offset and 2) f and f_1 have the same offset but differ in the orientation. On the symmetric state space domain $[-1, 1]^N$, case 1 introduces more bias in predictive/modeling capabilities of the reservoir model than case 2. In case 1, the output will be constantly over- (or under-) estimating the target, while in case 2 the biggest disproportions will occur only close to the faces of the state space hypercube. This is a consequence of using the function L_2 distance on the readout model space.

In the above, we assumed that the distribution of $\mathbf{x}$ is uniform in the integral range C. As mentioned before, in case of nonuniform $\mu(\mathbf{x})$, we can either use samples generated from μ or estimate it analytically using, e.g., a Gaussian mixture model.

Assume we have m sampled points $\mathbf{x}_i$, $i = 1, 2, \ldots, m$ from μ

$$L_2(f_1, f_2) = \left(\int_C \|f_1(\mathbf{x}) - f_2(\mathbf{x})\|^2 \, d\mu(\mathbf{x})\right)^{1/2} \approx \left(\frac{1}{m}\sum_{i=1}^{m} \|f_1(\mathbf{x}_i) - f_2(\mathbf{x}_i)\|^2\right)^{1/2}. \quad (3)$$

Alternatively, a Gaussian mixture model can be employed to represent μ

$$\mu(\mathbf{x}) = \sum_{i=1}^{K} \alpha_i \mu_i(x|\boldsymbol{\eta}_i, \Sigma_i), \quad \text{and}$$

$$\mu_i(x|\boldsymbol{\eta}_i, \Sigma_i) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\eta}_i)^T \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\eta}_i)\right)}{(2\pi)^{N/2} |\Sigma_i|^{1/2}}$$

where $\sum_{i=1}^{K} \alpha_i = 1$ and N is the dimensionality of $\mathbf{x}$.

Then, the distance $L_2(f_1, f_2)$ can be obtained as follows:

$$\begin{aligned} L_2&(f_1, f_2) \\ &= \left(\int_C \|f_1(\mathbf{x}) - f_2(\mathbf{x})\|^2 \, d\mu(\mathbf{x})\right)^{1/2} \\ &= \left(\int_C \|W\mathbf{x} + \mathbf{a}\|^2 \, d\mu(\mathbf{x})\right)^{1/2} \\ &= \left(\int_C \mathbf{x}^T W^T W\mathbf{x} + 2\mathbf{a}^T W\mathbf{x} + \mathbf{a}^T\mathbf{a} \, d\mu(\mathbf{x})\right)^{1/2}. \end{aligned}$$

According to [30, p. 42], the first moment of Gaussian variable $t \sim N(\eta, \Sigma)$, and mean of square variable $\mathbf{t}$ can be obtained as follows:

$$E(\mathbf{t}) = \boldsymbol{\eta}$$
$$E(\mathbf{t}^T W^T W\mathbf{t}) = \text{trace}(W^T W \Sigma) + \boldsymbol{\eta}^T W^T W \boldsymbol{\eta}.$$

Therefore, the following equations can be obtained:

$$L_2(f_1, f_2) = \sum_{i=1}^{K} \alpha_i \left\{ \begin{array}{c} \text{trace}(W^T W \Sigma_i) + \boldsymbol{\eta}_i^T W^T W \boldsymbol{\eta}_i \\ +2\mathbf{a}^T W \boldsymbol{\eta}_i + \mathbf{a}^T \mathbf{a} \end{array} \right\}. \quad (4)$$

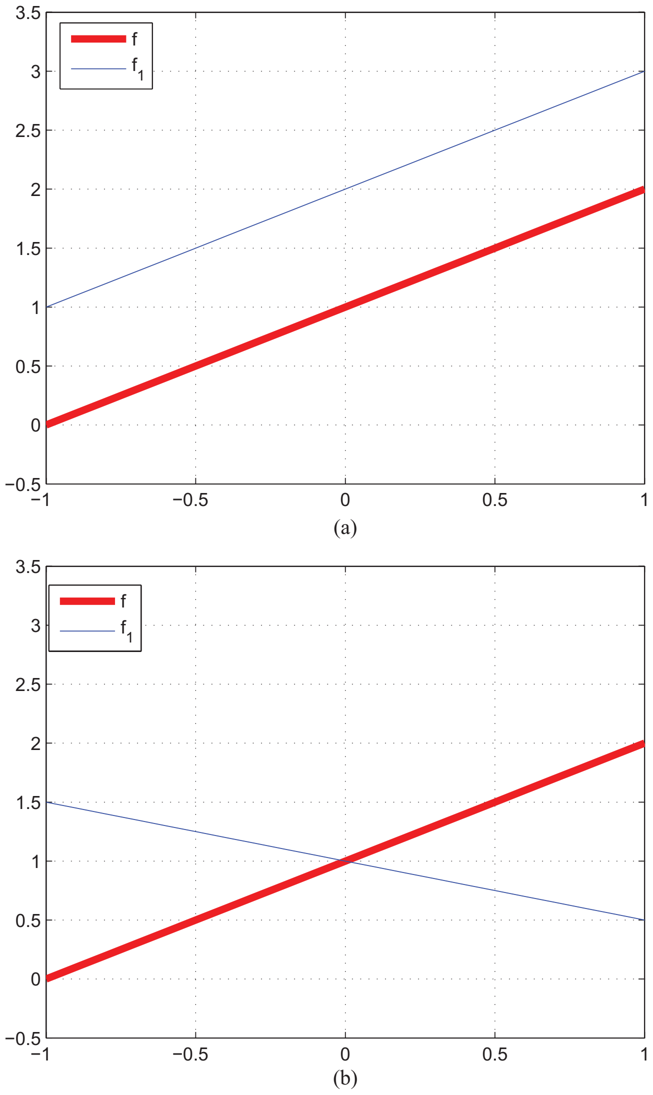


Fig. 2. (a) f and f_1 share the same orientation, but differ in the offset. (b) f and f_1 have the same offset but differ in the orientation.

a) Discussions on features characterizing the data chunks: In this paper, we have theoretically and empirically demonstrated how to use the functional distance in the model space as the representations of data chunks, instead of using any other types of features characterizing the shape, distribution, statistics of current data chunks, or directly using the parameters from the readout models as features.

If the model class that generated the data were known (e.g., AR of some fixed order), it would, of course, be advisable to use that model class to represent successive segments of the time series. But this is too strong an assumption in many practical situations. We cannot assume that we know the model class behind the data. Reservoir computing provides and studies "nonparametric" models of dynamic data. Reservoir models have been extensively shown to be able to successfully process and model time series of a surprisingly wide variety of types (from deeper memory deterministic chaotic systems to shorter memory stochastic sequences) [21]. The key idea is to use a fixed nonlinear high-dimensional nonautonomous dynamical system with fading memory as a general temporal filter, on top of which it is usually sufficient to train a linear readout mapping. The main point of our paper is that, given the capability of the reservoir architecture to function as a fixed general temporal filter, each time-series chunk is naturally represented by the linear readout from the dynamic filter that fits that data segment well. Besides the advantage of not having to specify the model class of the data in advance, we can calculate function distances between the linear readouts analytically. Thus "distances" between the data segments are translated into principled function distances between the corresponding readout mappings.

Of course, one can choose any set of features to represent the individual time series segments, if one finds them useful. The key idea in this paper is the representation of time-series chunks in a cognitive approach to fault detection through models that fit the data segments well. Learning is then transformed to learning in the model space (as opposed to learning in the original data/signal space). We argue that, even if we do not know the model class, we can still use learning in the model space framework by employing reservoir models as general temporal filters. Learning is then performed in the space of linear readouts with an appropriate function distance in that space. Distance between two functions should be parameter-free, in the sense that it should not depend on the particular parameterization used. If the models form a Riemannian manifold, parameterization is just a choice of local coordinate chart for the Riemannian model manifold in the function space. As an example, consider the class of univariate Gaussian distributions. One can parameterize 1-D Gaussians in many different ways, e.g., using the first two central moments (mean μ, variance σ^2). But one can use natural parameterization instead, or use (μ^{10}, σ^{-10}). Whatever parameterization one uses, the only relevant thing is how the two Gaussians differ as functions—e.g., in the L_2 sense—not what the Euclidean distance between their parameters. This is the principal reason why we avoided using *ad hoc* distance measures on parameters. Our features for data segments are not parameters of the corresponding linear readouts but the linear readouts themselves (as functions).

As for using other features characterizing the shape, distribution, and statistics of data chunks, of course, any appropriate features could be used. However, for calculating robust statistics-based features, one would need longer time segments, which is not optimal in the fault detection setting. In this paper, we use a model-based approach with a fixed reservoir—linear readouts are models characterizing the data chunks. Such models only require moderate amounts of data for learning.

III. Incremental One-Class Learning for Cognitive Fault Diagnosis

In fault diagnosis, it should be determined whether a running subsystem/component is in a normal operation condition or a faulty situation is occurring. It is relatively inexpensive and simple to obtain measurements from a normally working system (although sampling from all possible normal situations might still be expensive). In contrast, sampling from faulty situations requires the system to break down in various ways to obtain faulty measurement examples. The construction of a fault library will therefore be very expensive, or completely impractical. In this section, we focus on this challenge and aim to develop an algorithm that can identify unknown faults and construct a fault library dynamically, which will facilitate fault isolation based on this library.

Algorithm 1 Incremental One-Class Learning for Cognitive Fault Detection

```
1: Input: multiple input and multiple output data stream $\mathbf{s}_1, \ldots, \mathbf{s}_t, \mathbf{s}_{t+1}, \ldots$, where $\mathbf{s}_t = (u_1, \ldots, u_V, y_1, \ldots, y_O)^T$, $V$ is the number of signal inputs and $O$ is the number of outputs. The data segment $\mathbf{s}_1, \ldots, \mathbf{s}_t$ are healthy states of the system; parameters ($\sigma$ and $\nu$) of one-class SVMs; window size $m$.
2: Output: model library $lib$.
3: for each sliding window $(\mathbf{s}_i, \ldots, \mathbf{s}_{i+m-1})$, $1 \le i \le t+1-m$ do
4:   Fit the deterministic reservoir computing model:
5:   $drc(\mathbf{s}_i, \ldots, \mathbf{s}_{i+m-1}) \rightarrow f_i$
6: end for
7: Calculate the pairwise model distance matrix $\mathbf{L}_2(f_i, f_j)$, $1 \le i, j \le t+1-m$ according to (1). The kernel matrix for one-class SVMs can be calculated as $\mathbf{K} = exp\{-\sigma \cdot \mathbf{L}_2\}$.
8: Train a one-class SVM $\Theta_0$ using the existing kernel matrix: $OCS(\mathbf{K}, \nu) \rightarrow \Theta_0$ and add $\Theta_0$ in the model library $lib = \{\Theta_0\}$.
9: for sliding window $(\mathbf{s}_j, \ldots, \mathbf{s}_{j+m-1})$, $j > t$ do
10:   $drc(\mathbf{s}_j, \ldots, \mathbf{s}_{j+m-1}) \rightarrow f_j$;
11:   if $f_j$ belongs to a known fault $\Theta_k$ in the $lib$: i.e., evaluate the one class learner $\Theta_k$ with the data point $f_j$ to see whether $f_j$ is within the decision boundary of $\Theta_k$, i.e., $eval(\Theta_k, f_j) = 1$. then
12:     Incrementally train the one class learner $\Theta_k$ with the new "data point," i.e., model $f_j$, and empty the candidate pool;
13:   else
14:     put $f_j$ in the candidate pool;
15:   end if
16:   if size of candidate pool $> 0.5 * m$ then
17:     Train a new one-class learner $\Theta_{k+1}$ with the candidate pool
18:     Add $\Theta_{k+1}$ to $lib$ and empty the candidate pool
19:   end if
20: end for
```

Based on the "learning in the model space" framework (Fig. 1), one-class learning [7] will be employed in the model space for fault diagnosis. One-class classification is a special type of classification algorithm. One-class SVMs are used to discover a hyperplane that has maximal distance to the origin in the kernel feature space with the given training examples falling beyond the hyperplane [7].

Note that the signal characteristics can change at different positions of the sliding window. That means that the underlying measure μ over reservoir activations $\mathbf{x}$ can change.

Consider two readouts f_i and f_j obtained from two sliding window positions i and j. If reservoir activations in positions i and j are considered, we would obtain two distances $L_{\mu_i}(f_i, f_j)$ and $L_{\mu_j}(f_i, f_j)$, respectively.[2] The distance f_i, f_j based on the sampling approach is then

$$\tilde{L}_2(f_i, f_j) = L_{\mu_i}(f_i, f_j) + L_{\mu_j}(f_i, f_j). \tag{5}$$

In this paper, we propose an algorithm that can construct the fault library online. The idea is to use each one-class learner to represent each fault/subfault segment by using the "learning in the model space" approach. In the beginning, a normal one-class learner Θ_0 will be constructed based on the normal signal segments. With the sliding window moving forward, we continually apply Θ_0 to judge whether a fault occurs. If a fault is coming, we will train a new one-class learner Θ_i for fault i. Then, we keep monitoring the signal and determine whether the ongoing signal segment belongs to the normal state or a known fault. If not, we build a new one-class learner Θ_i and include it in the model library. This is illustrated in Algorithm 1, which includes the following major steps.

1) Prepare healthy data by applying deterministic reservoir model drc to the sliding windows (size m) in the first t steps, i.e., the healthy dataset $\{f_1, f_2, \ldots, f_{t+1-m}\}$ is sequentially induced (Lines 3–6).
2) Calculate the pairwise model distance matrix $\mathbf{L}_2(f_i, f_j)$ according to (1) and the kernel matrix $\mathbf{K} = \exp\{-\sigma \cdot \mathbf{L}_2\}$, and then train a one-class SVM (OCS) Θ_0 [7][3] using this kernel matrix $\mathbf{K}$ to act as the healthy model referee Θ_0 (Lines 7 and 8).
   In one-class SVMs, the Gaussian RBF kernel is employed with the data distance replaced by the model distance $L_2(f_i, f_j)$
   $$K_\sigma(f_i, f_j) = \exp\{-\sigma \cdot L_2(f_i, f_j)\}.$$
3) With the sliding window moving forward, if a new f_j belongs to an existing model Θ_k,[4] evaluate the one-class learner Θ_k (referee for fault k) at the "point" f_j, i.e., $eval(\Theta_k, f_j) = 1$, incrementally train the existing one class SVM Θ_k with this new data point f_j, and empty the candidate pool. Otherwise, put the "point" f_j in the candidate pool (Lines 9–15).
4) If the number of data points in the candidate pool exceeds half of the window size m, train a new one-class learner Θ_{k+1} (to act as the referee) for fault $k+1$ with the data points in the candidate pool and empty the candidate pool (Lines 16–18).

In the above algorithm, the assumption is that the system is running normally in the first t steps. Although the window size m should be relatively large (e.g., > 200 time steps) to accurately fit the dynamic models (e.g., deterministic reservoir computing in this paper), the sliding window is moved forward by one time step, which reduces fault detection delays.

[2]The measures μ_k will be represented by reservoir activation samples at window position k.

[3]Implemented by LibSVM: http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

[4]If the new point f_j is classified to more than one model by one-class SVMs, count the point in the last model because of sequential correlation.

TABLE I
ALGORITHMS AND PARAMETERS

Algorithm	Space	Parameters
T2	Signal	—
DBscan	Model	k Number of neighborhood ε Neighborhood radius
AP-model	Model	—
AP-signal	Signal	—
OCS-model	Model	σ Gaussian kernel parameter ν Upper bound of outliers
OCS-signal	Signal	σ Gaussian kernel parameter ν Upper bound of outliers
ARMAX-OCS	Model	σ Gaussian kernel parameter ν Upper bound of outliers m Number of nodes in reservoir (25) p p Autoregressive terms q Moving average terms b Exogenous inputs terms
RC-OCS	Model	σ Gaussian kernel parameter ν Upper bound of outliers m Number of nodes in reservoir (25)
DRC-OCS (sampling)	Model	σ Gaussian kernel parameter ν Upper bound of outliers m Number of nodes in reservoir (25)
DRC-OCS	Model	σ Gaussian kernel parameter ν Upper bound of outliers m Number of nodes in reservoir (25)

IV. EXPERIMENTAL STUDIES

This section presents experimental results in four fault-diagnosis scenarios, which include one synthetic nonlinear autoregressive moving average (NARMA) system with three different signals; one van der Pol oscillator with three faults imposed; one benchmark three-tank system with three faults; and the Barcelona water system with 31 faults. We investigate fault detectability and fault isolation capability using a number of approaches.

A. Experimental Settings

In our experiments, to evaluate the "learning in the model space" framework for fault diagnosis, a number of approaches have been adopted for comparison. The approaches include Hotelling's T-squared statistic test (T2) [31], a density-based algorithm for discovering clusters in large spatial databases with noise (DBscan) [32], affinity propagation [33] in the model space (AP-Model), affinity propagation in the signal space (AP-Signal), one-class SVMs [7] in the model space (OCS-Model), one class SVMs in the signal space (OCS-Signal), autoregressive moving-average model with exogenous inputs with incremental one-class leaner (ARMAX-OCS), reservoir computing with incremental one-class leaner (RC-OCS), deterministic reservoir computing with incremental one-class leaner (DRC-OCS), and DRC-OCS (sampling) where the model distance matrix is estimated by sampling method [3 and (5)]. Table I summarizes all the algorithms employed in this paper.

The signal space is generated by selecting p consecutive points, i.e., $\{\mathbf{s}_t, \ldots, \mathbf{s}_{t+p-1}\}$, where $\mathbf{s}_t = (u_1, \ldots, u_V, y_1, \ldots, y_O)^T$, as a training point by rearranging these p points to one vector. The order p is selected in the range $[1, 30]$.

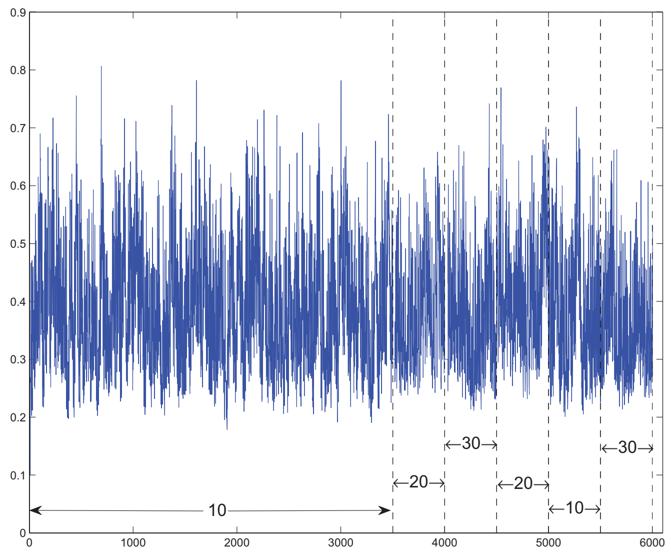


Fig. 3. Illustration of three NARMA sequences with different orders (10, 20, and 30).

In the following four datasets, we generate 3000 time steps for normal signal and each fault signal, respectively, and employ a sliding window (size 500) to generate a series of data segments, which are employed to train deterministic reservoir model. In each dataset, the first 1000 time steps of the signal are normal, i.e., the first 500 models are normal with window size 500.

The parameters of DBscan are optimized by minimizing the number of discovered classes and the false alarm rates using the first 500 normal points. The parameters of ARMAX are selected by minimizing the normalized mean squared error (NMSE) in the first 1000 time steps. The parameters of one-class SVMs in OCS-Model, OCS-Signal, ARMAX-OCS, RC-OCS, and DRC-OCS are optimized by fivefold cross validation using the first 500 data points.

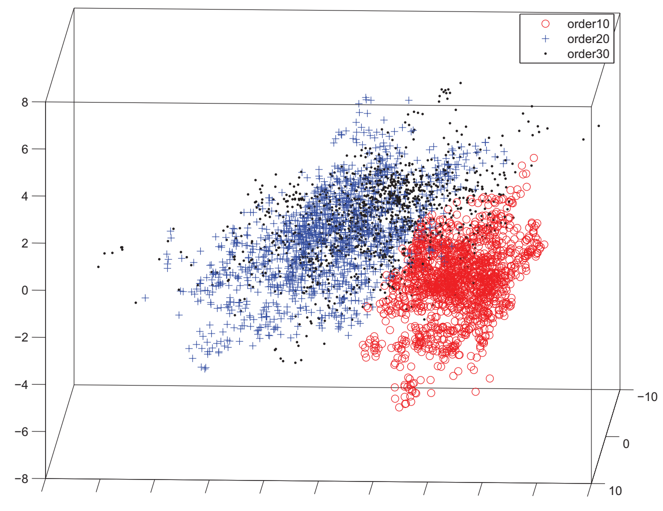
B. NARMA System

In NARMA, the current output depends on both the input and the previous output. Generally speaking, it is difficult to model this system because of high nonlinearity and possibly long memory. In this paper, we employ three NARMA time series with orders $O = 10, 20, 30$ which are given by (6)–(8), respectively

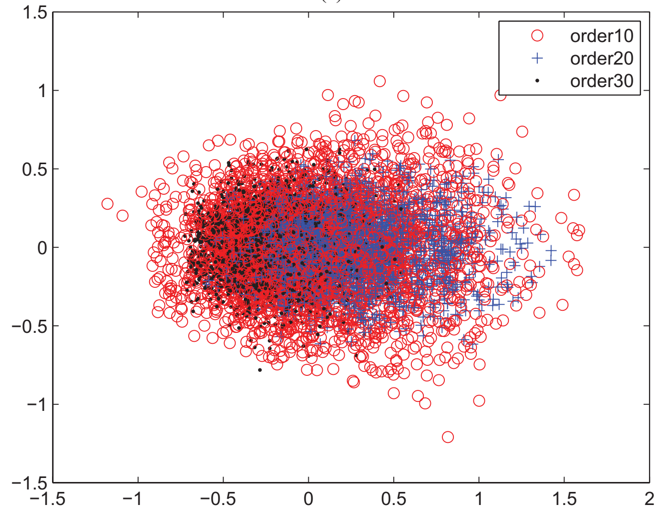
$$y(t+1) = 0.3y(t) + 0.05y(t)\sum_{i=0}^{9} y(t-i) + 1.5u(t-9)u(t) + 0.1 \tag{6}$$

$$y(t+1) = \tanh(0.3y(t) + 0.05y(t)\sum_{i=0}^{19} y(t-i) + 1.5u(t-19)u(t) + 0.01) + 0.2 \tag{7}$$

$$y(t+1) = 0.2y(t) + 0.004y(t)\sum_{i=0}^{29} y(t-i) + 1.5u(t-29)u(t) + 0.201 \tag{8}$$



(a)



(b)

Fig. 4. (a) Visualization of the NARMA dataset in the model space and (b) signal space ($p = 30$) by multidimensional scaling (MDS).

where $y(t)$ is the system output at time t, $u(t)$ is the system input at time t ($u(t)$ is an i.i.d stream generated uniformly in the interval $[0, 0.5)$.

The three sequences are illustrated in Fig. 3. The three NARMA sequences look quite similar, and it is very difficult to separate them on the basis of the signal only.

Fig. 4 shows MDS analysis[5] of the NARMA dataset in the model space (top) and in the signal space (bottom). Based on this figure, it is relatively easy to separate different classes in the model space, where most of the data points overlap in the signal space. The figure confirms that the model-based representation is able to effectively represent the signals. In Table III, several supervised classification techniques have been employed to confirm the benefits of using model-space-based approaches.

C. Van der Pol Oscillator

The Van der Pol oscillator [34] has been a subject of extensive research, and its discrete-time expressions play an

[5]MDS aims to preserve the pairwise distance between points, which is suitable for preserving the model distance for visualization.

TABLE II

PARAMETERIZATIONS OF FAULTS. MFD STANDS FOR MAXIMUM FLOW/DEMAND

ID	Faulty element	Type	Magnitude	ID	Faulty element	Type	Magnitude
1	iOrioles	1	−25%	17	iStaClmCervello	3	0.01%
2	iOrioles	2	−25%	18	iStaClmCervello	4	0.5%
3	iOrioles	2	−10%	19	iStaClmCervello	5	-
4	iOrioles	3	0.001%	20	iStaClmCervello	6	4
5	iOrioles	3	0.1%	21	iCesalpina1	1	10%
6	iOrioles	4	10%	22	iCesalpina1	2	-15%
7	iOrioles	4	1%	23	iCesalpina1	3	0.01%
8	iOrioles	5	-	24	iCesalpina1	4	0.75%
9	iOrioles	6	2	25	iCesalpina1	5	-
10	c175LOR	1	−20%	26	iCesalpina1	6	0.75
11	c175LOR	2	−15%	27	c263CES	1	30%
12	c175LOR	3	0.01%	28	c263CES	2	−15%
13	c175LOR	4	1%	29	c263CES	3	0.025%
14	c175LOR	5	-	30	c263CES	4	0.5%
15	iStaClmCervello	1	−15%	31	c263CES	5	-
16	iStaClmCervello	2	−7.5%				
	Details & Parameter			Type	Details & Parameter		
1	Additive offset (%MFD)			4	Additive drift (%MFD)		
2	Additive incipient offset (%MFD)			5	Abrupt freezing (-)		
3	Noise (variance %MFD)			6	Multiplicative offset (divided by)		

TABLE III

COMPARISONS OF MODEL-SPACE-BASED APPROACH AND SIGNAL-BASED APPROACH USING SUPERVISED LEARNING TECHNIQUES

Algorithm	NARMA		Van der Pol		Three-Tank		Water	
	Model	Signal	Model	Signal	Model	Signal	Model	Signal
CART	**0.00(0.00)**	0.33*(0.01)	**0.07(0.01)**	0.11*(0.01)	**0.01(0.00)**	0.02*(0.00)	**0.06(0.01)**	0.11*(0.00)
SVMs	**0.00(0.00)**	0.07*(0.01)	**0.05(0.01)**	0.07*(0.01)	**0.00(0.00)**	**0.00(0.00)**	**0.06(0.00)**	0.14*(0.00)
OCS	**0.04(0.01)**	0.32*(0.01)	**0.15(0.01)**	0.27 *(0.01)	**0.02(0.01)**	0.10*(0.01)	**0.09(0.01)**	0.23*(0.00)
Bagging	**0.00(0.00)**	0.24*(0.01)	**0.01(0.00)**	0.07*(0.00)	**0.00(0.00)**	0.01*(0.01)	**0.04(0.01)**	0.08*(0.01)
Boosting	**0.00(0.00)**	0.33*(0.01)	**0.15(0.01)**	0.22*(0.01)	**0.01(0.00)**	0.04*(0.00)	**0.07(0.01)**	0.16*(0.00)

The reported results are based on ten runs of fivefold cross validation. An $*$ means that the difference between model and signal representations are statistically significant. The boldface indicates better performance in either model or signal space for each algorithm on each data set.

important role in the numerical investigations. A discrete-time Van der Pol oscillator can be obtained as

$$\begin{aligned} y_1(k) &= y_2\Delta t + y_1(k-1) \\ y_2(k) &= y_2(k-1) + y_2(k-1)(1 - y_1(k-1)^2)\Delta t \\ &\quad - y_1(k-1)\Delta t + \epsilon \end{aligned}$$

where ϵ is Gaussian white noise with variance 0.01.

Three faults are imposed to the van der Pol oscillator by adding $0.75\sin(y_1(k-1))\Delta t$, $0.75\tanh(y_1(k-1))\Delta t$ and $0.75\cos\left(y_1(k-1)^2\right)$ to $y_2(k)$. The van der Pol oscillator and the three faults are illustrated in Fig. 5.

D. Three-Tank System

The well-known three-tank problem [3] in Fig. 6 is presented to illustrate the effectiveness of the proposed algorithm. The cross section of these tanks is $A_i = 1m^2$, and there is a cross section $A_p = 0.1m^2$ at the end of each tank. The outflow rate is c_j, $i, j = 1, \ldots, 3$. The level of each tank is denoted by x_i $(0 \le x_i \le 10,\ i = 1, \ldots, 3)$.

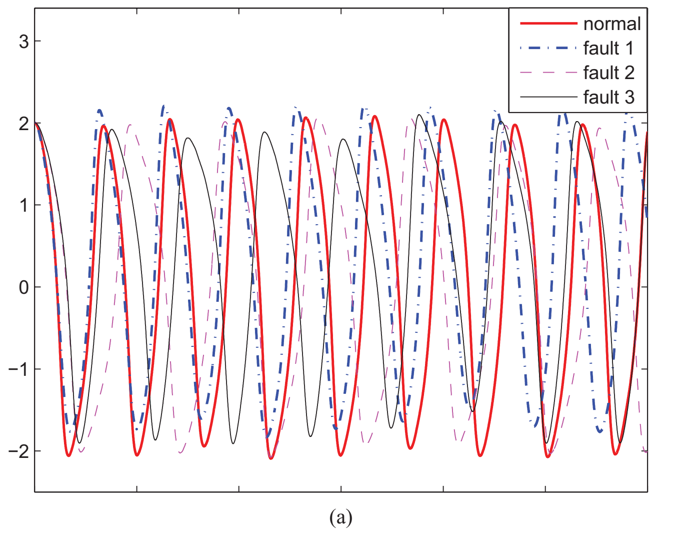
The input flows by two pumps are denoted by u_i with the restrictions $0 \le u_i \le 1m^3/s$, $i = 1, 2$. In this paper, the inflows are set as $u_1(k) = 0.2\cos(0.3kT_s) + 0.3$ and $u_2(k) = 0.25\cos(0.5kT_s) + 0.3$, respectively, and the initial levels in the tanks are 8, 6.5, and 5 m. In the model, three faults are introduced as follows.

1) Actuator fault in pump 1: the pump is partially or fully shut down.
2) Leakage in tank 3: there is a leak through a circular hole with unknown radius $0 < \rho_3 < 1$ in the tank bottom.
3) Actuator fault in pump 2: same as fault 1 but related to pump number 2.

Fig. 7 illustrates the water levels of three tanks in normal and three faulty situations.

E. Barcelona Water Distribution Network

The next application is the Barcelona Water Distribution Network (BWDN) [35]. BWDN supplies water to approximately 3 million consumers, distributed in 23 municipalities in a 424 km^2 area. Water can be drawn from both surface and underground sources. From these sources, water is supplied to 218 demand sectors through about 4645 km of pipelines. The complete transport network has been modeled using 63 storage tanks, 3 surface and 6 underground sources, 79 pumps, 50 valves, 18 nodes, and 88 demands.



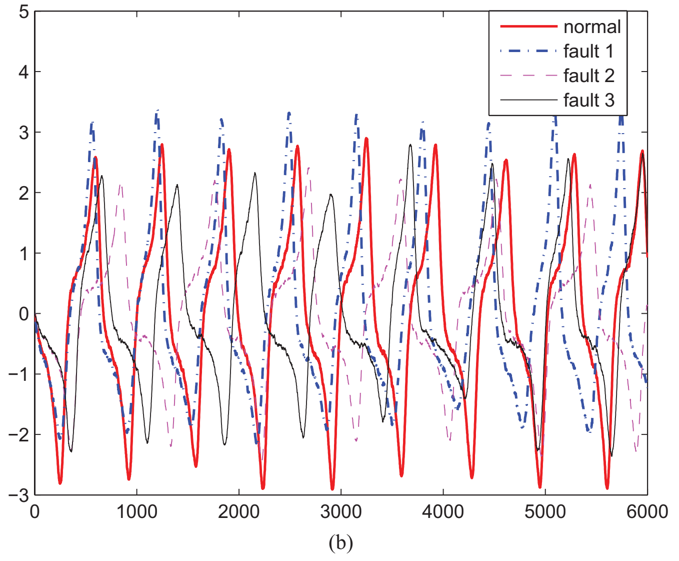


Fig. 5. Illustration of Van der Pol oscillator and three different faults. (a) $y_1(k)$ and (b) $y_2(k)$.

A detailed model of the BWDN has been developed using MATLAB/Simulink [35], which has been calibrated and validated using real data, which was contributed by our EU project partner, Prof. Joseba Quevedo, Universitat Politècnica de Catalunya, who has been working with Barcelona water supplying company Agbar (http://www.agbar.es/es/home.html) for more than ten years. Agbar has been using this software to compare the results with the actual reading from the sensor networks for fault diagnosis in their water system.

In this software, we can manipulate and inject different faults into the system. Studied faults are introduced in the two subsystems of the network shown in Fig. 8. In the two subsystems, we introduce 31 faults, which are detailed in Table II. These faults include actuator faults, actuator sensor faults, demand (input) sensor faults, and tanks (output) sensor faults. Four examples of faulty signals are illustrated in Fig. 9.

As there are two subsystems, two deterministic reservoir computing models, each with 25 nodes in the reservoir, have been employed in the proposed framework.

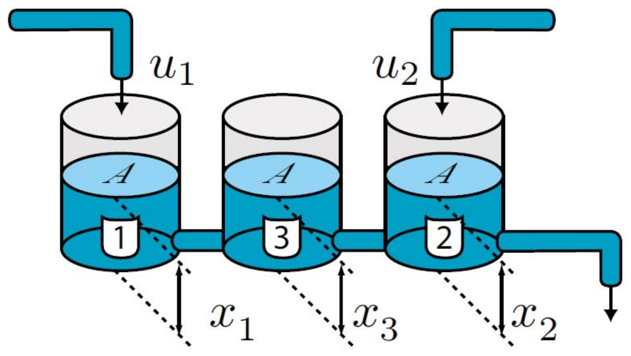


Fig. 6. Three-tank system [3].

F. Comparisons and Evaluations

This section will first report the comparisons of several supervised algorithms applied in the model space and signal space, respectively, and then evaluate those algorithms listed in Table I in terms of fault detectability and fault isolation ability.

In this section, we perform the statistical t-test for paired algorithms, e.g., classification and regression trees (CART) in the model space versus in the signal space in Table III, T2 versus DRC-OCS in Table IV, and DBscan versus DRC-OCS in Table V on each single dataset. We have carried out statistical tests on the metrics used in this paper including the error rate (Table III); fault detection rate (FDR) and false alarm rate (FAR) (Table IV); precision, recall, and specificity (Table V). The threshold of the statistical t-tests is set to 0.05.

In above section, the model space and signal space have been illustrated by the MDS algorithm. However, because of the high dimensionality, the visualizations might not reveal the real relationship of these data points in the high-dimensional space. In order to compare the model-space- and signal-space-based approaches, Table III reports the comparisons of the representations of model space and signal space using a number of supervised learning algorithms, including CART, SVMs, OCS, Bagging (100 trees), and Adaboosting (100 trees).

In the signal space approach, the order p will be selected in the range $[1, 30]$ by a fivefold cross validation approach. The parameters of SVMs and one-class SVMs are optimized by fivefold cross validation.

Since the default setting of MATLAB is to optimize the CART algorithm,[6] we follow the default setting in MATLAB for CART. Bagging and Adaboosting are ensemble algorithms with decision trees (CARTs) as based learners (CARTs have been optimized by MATLAB). They have only one parameter[7] to specify, i.e., the number of trees in the ensembles. We use a popular choice (100 decision trees) in this comparisons.

The reported results in Table III are based on ten runs of fivefold cross validation. In Table III, model space

[6]In MATLAB, the function "classregtree", the default is to compute the full tree and the optimal sequence of pruned subtrees.

[7]Different variants of Bagging and Adaboosting may require more parameters.

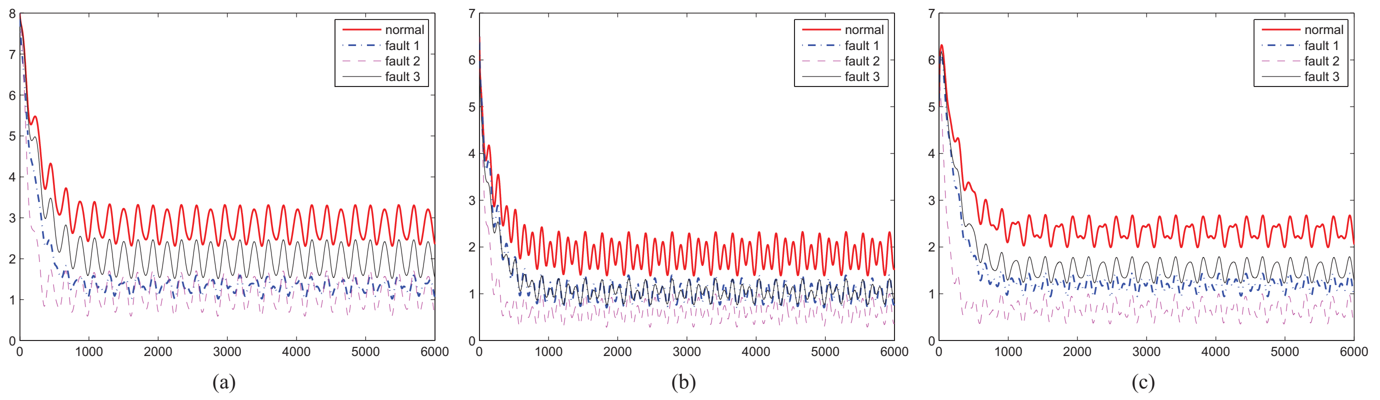


Fig. 7. Illustration of levels in the three tanks in the three-tank system and three different faults (a) tank 1, (b) tank 2, and (c) tank 3.

TABLE IV
COMPARISONS OF SEVERAL ALGORITHMS IN TERMS OF FAULT DETECTION ABILITY, I.E., FAULT DETECTION RATE (FDR) AND FALSE ALARM RATE (FAR)

	NARMA		Van der Pol		Three-Tank		Barcelona Water	
Algorithm	FDR	FAR	FDR	FAR	FDR	FAR	FDR	FAR
T2	0.9072*	0.1000*	0.3009*	0.0998*	0.2311*	0.0999*	0.2316*	0.1384*
DBscan	1*	0.0917*	0.9146*	0.2317*	0.8958*	0.0683*	0.7981*	0.1368*
OCS-model	1*	0.1102*	0.9310*	0.0509*	0.8521*	0.1082*	0.9313*	0.2683*
OCS-signal	0.7042*	0.2097*	0.7686*	0.2104*	0.7521*	0.2082*	0.4920*	0.3796*
AP-model	**1***	**0**	1.0000*	0.3405*	0.8407*	0.1128*	0.9014*	0.2678*
AP-signal	1*	0.5427*	1.0000*	0.7405*	0.7155*	0.2387*	0.8879*	0.2458*
ARMAX-OCS	0.9882*	0.0517*	0.8727*	0	0.9776*	0	0.7369*	0.1588*
RC-OCS	0.9747*	0.0558*	0.9762*	0.0158*	0.8387*	0	0.8271*	0.1079*
DRC-OCS(Sampling)	0.9789*	0	0.9804	0	**0.9926**	**0**	0.9327*	0.0817*
DRC-OCS	0.9921	0	**0.9818**	**0**	0.9919	0	**0.9762**	**0.0473**

An $*$ means that the difference between drc-ocs and other algorithms is statistically significant in terms of fdr and far. The boldface indicates the best performance among these algorithms for each dataset in terms of fdr and far.

TABLE V
COMPARISONS OF SEVERAL ALGORITHMS IN TERMS OF FAULT ISOLATION ABILITY

	NARMA (3 classes)				Van der Pol (4 classes)			
Algorithm	Classes	Precision	Recall	Specificity	Classes	Precision	Recall	Specificity
DBscan	4	0.6690*	0.7650*	0.8825*	10	0.7629*	0.6842*	0.8018*
AP-Model	271	0.9699*	0.9698*	0.9899*	367	0.8778*	0.8757*	0.9585*
ARMAX-OCS	5	0.9354*	0.9229*	0.9615*	2	0.4309*	0.4880*	0.7868*
RC-OCS	3	0.9637*	0.9615*	0.9808*	6	0.9606*	0.9583*	0.9861
DRC-OCS(Sampling)	3	0.9683*	0.9692*	0.9914	5	0.9617*	0.9726	0.9819*
DRC-OCS	3	**0.9861**	**0.9858**	**0.9929**	5	**0.9736**	**0.9731**	**0.9910**
	Three-tank (4 classes)				Barcelona Water (32 classes)			
Algorithm	Classes	Precision	Recall	Specificity	Classes	Precision	Recall	Specificity
DBscan	14	0.8742*	0.7561*	0.9253*	61	0.8019*	0.7326*	0.8654*
AP-Model	272	0.9713*	0.9704*	0.9901*	654	0.9366*	0.9428*	0.9751*
ARMAX-OCS	5	0.9914*	0.9923	0.9984	57	0.7826*	0.7419*	0.8237*
RC-OCS	9	0.9182*	0.8788*	0.9596*	44	0.8913*	0.8942*	0.9263*
DRC-OCS(Sampling)	7	**0.9940**	**0.9949***	**0.9988**	39	0.9219*	0.9310*	0.9513*
DRC-OCS	10	0.9931	0.9931	0.9977	48	**0.9538**	**0.9640**	**0.9871**

An $*$ means that the difference between drc-ocs and other algorithms is statistically significant in terms of precision, recall, and specificity for the four datasets. The boldface indicates the best performance among these comparable algorithms for each dataset in terms of precision, recall, and specificity, respectively.

representation usually achieves a lower error rate. In some cases, e.g., CART/SVMs in NARMA and SVM/Bagging in the three-tank system, model space representation can even achieve 100% accuracy. These results are consistent with those MDS visualizations, and confirm the benefits of using model space rather than signal space in fault diagnosis.

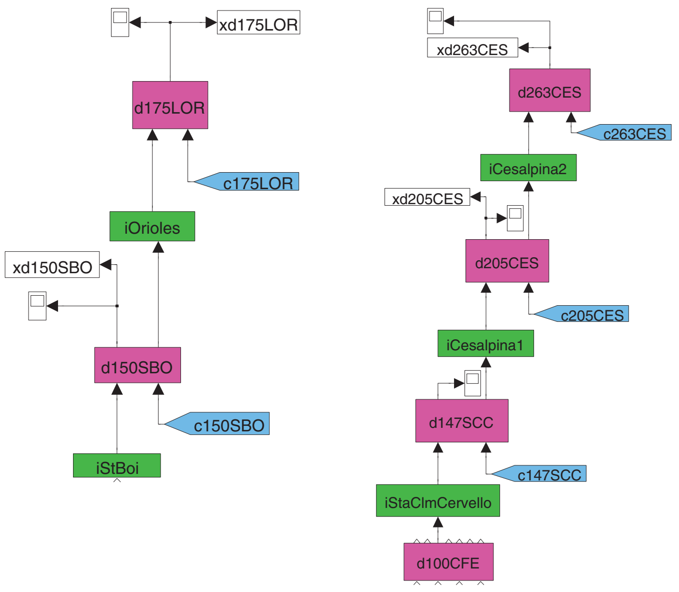


Fig. 8. Subsystems of the water network where faults are introduced. iOrioles, iStaClmCervello, iCesalpina1, and iCesalpina2 are actuators (controller). c175LOR, c147SCC, c205CES, and c263CES are demands (input). d175LOR, d147SCC, d205CES, and d263CES are tank level (output).

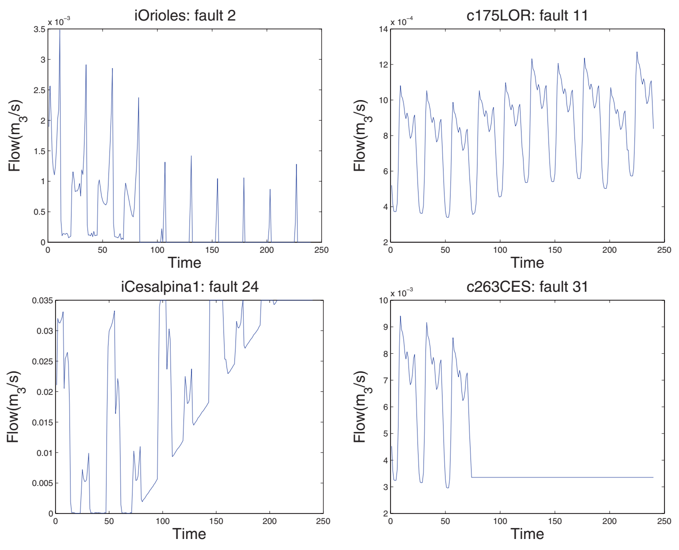


Fig. 9. Examples of faulty signals.

The reason for showing Table III in this paper is to confirm the benefits of the model space representation over the signal space representation in fault diagnosis. Therefore, we employ five supervised algorithms to demonstrate their separability in both model and signal spaces. In our algorithm, we did not use the supervised algorithms listed in Table III for fault diagnosis.[8] In other words, the objective here is not to compare the algorithms in terms of which is the best, but to show the differences between using the model space and signal space representations.

In fault diagnosis, the first step is to discriminate faults from normal situations. Table IV reports fault detection results using a number of algorithms listed in Table I. The parameters related to DBscan, one-class SVM, and ARMAX are optimized by fivefold cross validation in the normal period. In this table, the FDR and FAR are employed as two metrics.

According to Table IV, model space-based algorithms, such as DRC-OCS and RC-OCS, are superior to other algorithms. Since a deterministic reservoir is more stable than a random reservoir and there is no model assumption in DRC,[9] DRC-OCS is better than RC-OCS and ARMAX-OCS.

Although the sampling method of DRC-OCS could potentially obtain better estimates when the readout parameters are nonuniform, it would require dense sampling points, i.e., a large window size m in this case, with increased computational cost. However, due to real-time requirements and computational restrictions, the window size should be restricted for prompt response to faults. Hence, DRC-OCS (sampling) is often inferior to DRC-OCS.

The statistical-test-based algorithm T2 acts as a base line algorithm and usually has a lower FDR and a fair FAR. DBscan and affinity propagation (AP) are clustering-based algorithms. As these clustering algorithms do not make use of the information that the first t steps are normal, these algorithms did not perform well in the four applications.

In a time-varying environment, there may be unanticipated fault scenarios that would not have been encountered before. In this paper, we propose a dynamic fault library construction framework and its application on fault isolation. These results are reported in Table V.

In Table V, we first report the true number of classes and the discovered classes (i.e., number of faults plus normal class) using a number of algorithms for each dataset.[10] Then, we report the fault isolation performance of these algorithms in terms of precision, recall, and specificity.

Since the number of discovered faults does not equal to the true number of faults, we compare each true cluster Λ_i and these discovered clusters and merge those clusters by maximizing the overlap with Λ_i to a pseudo-cluster $\tilde{\Lambda}_i$. The performance metrics are obtained by comparing Λ and $\tilde{\Lambda}$.

Based on Table V, DRC-OCS usually outperforms other algorithms under these three metrics. AP-model performs well on the isolation stage, but often generates too many subfaults in the library, e.g., 270 subfaults versus two faults.

In the three "learning in the model space" approaches, i.e., DRC-OCS, RC-OCS, and ARMAX-OCS, DRC-OCS is the best and ARMAX-OCS is the most inferior one as it requires the model order selection for different applications. Without prior information for complex applications, it is usually difficult to select the model order. With limited sampling points due to real-time requirement, the sampling method of DRC-OCS is often inferior to DRC-OCS, though it often outperforms other approaches.

[8]One-class SVMs are not considered supervised learning algorithms as they are trained on examples from a single class only (no labels). They are used as "anomaly detectors" in our proposed cognitive fault diagnosis approach.

[9]ARMAX model assumes the model order and ARMAX-OCS might not perform well on signals with incorrect model assumption.

[10]Because of the assumption that the types of faults are unknown in advance, these compared algorithms always discover more faults than true number of faults by decomposing each true fault to a number of small fault segments.

Based on the results presented in Tables III, IV, and V, the proposed approach DRC-OCS achieves the best results and these results also confirmed that "learning in the model space" is an effective framework for fault diagnosis.

V. Conclusion

In this paper, an effective cognitive fault diagnosis framework was proposed to tackle the challenges in complex engineering systems in time-varying or unformulated environments. Instead of investigating the fault diagnosis in the signal space, this paper introduced "learning in the model space" framework which represents the MIMO data as a series of models fitted using a sliding window. By investigating the characteristic of these fitted models using a learning approach in the model space, one can identify and isolate faults effectively, and construct a fault library dynamically.

This paper applied the deterministic reservoir models to fit the MIMO data, since reservoir models are generic to fit a wide variety of dynamical features of the input-driven signals, and the deterministic reservoir models further simplify the model structure and thus improve the fitting performance.

To rigorously investigate these fitted models for fault diagnosis, this paper demonstrated the application of the distance definition in the model space for linear readout models. The model distance differs from the squared Euclidean distance of the readout parameters, indicating that more importance is given to the "offset" than "orientation" of the readout mapping. We also presented the estimated forms of model distance by using either sampling methods or a Gaussian mixture model when the domain of readout parameters is nonuniform.

By replacing the data distance matrix with the model distance matrix, one-class SVMs are able to "learn" in the model space to identify normal/abnormal models. To accommodate unknown faults, the algorithm "incremental one-class learning in the model space" was proposed to identify and isolate faults, and simultaneously construct the fault library.

To evaluate this proposed framework with other related fault diagnosis approaches, three benchmark systems and one simulated software for Barcelona water system have been employed. The results confirmed both the benefits of representing MIMO data in the model space and the effectiveness of "learning in model space" framework.

"Learning in the model space" is an effective framework for complex data representation and fault diagnosis.[11] Instead of using reservoir models and one-class SVMs as fitting and discriminating models, respectively, there should be other effective opinions or combinations for various application systems, which comprise our future work.

Acknowledgment

The authors would like to thank the anonymous reviewers for their constructive comments which helped improve this paper significantly.

[11]Although model-space-based approaches can be robust and effective for fault types occurring over a period of time, it might be ineffective for very short term and minor faults, where signal-based approaches can be more effective.

References

[1] J. Chen and R. J. Patton, *Robust Model-Based Fault Diagnosis for Dynamic Systems*. Norwell, MA, USA: Kluwer, 1999.

[2] J. J. Gertler, *Fault Detection and Diagnosis in Engineering Systems*. New York, NY, USA: Marcel Dekker, 1998.

[3] X. Zhang, M. Polycarpou, and T. Parisini, "A robust detection and isolation scheme for abrupt and incipient faults in nonlinear systems," *IEEE Trans. Autom. Control*, vol. 47, no. 4, pp. 576–593, Apr. 2002.

[4] X. Zhang, T. Parisini, and M. Polycarpou, "Sensor bias fault isolation in a class of nonlinear systems," *IEEE Trans. Autom. Control*, vol. 50, no. 3, pp. 370–376, Mar. 2005.

[5] X. Yan and C. Edwards, "Nonlinear robust fault reconstruction and estimation using a sliding mode observer," *Automatica*, vol. 43, no. 9, pp. 1605–1614, Sep. 2007.

[6] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, Aug. 2009.

[7] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.

[8] A. Vemuri and M. Polycarpou, "Neural-network-based robust fault diagnosis in robotic systems," *IEEE Trans. Neural Netw.*, vol. 8, no. 6, pp. 1410–1420, Nov. 1997.

[9] V. Palade and C. Bocaniala, *Computational Intelligence in Fault Diagnosis*. New York, NY, USA: Springer Publishing Company, Inc., 2010.

[10] V. Venkatasubramanian, R. Rengaswamy, S. Kavuri, and K. Yin, "A review of process fault detection and diagnosis: Part III: Process history based methods," *Comput. Chem. Eng.*, vol. 27, no. 3, pp. 327–346, Mar. 2003.

[11] P. Kankar, S. Sharma, and S. Harsha, "Fault diagnosis of ball bearings using machine learning methods," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 1876–1886, Mar. 2011.

[12] M. Seera, C. P. Lim, D. Ishak, and H. Singh, "Fault detection and diagnosis of induction motors using motor current signature analysis and a hybrid FMM-CART model," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 1, pp. 97–108, Jan. 2012.

[13] M. Barakat, F. Druaux, D. Lefebvre, M. Khalil, and O. Mustapha, "Self adaptive growing neural network classifier for faults detection and diagnosis," *Neurocomputing*, vol. 74, no. 18, pp. 3865–3876, Nov. 2011.

[14] I. Yélamos, G. Escudero, M. Graells, and L. Puigjaner, "Performance assessment of a novel fault diagnosis system based on support vector machines," *Comput. Chem. Eng.*, vol. 33, no. 1, pp. 244–255, Jan. 2009.

[15] H. Zhang, J. Liu, D. Ma, and Z. Wang, "Data-core-based fuzzy min–max neural network for pattern classification," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 2339–2352, Dec. 2011.

[16] A. Barua and K. Khorasani, "Hierarchical fault diagnosis and health monitoring in satellites formation flight," *IEEE Trans. Syst., Man, Cybern., Part C, Appl. Rev.*, vol. 41, no. 2, pp. 223–239, Mar. 2011.

[17] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," German Nat. Res. Center Inf. Technol., St. Augustin, Canada, Tech. Rep. GMD-148, 2001.

[18] D. Li, M. Han, and J. Wang, "Chaotic time series prediction based on a novel robust echo state network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 787–799, May 2012.

[19] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural Comput.*, vol. 14, no. 11, pp. 2531–2560, Nov. 2002.

[20] J. J. Steil, "Backpropagation-decorrelation: Online recurrent learning with O(N) complexity," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Jul. 2004, pp. 843–848.

[21] A. Rodan and P. Tiňo, "Simple deterministically constructed cycle reservoirs with regular jumps," *Neural Comput.*, vol. 24, no. 7, pp. 1822–1852, Jul. 2012.

[22] K. Doya, "Recurrent networks: Supervised learning," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1995, pp. 796–800.

[23] H. Jaeger, "Tutorial on training recurrent neural networks," German Nat. Res. Inst. Comput. Sci., Berlin, Germany, Tech. Rep. GMD-159, 2002.

[24] A. Rodan and P. Tiňo, "Minimum complexity echo state network," *IEEE Trans. Neural Netw.*, vol. 22, no. 1, pp. 131–144, Jan. 2011.

[25] K. Brodersen, T. Schofield, A. Leff, C. Ong, E. Lomakina, J. Buhmann, and K. Stephan, "Generative embedding for model-based classification of fMRI data," *PLoS Comput. Biol.*, vol. 7, no. 6, p. e1002079, Jun. 2011.

[26] D. Haussler, "Convolution kernels on discrete structures," Ph.D. dissertation, Dept. Comput. Sci., Univ. California at Santa Cruz, Santa Cruz, CA, USA, Tech. Rep. UCSC-CRL-99-10, Jul. 1999.

[27] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *Neural Information Processing Systems*. Cambridge, U.K.: Cambridge Univ. Press, 1999, pp. 487–493.

[28] T. Jebara, R. Kondor, and A. Howard, "Probability product kernels," *J. Mach. Learn. Res.*, vol. 5, pp. 819–844, Jan. 2004.

[29] A. Bosch, A. Zisserman, and X. Muoz, "Scene classification using a hybrid generative/discriminative approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 4, pp. 712–727, Apr. 2008.

[30] K. Petersen and M. Pedersen, "The matrix cookbook," Dept. Inform. Math. Modell., Tech. Univ. Denmark, Kongens Lyngby, Denmark, Tech. Rep. 20121115, 2008.

[31] J. Cho, J. Lee, S. Wook Choi, D. Lee, and I. Lee, "Fault identification for process monitoring using kernel principal component analysis," *Chem. Eng. Sci.*, vol. 60, no. 1, pp. 279–288, Jan. 2005.

[32] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.

[33] B. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Jan. 2007.

[34] D. Kaplan and L. Glass, *Understanding Nonlinear Dynamics*. New York, NY, USA: Springer-Verlag, 1995.

[35] J. Quevedo, V. Puig, G. Cembrano, J. Blanch, J. Aguilar, D. Saporta, G. Benito, M. Hedo, and A. Molina, "Validation and reconstruction of flow meter data in the Barcelona water distribution network," *Control Eng. Pract.*, vol. 18, no. 6, pp. 640–651, Jun. 2010.

Huanhuan Chen (M'09) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 2004, and the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2008.

He is currently a Full Professor with USTC-Birmingham Joint Research Institute of Intelligent Computation and Its Applications, School of Computer Science and Technology, University of Science and Technology of China, Hefei. His current research interests include statistical machine learning, data mining, fault diagnosis, and evolutionary computation.

Dr. Chen received the 2011 IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award (the only winner) and the 2009 CPHC/British Computer Society Distinguished Dissertations Award (the runner up). His work "Probabilistic Classification Vector Machines" on Bayesian machine learning has been awarded the IEEE Transactions on Neural Networks Outstanding Paper Award (bestowed in 2011 and only one paper in 2009).

Peter Tiňo received the M.Sc. degree from the Slovak University of Technology, Bratislava, Slovakia, and the Ph.D. degree from the Slovak Academy of Sciences, Bratislava.

He was a Fulbright Fellow with the NEC Research Institute, Princeton, NJ, USA, and a Post-Doctoral Fellow with the Austrian Research Institute for AI, Vienna, Austria, and with Aston University, Birmingham, U.K. Since 2003, he has been with the School of Computer Science, University of Birmingham, Birmingham, where he is currently a Reader in complex and adaptive systems. His current research interests include dynamical systems, machine learning, probabilistic modeling of structured data, evolutionary computation, and fractal analysis.

Dr. Tiňo was a recipient of the Fulbright Fellowship in 1994, the U.K. Chong-Kong Fellowship for Excellence in 2008, three Outstanding Paper of the Year Awards from the IEEE TRANSACTIONS ON NEURAL NETWORKS in 1998 and 2011, the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION in 2010, and the Best Paper Award at ICANN 2002. He serves on the editorial boards of several journals.

Ali Rodan received the M.Sc. degree from Oxford Brookes University, Oxford, U.K., and the Ph.D. degree from the University of Birmingham, Birmingham, U.K., in 2012.

He is an Assistant Professor with the University of Jordan, Amman, Jordan. His current research interests include recurrent neural networks, dynamical systems, and machine learning.

Dr. Rodan has co-organized two special sessions devoted to learning on temporal data at international conferences. Currently, he specializes in reservoir computation models and their design and theoretical properties.

Xin Yao (F'03) is a Chair (Professor) of computer science and the Director of the Centre of Excellence for Research in Computational Intelligence and Applications, University of Birmingham, Birmingham, U.K. He has been invited to give more than 70 keynote and plenary speeches at international conferences. He has published more than 400 refereed publications in international journals and conferences. His current research interests include evolutionary computation and ensemble learning.

He is a Distinguished Lecturer of the IEEE Computational Intelligence Society. He received the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, the 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), the 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and many other best paper awards at conferences. He received the Prestigious Royal Society Wolfson Research Merit Award in 2012 and was selected to receive the 2013 IEEE CIS Evolutionary Computation Pioneer Award. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008.