IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 29, 2018, accepted September 26, 2018, date of publication October 4, 2018, date of current version October 29, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2873770

Applications of Metaheuristics in Reservoir Computing Techniques: A Review

ABUBAKAR BALA[1,2], (Member, IEEE), IDRIS ISMAIL[1], ROSDIAZLI IBRAHIM[1], (Member, IEEE), AND SADIQ M. SAIT[3], (Senior Member, IEEE)

[1]Electrical and Electronics Engineering Department, Universiti Teknologi PETRONAS, Malaysia, Seri Iskandar 32610, Malaysia
[2]Electrical Engineering Department, Bayero University Kano, PMB 3011, Kano, Nigeria
[3]Computer Engineering Department and Center for Communications and IT Research, Research Institute, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

Corresponding author: Abubakar Bala (abala.ele@buk.edu.ng)

This work was supported in part by Universiti Teknologi PETRONAS (UTP), Malaysia, and in part by the King Fahd University of Petroleum and Minerals, Dhahran.

ABSTRACT Reservoir computing approaches have been around for almost two decades. They were developed to solve the difficult gradient-descent training of recurrent neural networks. However, in reservoir computing, the choice of parameters and architecture often leads to an optimization challenge. Most early applications have used trial and error with expert knowledge to select the right value for parameter(s)/architecture. This approach is usually cumbersome and difficult considering the large search space. Metaheuristics have been known to perform well in solving such kinds of problems. This review discusses areas where metaheuristics are used in the echo state network—a pioneer in the reservoir computing field. In addition, trends and research gaps are also discussed.

INDEX TERMS Artificial intelligence, artificial neural networks, echo state network, machine learning, metaheuristics, optimization algorithms, reservoir computing, review, survey.

I. INTRODUCTION

Artificial neutral networks (ANNs) or just Neural networks (NNs) are information processing systems that are built on a mathematical model. They attempt to mimic the parallel and non-linear information structures of the human brain by obtaining knowledge via experience. They copy the human brain in a minimum of two aspects. (i) The acquiring of knowledge from its environment via a training scheme. (ii) The storing of the obtained knowledge in connections between the neurons called synaptic weights [1]. McCulloch and Pitts were one the firsts to build a simple artificial neuron called the perceptron that mimicked the human brain in 1943 [2]. Shown in Fig. 1, it consists of many inputs (dendrites) and a single output (axon). From the figure, if we define $z(i) = \sum w_i x_i$ then the output is: $s(i) = f(z(i))$. Where $w_1, \ldots, w_n$ are the synaptic weights used to store learned knowledge and $f(.)$ is an activation function that is usually a sigmoid (often, hyperbolic tangent or logistic function) that introduces non-linearity to the network, aiding it in performing the required task. The perceptron marked the inception of neural network and artificial intelligence.

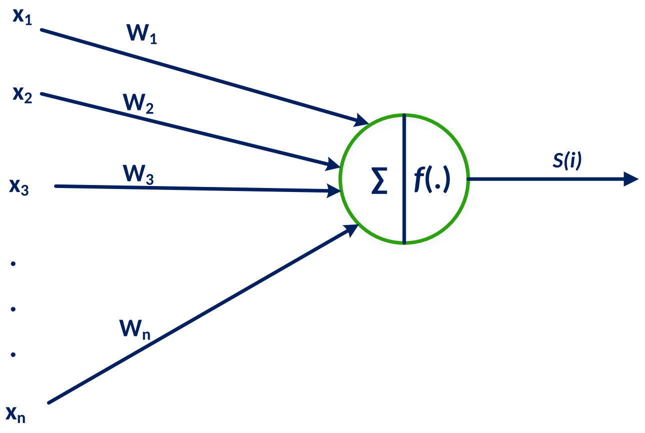


FIGURE 1. The basic perceptron [3].

The neural network is often applied by first dividing the obtained data into three sets: the training, the validation, and, the test sets. Firstly, random values are given to the synaptic weights. Subsequently, the training set which is usually larger than the other two sets is used to train/adapt the weights based on the known outputs within the training set. This is typically done by employing the use of an appropriate learning technique, such as linear regression or ridge regression [3]. The optional validation set is then used to fine tune the selection of network parameters such as the network layer size and number of hidden layers. The test set is then finally used to test the performance of the final model on fresh/unseen data.

ANNs have been categorized in various ways depending on one or more of its related characteristics [4]. In general, the ANN categorization may be based on: (i) The task it performs e.g., clustering, classification etc; (ii) The degree of connectivity of the network neurons (partial/full); (iii) The type of learning algorithm adopted; (iv) The type of learning rule, which is the engine that drives the learning algorithm; (v) The level of learning supervision needed (e.g., supervised, reinforced, unsupervised learning and their hybrids); and (vi) The direction of the flow of information within the network (recurrent versus feedforward).

The last category is one of the major classification of ANNs. The feedforward networks are the simplest form of neural networks (NNs) in which connections between network neurons are not allowed to form cycles. Thus, all information moves from the input(s) to the output(s) in one direction without feedbacks. Famous in this category is the multilayer perceptron (MLP). This lack of feedback connection limit their applications to mostly functional mapping problems [3]. In contrast, the recurrent neural networks (RNN) allow connections between neurons to form directed cycles. Thus, enabling temporal characteristics that make them suitable for solving sequential tasks.

This paper focuses on the RNN type of networks due to their versatility and numerous applications. In the next section we discuss the RNN in a bit more detail.

A. RECURRENT NEURAL NETWORKS (RNNS)

As stated earlier, RNNs possess cycles within their connections. The presence of these cycles gives the RNN two powers over the feedforward networks:

- When driven by input signals, its inner states can store nonlinear transformation of the inputs. Thus, it possesses a dynamic memory that enables it to perform time related tasks.
- In the absence of inputs, it can maintain an autonomous time related activation dynamics in its recurrent connections. Hence, while feedforward networks are regarded as functions, RNNs are dynamical systems.

From the view point of dynamical systems, two classes of RNN exists. In the first category, the models are described by symmetric interconnections and energy-reducing random dynamics. Their mathematical background is ingrained in statistical physics. Examples in this group include: Hopfield networks [5], Boltzmann machines [6], and deep belief networks [7]. These networks often have unsupervised learning and find applications in associative memories, unsupervised modeling of distributed data, data compression and static classification of patterns. On the other hand, the second category comprises of models that are characterized by connections that are directed with deterministic dynamics. They often perform nonlinear filters that convert inputted time series into a transformed time series at its output. Moreover, they have mathematical roots in nonlinear dynamic systems and are often trained in a supervised manner. RNNs in this category have been shown to be powerful tools for time series processing [8]. This review is concerned with RNNs in this category. Inspite of its huge potentials and excellent performance in academia and industry, the RNN has had limited impact in nonlinear modelling for a long time. This is mainly because it is tasking to train the RNN by gradient-descent-based techniques that aim to iteratively minimize training error. Even though many training algorithms had been developed, they have the following shortcomings:

- During learning, a progressive change in network parameter drives the network dynamics into bifurcations [9]. When this happens, the gradient data deteriorates and may not be properly defined. Thus, an assured convergence can be rarely achieved.
- An update of one parameter can have high computational price, and several cycles of update may be needed. This causes a lengthy training, making the training possible for mostly small networks having a little number of neurons.
- There is an inherently difficultly in learning dependencies needing long-range memory. This is because the required gradient data exponentially withers as time passes. However, this can now be overcome using long short-term memory networks [10].
- Moreover, advance training algorithms are computationally expensive. Additionally, they are required to be parameterized by several control parameters that are difficult to optimize. Thus, application of such algorithms need special skill and expertise.

Due to these challenges, in 2001, researchers developed a new paradigm for RNN design and training. The echo state network (ESN) [11] by Jaeger and the liquid state machine (LSM) [12] by Maass *et al*. These approaches have antecedents in computational neuroscience [13] and sequels in machine learning as backpropagation-decorrelation (BPDC) [14] rule of learning. They are now generally called reservoir computing (RC).

The RC approach to the RNN escapes from the drawbacks of the gradient-descent training of RNN with the following:

- The RNN is created randomly and remains fixed throughout the training cycle. The RNN in this case is referred to as a ''reservoir'' and is passively excited by the input signals. Moreover, it retains within its state, a nonlinear transformation of the inputs.
- The desired output is represented as a linear combination of signals from the reservoir units. It is often obtained via linear regression by using the teacher's output as a target.

Thus, as opposed to the gradient-descent based training, where all weights are trained, in the RC, the weights that are adapted are only those between the reservoir and the readouts. The RC methods have seen wide application because of its excellent model accuracy, good model capacity, ease of extension, and its linear connection to the architecture and dynamics of the mammalian brain. However, despite its excellent performance and wide applications, it is still an evolving field and has its own challenges. Precisely, simply generating a fixed random reservoir is not good enough. Thus, researchers have suggested that reservoir design should be tailored to the specific modelling task at hand.

Hence, main research findings in the RC field is aimed at comprehending how task performance is affected by reservoir properties. Moreover, some works are done in identifying appropriate reservoir construction and its adjustment techniques [8]. Additionally, novel ways of reading out from the reservoir, in addition to combining them into larger structures is being explored [15]. Although these new concepts deviate from the original RC idea of a fixed reservoir, the RC paradigm still stands and distinguishes itself from other RNN training techniques since the creation/training of the reservoir is independent and differs from that of the readouts.

B. OTHER REVIEWS

Some of the most prominent appraisals of the RC concepts include the work of Schrauwen *et al.* [16], where an overview of the RC methods are presented and research directions outlined. In a related review, Lukoševicius and Jaeger [17] discussed the RC techniques, particularity highlighting the paradigm shift in the RC concepts from the original proposals. Additionally, Lukoševicius and Jaeger [8] which is one of the most cited review of the RC presented a comprehensive appraisal of the RC, focusing more on the ESN. The survey discussed reservoir adaptation techniques and training methods. In a more recent work, Lukoševicius *et al.* [15] presented trends of the reservoir concepts after 10 years of its development.

In this review, we consider the application of metaheuristics in the RC concepts, particularly targeting the ESN which is a pioneering RC method.

Organization: The outstanding part of this review are arranged thus: Section II presents a summary of existing RC concepts. In Section III, the echo state network (ESN) is discussed. Section IV introduces metaheuristics with few examples of them. Section V presents works that employ metaheuristics to optimize the ESN. Furthermore, Section VI discusses the trends and current research gaps we deducted from the reviewed papers. Finally, Section VII concludes the paper.

II. TYPES OF RESERVOIR COMPUTING TECHNIQUES

This section presents a brief summary of the existing reservoir computing techniques. The subsequent sections then concentrates on the echo state network (ESN) which is one of the pioneering works in reservoir computing, and the target of this review.

A. THE LIQUID STATE MACHINE (LSM)

LSMs [18] are one of the pioneering techniques of reservoir computing, developed separately and concurrently with the echo state networks (ESN). LSMs were created from a framework of computational neuroscience. They aim to interpret the primary computational characteristics of neural microcircuits [12]. Moreover, they employ a more complex and practical biological model of spiking integrate-and-fire neurons together with a dynamic synaptic interconnection within its reservoir. The reservoir of the LSM is called a ''liquid'' which likens the excited states to ripples that form when an object is dropped in a pool of water. Compared with other models, the LSM is designed to handle real-time computations on a continuous data stream like spike trains. Thus, both the input and output of the LSM are data streams of continuous time. The process works by inserting the input data stream into a complex recurrent neural network that is big enough. Subsequently, owning to its dynamic characteristics, the network then transforms the lower resolution input stream into ''liquids'' with higher resolution [19]. These liquids are then mapped to produce the target output through a memory-less readout function. With these features, the LSM is claimed to perform well on non-linear system tasks. However, LSMs with complex synaptic models or spiking neurons are difficult to implement, initialize or fine tune. Additionally, they are more computationally expensive than the ESN type RNNs.

B. BACKPROPAGATION-DECORRELATION (BPDC)

The BPDC is not a network architecture of reservoir computing, but rather a learning rule. Developed in 2004 by Steil [14], it comprises the following: (i) A one-step backpropagation of errors by virtual teacher forcing. (ii) The utilization of the time related memory within the network that is modified on the basis of decorrelation of the activations within the reservoir. (iii) The utilization of a fixed reservoir neurons to minimize complexity. The learning complexity is reduced to $\mathcal{O}(N)$, since the learning rule is employed only for the readout/output weights. With a similar architecture to the ESN, the BPDC claims to be insensitive to reservoir parameter(s). Moreover, the BDPC can achieve a fast learning rate and hence can track highly dynamic signals. However, the disadvantage of this characteristic is that the trained network may easily forget the previously seen signals and become heavily biased to current data.

C. ECHO STATE NETWORK (ESN)

Developed in 2001 by Jaeger [11], the ESN is one of the leading techniques of RC. It is built on the idea that if a random RNN consists of certain arithmetic characteristics, training only the linear readout from the RNN is enough to produce good results in practical implementations. The untrained portion of the ESN is referred to as the ''dynamic reservoir'' and the states within the reservoir are said to be ''echoes'' of the input history.

This work focuses on this type of reservoir computing concept. This is because, not only is it one of the pioneering RC concepts, from existing literature point of view it is one of the most successful RC technique that exists thus far. The next section gives detail of the architecture and properties of the ESN network.

III. THE ESN ARCHITECTURE & OPTIMIZATIONS

The architecture of the basic ESN is shown in Fig. 2. It comprises three layers: an input layer with K units, a reservoir layer with N internal states and a readout/output layer with M units. The directed arrows in the figure denote synaptic weight connections. With the thick arrow lines representing required connections while the dashed arrow lines denote optional weight connections. Moreover, although omitted from the figure for brevity, the network often has an input bias unit, usually set to a value of one. This work considers a discrete ESN for temporal tasks. It has the following: $u(t) = [u_1(t), u_2(t), \ldots, u_K(t)]^{\mathsf{T}}$ inputs at time (t); N internal network units with internal states depicted as: $x(t) = [x_1(t), x_2(t), \ldots, x_N(t)]^{\mathsf{T}}$; and M output units as: $y(t) = [y_1(t), y_2(t), \ldots, y_M(t)]^{\mathsf{T}}$. Additionally, from the figure, the reservoir state $x(t)$ is given as:

$$x(t) = f(W^{in}u(t) + W^{r}x(t-1) + W^{back}y(t-1)) \qquad (1)$$

Where $f = \{f_1, f_2, \ldots, f_N\}$ are non-linear activation functions of the internal units, which is usually chosen as sigmoidal. While, $W^{in} = w^{in}_{i,j}$ is the $K \times N$ weight connections between the inputs and reservoir neurons. Moreover, $W^r = w^r_{i,j}$ is the $N \times N$ internal weight matrix of connections within the reservoir. In contrast, $W^{back} = w^{back}_{i,j}$ is the $M \times N$ output feedback weight matrix that connects the output back to the reservoir neurons. The ESN's output equation is given as:

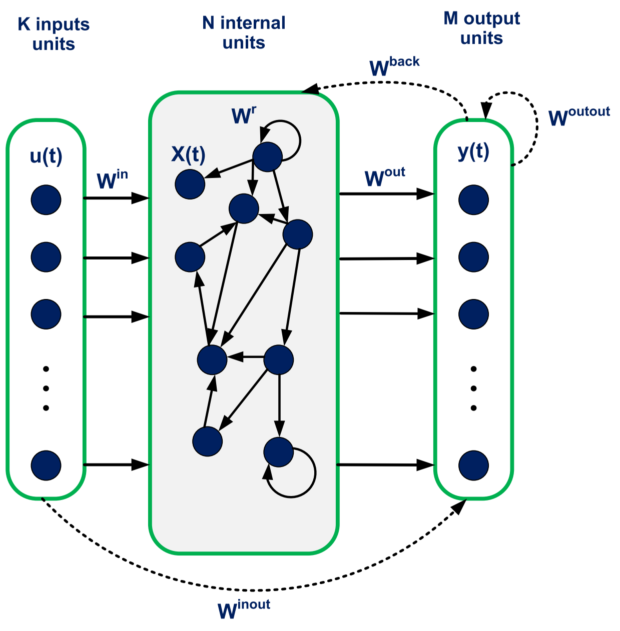


FIGURE 2. The basic ESN Architecture [20].

$$y(t) = f^{out}(W^{inout}u(t) + W^{out}x(t) + W^{outout}y(t-1)) \qquad (2)$$

However, if all links to the output neurons are depicted by W^{out_gen} i.e.,
$= W^{out\_gen} = \{W^{inout}, W^{out}, W^{outout}\}$ then the output may be given as:

$$y(t) = f^{out}\left(W^{out\_gen}[u(t); x(t); y(t-1)]\right) \qquad (3)$$

Where $f^{out} = \{f^{out}_1, f^{out}_2, \ldots, f^{out}_M\}$, are the output activation functions that are often chosen to be linear. Whereas, $W^{out\_gen} = w^{out\_gen}_{i,j}$ is the output weights with dimension $M \times (K + N + M)$. Furthermore, $[u(t); x(t); y(t-1)]$ is a concatenation of the input, internal and the past output.

Thus, the ESN may be depicted as a set of matrices: ESN $= \{W^{in}, W^{inout}, W^{r}, W^{out}, W^{outout}, W^{back}\}$. However, as shown in Fig. 2, the weights W^{inout}, W^{outout}, and W^{back} are optional. Moreover, only weights connected to the output neurons (W^{out_gen}) require training. In contrast, all other weights are often randomly initialized and remain unchanged during training.

A. ESN TRAINING

The training of the basic ESN involves finding the optimal value of the weight matrix: $W^{out\_gen} = \{W^{inout}, W^{out}, W^{outout}\}$. This is often accomplished by least square linear regression technique which aims to minimize the error between the network's output and the target output signal.

First, the ESN weights: $\{W^{in}, W^{r}, W^{back}\}$ are randomly initialized. However, for the reservoir weight, W^r, it is important that it is generated such that the echo state property (ESP) is ensured. The ESP postulates that the influence of a previous state $x(t)$ and a past input $u(t)$ on a future state $x(t+n)$ should diminish slowly with time and not continue or even increase. According to Yildiz *et al.* [21], the ESP is often met if W^r is scaled such that its spectral radius, which is the largest absolute eigenvalue of W^r is less than 1. Other parameters that need to be initialized include the reservoir size N, the input/output scaling, the input/output shift, presence or otherwise of optional weights among others. Subsequently, the network is fed with the training sequence: $[u(1), y(1); u(2), y(2); \ldots; u(T), y(T)]$ as input. An important term, although not directly related to the ESN that needs to be set is the washout time T_0. This is a short time set to avoid the influence of initialized random weights on the trained network. Thus, the network states obtained before T_0 are not considered in the training algorithm and are flushed out. Hence the training set considers points in the range $T_0 < t \leq T$.

Moreover, the zero states of the ESN is also initialized i.e., $x(0), y(0) = 0$. This is used to obtain $x(1)$. Furthermore, the reservoir states for the training set $x(t)$ is collected by using (1).

Before proceeding to the next steps, let us make some definitions. Assuming that the output activation function f^{out} from (3) is invertible, then the equation can be changed to a linear equation as:

$$f^{out^{-1}}(y(t)) = W^{out\_gen}[u(t); x(t); y(t-1)] \tag{4}$$

For the training samples, we can change (4) to its matrix version as:

$$Y = Q.(W^{out\_gen})^{\mathsf{T}} \tag{5}$$

Where, $Y = f^{out^{-1}}[y(T_0), y(T_0+1), \ldots, y(T)]^{\mathsf{T}}$, $Q = [q(T_o), q(T_0+1), \ldots, q(T)]$, $q(t) = [u(t), x(t), y(t-1)]$.

Subsequently, the values of the matrix Q with dimension $(T - T_0 + 1) \times (K + N + M)$ and the matrix Y with dimension $(T - T_0 + 1) \times M$ are collected. Finally, the output weights is then obtained via:

$$W^{out\_gen} = (Q^{-1}.Y)^{\mathsf{T}} \tag{6}$$

Where the inverse may be obtained by the Moore-Penrose pseudoinverse or modified so that the Tikhonov regularization can be applied [8].

Despite the easy training of the ESN, one of the challenges of the network is that of the optimal choice of parameters and topologies. In the next section, we define some of the common topology and parameter that need to be set in the initialization of the ESN.

B. ESN OPTIMIZATION PARAMETERS

This section lists some of the most common ESN parameters that are optimized in existing literatures. They include:

1) Reservoir size: The reservoir size N represents the number of neuron units within the reservoir. It is a very crucial parameter, since it decides the maximum number of possible connections within the reservoir (N^2). Several research works have proven that the reservoir size denotes the memory capacity potential of the ESN [22]. Jaeger [22] has suggested that N be in the range ($\frac{T}{10} \leq N \leq \frac{T}{2}$) with T as the length of training data. However, he has also noted that the choice of the optimal value for N should depend on the periodicity of the training data and learning task complexity. Thus, the choice of a good value for N is still a difficult task. A reservoir that is too small may cause model inaccuracy, while a too large reservoir may lead to slow training and data overfitting. In overfitting, the network model will work excellently well on its training data but very poorly on ''unseen'' data.
2) Spectral radius ($\rho(W^r)$ or SR): This is another critical ESN parameter that needs to be initialized. It is described as the maximum absolute eigenvalue of the reservoir weights (W^r). It scales the width of the distribution of non-zero entries in the W^r matrix. Additionally, it defines the length of memory the reservoir can retain. The larger the spectral radius, the longer the previous inputs can influence the present output. It is often set up by first randomly generating a sparse W^r; its spectral radius is then computed as $\rho(W^r)$; W^r is then divided by $\rho(W^r)$ to give a matrix with a spectral radius of unity; this created matrix is subsequently scaled with the absolute spectral radius obtained via a fine-tuning process. According to Jaeger [23], setting the $\rho(W^r) < 1$, in most cases guarantees echo state property (ESP). However, he highlighted that for non-zero inputs, the ESP may still be met for $\rho(W^r) > 1$. As a rule of thumb, the spectral is set higher for tasks that require extended memory of inputs.
3) Input/Output scaling: The input weight (W^{in}) scaling is also a relatively important ESN parameter that needs to be chosen carefully. It influences the level of the linearity of the responses of reservoir units. For a W^{in} that is uniformly distributed, the input scaling b is referred to as a range $[-b; b]$ from which values of W^{in} are drawn. If values of W^{in} are normally distributed, then the standard deviation may be used as a scaling factor. To reduce the number of tunable ESN parameters, all columns of W^{in} are scaled together. However, the first column of W^{in} which represents the bias input may be scaled differently. Moreover, it is suggested that scaling optimization of the remaining active inputs be done separately if they contribute to the task in distinct manners. Large absolute values of W^{in} signify a network that is heavily driven by the input. Whereas, larger values could push the internal units near the saturation of the sigmoid activation function, thus causing a more nonlinear behavior of the model. Moreover, extremely large values of W^{in} drive the internal units to closer to -1/+1, binary dynamic behavior. In contrast, small absolute values of W^{in} denote network states that are only marginally excited about the reservoir unit zero states. Thus, the reservoir neurons function around the linear middle portion of the sigmoid, resulting in a near linear dynamic network. Since the input scaling controls the linearity of reservoir units, its value is set based on the linearity of the task at hand, However, it is often difficult to determine the linearity of tasks, thus the value is often set via trial and error. In ESNs that have the feedback connection weights (W^{back}), the feedback weight scaling has similar effects on the reservoir as the W^{in}.
4) Input/Output shift: This parameter often needs to be set in the initial ''baking'' of the ESN. It is an optional constant value added to the input vector $u(n)$. Although this parameter has gained prominence in early ESN publications, according to Lukoševičius [23], scaling of the input/feedback output without shifting generally suffices. Thus, the shifting may be avoided without any harm. However, in cases where both scaling and shifting is applied, it should be noted that the shifting factor has similar effect on reservoir linearity as scaling. Moreover, another important use of the shifting is in handling the case of symmetric-input fallacy.

This problem occurs because the sigmoidal units in the reservoir are symmetric devices and as such, if they receive input sequence $u(n)$ and produce output of $y(n)$, then an input of $-u(n)$ would give $-y(n)$. Thus, it is nearly impractical to produce output of $y(n) = u(n^2)$ type for inputs that take both positive and negative values. Hence, to handle this problem, the inputs are shifted so that all the input sequences are now positive.

5) Type of readout function: This represents the technique used to obtaining the output weights $\left(W^{out\_gen}\right)$. Although Lukoševičius and Jaeger [8] highlighted that any of the established methods for obtaining readout in machine learning may be adopted. However, for the ESN considered in this work, the choice is often between numerical stability and cost of computations. Thus, authors have chosen either the direct Moore-Penrose pseudoinverse in (6) (linear regression) or the ridge regression [20], [24], [25]. The direct Moore-Penrose inverse has higher numerical stability but is more computationally expensive compared to the ridge regression [8].
6) Regularization parameter of ridge regression (reg.): Regularization is often aimed at reducing the noise sensitivity of the network and also to prevent overfitting [26]. When ridge regression is employed, the regularization parameter controls the amount of regularization needed, and is clearly a design decision.
7) Reservoir activation function (Ra): For the ESN, the reservoir activation is non-linear function such as sigmoidal functions. In most works, the function of choice has been the *tanh*(.) or positive logistic *sign*(.). Also, some authors like Wang *et al.* [27] have recommended the application of different activation function within the same reservoir to improve the richness of the network.
8) Leaking rate (a): This parameter is associated with leaky integrator ESNs (LI-ESNs) [28]. These are ESNs whose reservoir neurons perform leaky integration of their activations from past steps of time. In these ESNs, an important parameter to be set is the leakage rate a which determines the momentum of the dynamics [29].
9) Noise scaling: This step is also called noise immunization and is applied to models trained with noiseless data. This is because networks trained with clean dataset for one step time forecasting diverge quickly in generative mode. A remedy to this issue is often the introduction of a scaled noise to the reservoir state $x(n)$ during training [11]. As a result, the generator can learn how to obtain the target output from a region of the present state $x(n)$ since it has seen a noisy version of it in the training phase. Setting the appropriate value for this noise scaling is a complicated task and is usually a compromise between accuracy of the prediction and model stability. Another solution to clean data is the use of regularization from ridge regression which is computationally less expensive or the pruning of W^{out}.

C. ESN OPTIMIZATION TOPOLOGIES

In this work, we define topology as any decision making that affects physical shape/architecture of the ESN. These include:

1) Presence/absence of optional weights. As shown in Fig. 2, the optional weights in the ESN architecture include: the input to output weight connection (W^{inout}), the feedback weight connections from output back to reservoir (W^{back}), and the cycle between the output units (W^{outout}). Whether these optional weights are included or not depends on the application and an often compromise between accuracy and computation costs.
2) Reservoir connectivity: This denotes the number of non-zero values/connections within the reservoir weights (W^r). Although a low priority parameter [23], its value is known to influence the complexity of the reservoir. ESN research works recommend a sparsely connected reservoir, i.e., a reservoir in which most values of the weight matrix (W^r) are set to zero [15]. Reservoirs with sparse connectivity have been reported to perform a little better than heavily connected ones. Too much connections within the reservoir may reduce the ESNs ability to be trained properly. This is because many connections may cause strong coupling effect of the reservoir neuron states, and thus decrease the diversity of the reservoir states. Typical values connectivity of the is between 0.01 and 0.2 [30].
3) Presence/Absence of input bias: The presence/otherwise of the input bias and its connectivity to other layers if it is present is often optimized. Its presence often helps in handling symmetric input-fallacy (Explained in the Input/Output shift parameter in Section III-B).
4) Connectivity of output weights: Another item often optimized is the connection of output weights (W^{out}).

It is clear from the aforementioned parameters and topologies, that choosing the best values for a particular task by "hand" is impractical. Moreover, since a brute force approach is also out of the question due to the large search space, researchers have resorted to metaheuristic methods [31] to find near optimal values for these parameters/topologies within a reasonable computational time. In the next section we shall briefly introduce metaheuristic methods and discuss typical types of them.

IV. METAHEURISTIC OPTIMIZATION TECHNIQUES

Metaheuristics are a primary sub class of stochastic optimization methods. They are iterative techniques and algorithms that use some level of randomness to discover optimal or near-optimal solutions to computationally hard problems. These techniques come in handy in problems that are generally difficult to solve. Especially where there is no principled method of traversing the solution search space and obtaining the optimal solution, but there exists is a way to evaluate the "goodness" of obtained solutions [32].

Metaheuristics may be broadly classified into three: the single-solution based, population-based, and hybrid methods. In all these varieties, the search space is explored and exploited intelligently so that good solutions can be found. In the single-solution based techniques, such as the simulated annealing (SA) [33], there is only one solution that gets tweaked/perturbed so as to obtain an optimal solution. In contrast, in the population-based methods like genetic algorithm (GA) [34], there is a pool of solutions whose elements interact and also get tweaked in each iteration in order to find better solutions. Finally, hybrid methods would combine two metaheuristics together, taking from the strength of each [31].

Metaheuristics have been employed to tune/train the parameters/topology of neural network for a very long time [35]. However, this review is concerned with metaheuristics that are used to optimize the ESN.

V. WORKS ON ESN OPTIMIZATION

In this section, we discuss works that optimize the ESN by employing metaheuristic algorithms (MAs). These works could be divided into four classes. As shown in Fig. 3, the first class are those that optimize the global parameters of the ESN such as the spectral radius (SR), and the scaling of weights. The Second class are works that optimize the topology/architecture of the network, such as weight connections. The third group are those that attempt to employ MAs to train the output weights of the ESN often in unsupervised tasks. The fourth group of papers would do a hybrid of the other groups. The works are categorized based on the MA they employ to optimize the ESN.

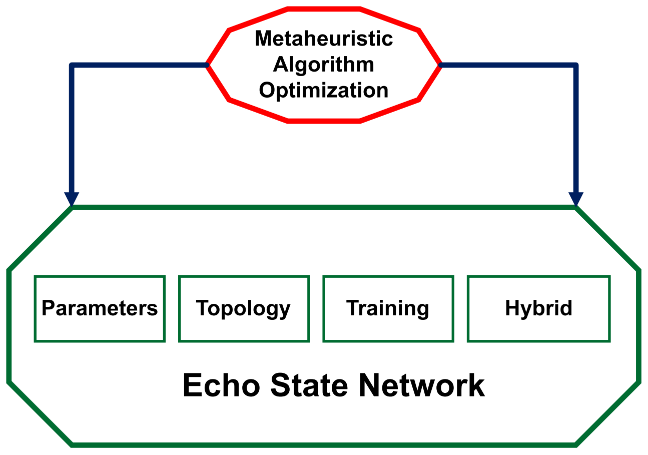


FIGURE 3. A schematic of application of metaheuristics in ESN optimization.

A. STATE REPRESENTATION, MOVES AND OBJECTIVE FUNCTIONS

In this section, we briefly present how most of the reference works represent solutions, make moves and the mostly employed objective function(s).

1) PARAMETERS OPTIMIZATION

In this case, the state representation is a single dimensional vector, with length equal to the number of parameters to be optimized. E.g., $[P_1, P_2, P_3, \ldots, P_d]$, where P is the parameter to be optimized and d is the number of parameters to be optimized.

a: INITIAL SOLUTION

The first set(s) of solutions are often randomly initialized to take values between the minimum and maximum value each parameter can have. Here, we would denote these values as $P_{i,min}$ and $P_{i,max}$.

b: MOVES/PERTURBATIONS

This denotes how the algorithms traverse the design space. In some works [36], [37] some parameter values are broken into discrete parts between $P_{i,min}$ and $P_{i,max}$. For example, if the spectral radius (SR) of the reservoir has: $P_{i,min} = 0.1$ and $P_{i,max} = 0.9$, then the values the SR can take may be broken into $\{0.1, 0.5, 0.9\}$ and then each of the value may be chosen based on a uniformly distributed probability. In other works, the value the parameter can have is continuous from $P_{i,min}$ and $P_{i,max}$, often represented as: $(P_{i,max} - P_{i,min}) \times rand + P_{i,min}$. Where the function *rand* produces uniformly distributed random numbers between 0 and 1. The actual moves to be made per iteration would then depend on the individual metaheuristics.

2) TOPOLOGY OPTIMIZATION

In this case, weight connections are often represented as a two-dimensional binary matrix. For example, in matrix B, entry $B_{i,j}$ denotes connection between neuron i and j, a "1" in the location will signify the presence of a connection between i and j, while a "0" will represent the absence of a link. In the case where the presence/absence of weights/bias are optimized, the representation is often a one-dimensional matrix, with "0" indicating presence and "1" denoting absence.

OBJECTIVE FUNCTION(s)

Since the training of the ESN is generally fast because only W^{out_gen} weights are trained, the objective function mostly employed are those denoting prediction accuracy of the ESN such as: mean squared error (MSE), root mean squared error (RMSE), normalized RMSE (NRMSE), mean absolute error (MAE), mean absolute percentage error (MAPE) etc. However, in this review, we suggest the adoption of the "goodness" measure of the reservoir as an objective function, because we believe that this will speed-up the "baking" process of the ESN. In the next section, we discuss works that employ MAs to optimize the ESN.

B. EVOLUTION STRATEGY (ES)

The group of Evolution strategy (ES) based algorithms are population-based metaheuristics, first invented in 1973 by Rechenberg [38]. It basically involves a simplified method of selection and often employs only mutation as the perturbation operator. A comprehensive review of ES and its variants can be found in Hansen [39].

ES has been employed to optimize the ESN. For e.g., van der Zant *et al.* [40] used the ES to select the values of three parameters of the ESN: SR, N & connectivity of reservoir units (C). They tested the method on the motion prediction of the twin-burger autonomous under water robot. Results they obtain show that their technique outperformed other methods as well as an evolutional algorithm (EA). Similarly, the same authors extended their work in Ishu *et al.* [41]. Here they added the optimization of all the network weights: W^{in}, W^{r}, W^{back}. Moreover, they employed an evolutional algorithm (EA) to optimize the weight values on the same twin-burger autonomous robot. Comparing the new technique with their earlier method shows the superiority of the new technique. Similarly, Devert *et al.* [42] proposed an ES strategy based ESN for unsupervised learning tasks. They claim to be the first to employ the echo state network in an unsupervised task. Moreover, the training of the ESN was turned into an optimization problem, and the ES strategy was employed to optimize the network and obtain the output weights. Testing the technique on two benchmark problems showed that the method was able to perform better than neuroevolution of augmenting topologies (NEAT) [43] in most cases. In a related work, Jiang *et al.* [44] proposed a covariance matrix adaptation ES (CMA-ES) [45] method for training the ESN's output weights for unsupervised tasks. Testing the technique on a supervised task and unsupervised tasks of the double pole balancing control problem shows the good results over compared methods. Moreover, Hartand *et al.* [46] proposed an ESN based autonomous robot controller with counting capabilities. The CMA-ES was employed to optimize the ESN. The approach was compared to NEAT [43] on the Tolman maze problem and it was found to be competitive with less tuning parameters.

C. PARTICLE SWARM OPTIMIZATION (PSO)

PSO is another population-based MA inspired from the flocking and swarming attributes of animals. Developed in 1995 by Eberhart and Kennedy [47], it is known to have no selection procedure but rather a fixed population of solutions called particles that are perturbed as more discoveries are made of the search space. Each particle's new position is affected by its new velocity. A general overview of the PSO algorithm and its variants can be found in Alam *et al.* [48].

The particle swarm optimization (PSO) and its variants have been one of the most widely employed metaheuristics to optimize the ESN. In Zhou *et al.* [49], PSO was employed to optimize four parameters of the leaky integrator ESN (LI-ESN). The parameters are: spectral radius, value ranges of W^{in} and W^{back} and the time constant (τ) of the system. The PSO optimized LI-ESN was tested on the motion prediction of a monkey's wrist. Simulation results showed the power of the technique. Similarly, Song *et al.* [50], [51] demonstrated that the classical ESN cannot guarantee asymptotic stability for closed-loop ESNs i.e., ESNs with W^{back}. They highlighted further that existing solutions of ridge regression and noise immunization are not easy to set up. Thus, they proposed a PSO-based training approach for the closed-loop ESN. They applied the technique on the ''figure-eight'' generation task. Subsequently, simulation results showed that the proposed method achieved better prediction and stability compared to the classical ESN. Furthermore, in a similar work, Song *et al.* [52] employed PSO to train the output weight of the LI-ESN but this time they tested on the multiple superimposed oscillator (MSO) problem. The approach was compared to the DESN [53] and Evolino [54] and it was found to be better. In a similar work, Zhou *et al.* [55] employed PSO to optimize the ESN's parameters for modelling the pneumatic artificial muscle (PAM). The results they obtained showed that their method outperformed the back-propagation neural network (BPNN) in terms of precision and speed. Similarly, Sergio and Ludermir [36] proposed a PSO-based technique for optimizing the ESN. They compared their method with an exhaustive search on five time series prediction tasks and found their technique to be competitive.

Furthermore, Rabin *et al.* [56] proposed a PSO optimized ESN for electricity load forecasting. They proposed a multi-reservoir ESN and employed the sensitivity oriented linear learning [57] to obtain the output weights. Subsequently, the parameters of the new ESN are optimized by PSO. Two additional parameters related to the learning scheme were added to the parameters optimized by PSO. Testing the methods on the load time series prediction task produced results that outperformed the GA optimized SVM and other techniques. In a related work, Yang *et al.* [58] proposed a new ESN architecture. The new ESN consists of fixed cyclic neurons with fixed feedback connections. Furthermore, the novel ESN is applied for the spectrum prediction of cognitive radio. The improved PSO (θ-PSO) [59] was employed to optimize the ESN parameters. Test results on time series data and spectrum prediction task showed that the new ESN is comparable to the classical ESN and Elman [60].

Wang and Yan [61] developed a binary PSO (BPSO) based approach to optimize the output weight connections of a trained ESN. They highlighted that since the reservoir is sparsely connected, it is contradictory to connect all the reservoir neurons to the output. The data is divided into three. Training data is first used to train the ESN. The validation data is then employed to optimize the W^{out} by PSO. Its performance is then tested on a third unseen data called the test data. The technique outperformed the classic ESN and of least angle regression (LAR) method on three classes of time series. In contrast, Basterrech *et al.* [62] proposed a PSO optimized ESN. In the method, a subgroup of the reservoir weights are picked and their values are optimized by PSO. The method has less computation effort since the computation of spectral radius is not needed. The technique outperformed the classical ESN. Moreover, Chouikhi *et al.* [63] proposed a PSO approach to extend the work of Basterrech *et al.* [62]. In the method, the ESN is first trained in the normal way.

Then a subset of non-zero reservoir weights, input weights, and feedback weights are selected and their values are optimization. The PSO optimized weights are then reinserted into the trained ESN. They highlighted that in some cases, optimizing only the value of a subset of the reservoir weights may suffice as in [62]. The technique has less computational effort since only a portion of each weight matrix is selected for optimization and the calculation of spectral radius is not needed. The method performed better than the classic ESN and Basterrech *et al.* [62] on the Mackey-Glass time series. In a related work, Chouikhi *et al.* [64] proposed a single objective and multiobjective PSO optimized ESN. In the single objective technique, only the mean squared error (MSE) between the target and actual output predictions is minimized. In contrast, in the multiobjective method, both the MSE and reservoir complexity are optimized. The reservoir complexity is represented by the reservoir size and the reservoir connectivity. The output of the single objective PSO is a single solution while that of the multiobjective PSO is a set of Pareto solutions. Both techniques perform better than many existing methods in predicting future values of the NARMA and Lorenz time series. Recently, Chouikhi *et al.* [65] presented a two level optimization approach to the single and multilevel ESN [66]. The target application was feature extraction from a large dataset. In the first level, a multiobjective PSO is used to optimize the ESN parameters targeting error minimization and reduction of network complexity. This optimization produces a Pareto front. These Pareto front solutions are then subjected to a second weight optimization using single objective PSO that aims only at error minimization. The features extracted from the optimized ESN are then inserted into an SVM classifier. Results obtained from the optimized ESN feature extractor was shown to perform competitively in complex classification tasks.

Similarly, Xu *et al.* [67] proposed a hierarchical neural network (HNN) for multivariate time series prediction task. The HNN consists of a simple cycle reservoir (SCR) ESN [68] at the first stage and the ELM at the second stage. PSO algorithm is then employed to optimize the leaky rate, and scaling of the input weights and reservoir weights of the SCR-ESN. Testing the HNN on two real world time series showed good results.

In a similar work, Wu *et al.* [69] proposed a parallel PSO (P-PSO) to train the ESN's output weights. They highlighted that the centralized training of the ESN is unsuitable for most big data applications where the data is distributed over different devices. The new technique was implemented on the spark framework over four datasets. In most cases, it was discovered that the P-PSO based method outperformed its competitors. In contrast, Sheng *et al.* [70] proposed a granulated ESN for the prediction of interval construction. They view the ESN training as an optimization problem and used PSO to obtain the output weights. Implementation of the technique over the map-reduce (MR) frame work on data time series and industrial data showed the efficacy of the method. In Chouikhi *et al.* [71] the work in [63] was extended. In the new work, a subset of the input, reservoir and feedback weights are selected and optimized by PSO in a pretraining phase before the actual network training process. The extension involved more tests over additional datasets and more comparisons with other works. The technique further showed interesting results.

Wei and Haitian [72] proposed a PSO optimized ESN for the electric load prediction. The PSO is used to optimize four parameters of the ESN. Testing the method on power and climate data prediction tasks showed better performance over the SVM, BPNN and classical ESN. Moreover, Salah *et al.* [73] proposed a PSO optimized ESN for the remaining useful life (RUL) prediction of a turbofan engine. The PSO is employed to optimize the input/output shift, input/output scaling, the number of reservoir units and the spectral radius. The obtained result outperformed the classical ESN. Furthermore, Abdelbari and Shafi [74] modified the ESN for the conceptual modeling of complex dynamical system. They employed GA, PSO, and DE to optimize the parameters of the ESN. The techniques were tested on the modeling of four complex systems and the GA optimized network performed better.

D. GENETIC ALGORITHM

GA is another famous population-based MA that is inspired from natural selection. The first GA was invented by Holland [75]. Although there are several versions of GA, they will be distinguished by the type of selection, crossover, or mutation functions they employ. Details of GA and many of its variants can be found in [76].

GA has been employed in several works to optimize the ESN's parameters/topology. This section discusses some of these works. Xu *et al.* [77] proposed an ESN based direct adaptive controller for non-linear dynamical systems. The training of the ESN is turned into an optimization problem and GA was employed to solve it. Comparing the technique with the PID controller showed the efficiency of the new technique. In contrast, Ferreira and Ludermir [37] claims to be the first work that optimized the ESN by GA. They employ GA to optimize the parameters and topology of the ESN including the type of reservoir activation function to be used i.e., either *tanh* or *signum*. Comparing the performance of the technique on an hourly wind speed series shows better results compared to an exhaustive search and a previous work in [78]. Furthermore, Ferreira and Ludermir [24] proposed another GA for the ESN optimization. It is an extension of their work in [37]. Here the parameters to be optimized include the type of readout training to be adopted i.e., either pseudo inverse or ridge regression, regularization parameter of the ridge regression, leak rate and the connection(s) of the bias unit. Testing the method on wind speed time series prediction task, they found the method to outperform the AG search which optimizes only the trio of: spectral radius, reservoir size, and reservoir connectivity. In an extension of the work, Ferreira *et al.* [20] improved the comparison benchmarks for the technique called the RCDESIGN. The method was further tested on two time series data. Again, the technique outperformed the AG search. One of the fascination of the works in [24] and [20] is the fact that since the spectral radius is not optimized, it does not have to be computed, thus saving a lot of computation time. Moreover, they highlighted that the concept of adjusting the spectral radius within a unit circle in the complex planes is derived from linear system theory and may not be applicable in non-linear systems.

In other works, Bianchi *et al.* [79] proposed an ESN for forecasting telephone calls load. GA was employed to select the network's parameters as well as the picking of the most suitable additional signal to be added as an external input to the ESN. Three training techniques were used: the least square regression (linear regression), linear support vector regression (SVR) and non-linear SVR. The best methods were the linear regression and the SVR with a Gaussian kernel. In a related work, Bianchi *et al.* [80] employed a GA optimized ESN for electric load forecasting. The method was able to outperform the general ARIMA [81] method. In addition, Deihimi and Showkati [82] proposed an ESN approach for short term electric load forecasting. They employ GA to optimize the reservoir size and spectral radius of the ESN. Similarly, Løkse *et al.* [83] presented a novel framework for ESN training. Their technique improves the ESN's generalization abilities via a regularization constraint brought in by a smoothening effect of a dimensionality minimization process. GA was employed to optimize the parameters of the ESN. In Zhong *et al.* [84] a double reservoir ESN (DRESN) for the prediction of multi-regime time series is proposed. GA was employed to select the quantity of reservoir units and the spectral radii of both reservoirs. Testing the technique on turbofan multi-regime time series prediction produced interesting results. Moreover, Ma *et al.* [85] highlighted that the classical ESN may have challenges in predicting time series with multiscale structures. Thus, they proposed a hierarchical ESN, which consists of several layers of ESN. They named the method as Deep-ESN. Subsequently, GA is employed to optimize the number of reservoir units, spectral radii and leak rate of the reservoirs.

E. DIFFERENTIAL EVOLUTION

DE is another population-based MA. It is a type of evolutionary computation intended for multi-dimensional real valued search. The original DE was invented by Storn and Price [86] and is most commonly known for its adaptive mutation function that depends on the variance of solutions in the population. More details on DE and its types can be obtained from Das *et al.* [87].

DE [86] and its variants have seen many application in engineering optimization. They have also been employed to optimize the ESN. Zhang *et al.* [88] presented a DE optimized ESN for time series prediction. Four parameters of the ESN are selected for optimization, which are: size of the reservoir, its spectral radius, its connectivity and the scaling of input weights. Testing the technique on Lorenz system showed the method performing better than the classical ESN. In a similar work, Rigamonti *et al.* [89] proposed a DE optimized ESN for forecasting the remaining useful life of (RUL) of industrial systems. Eight parameters of the ESN were selected for optimization including the output weight shifting and scaling. The method was applied to predict the RUL of turbofan engine and fascinating results were obtained. In a more recent work [90], the same authors presented an ensemble of ESNs for the RUL prediction of industrial equipment. Again, they employed DE to optimize the ESNs. Similarly, Yang *et al.* [91] proposed a DE based technique for optimizing the ESN. First, the reservoir weights are constructed through the singular value decomposition (SVD) [92]. Subsequently, the singular values of the reservoir are then optimized by the DE algorithm. The method outperformed many existing techniques on time series prediction tasks. In a recent work, Wang *et al.* [93] devised a DE optimized ESN for electrical energy consumption prediction. DE is employed to optimize the reservoir size, its connectivity and spectral radius. Testing the technique on real life data showed fascinating results.

F. OTHER METAHEURISTICS

Some other works that have employed MAs for ESN optimization include Cui *et al.* [94] where a biogeography-based optimization (BBO) [95] is employed to optimize the ESN. The ESN was incorporated with other soft computing methods to solve the problem of obtaining the vinyl chloride monomer (VCM) conversion rate for real time online measurement in polyvinyl chloride (PVC) production. In contrast, Amaya and Alvares [96] presented an artificial bee colony (ABC) [97] to optimize the ESN for predicting the RUL of turbofan engines. Their technique outperformed that in Peng *et al.* [98]. Duan *et al.* [99] proposed an orthogonal pigeon-inspired optimization (OPIO) [100] algorithm to optimize the ESN for image restoration tasks. OPIO was used to optimize the size of the reservoir, its connectivity, spectral radius and the input weight scaling. Moreover, they compared the technique with other image restoration methods and ESNs optimized by other evolutionary methods. The proposed method was found to be competitive. In a related work, Han *et al.* [101] employed the quantum-behaved fruit fly optimization algorithm (QFOA) to optimize four parameters of the ESN. The target application is network traffic prediction. Their method was able to outperform the harmony search (HS) [102], PSO, black hole (BH) [103] optimized ESN. Recently, Bala *et al.* [104] employed the modified cuckoo search (MCS) algorithm [105] to optimize the parameters of the ESN. They targeted the reservoir size, its connectivity and spectral radius. The technique was able to outperform the classic ESN in predicting future values of the Mackey-Glass time series.

In a related work, Morando *et al.* [106] developed a big bang-big crunch (BB-BC) [107] optimized ESN for fault diagnosis of fuel cells. Similarly, Deihimi and Rahmani [108] presented an ESN based technique estimating distortions in the voltage harmonic waveforms at sensitive loads that are non-monitored. The grey wolf optimizer (GWO) [109] was employed to optimize three parameters of the ESN. The method was found to outperform other NN techniques. Similarly, Pan *et al.* [110] proposed a bat algorithm (BA) [111] optimized ESN for forecasting internet traffics. The BA was employed to optimize the values of the input weights. Testing the technique on four datasets produced interesting results.

G. HYBRIDS

In hybrid metaheuristics, often two metaheuristics are combined to complement each other [31]. This is deliberately done so that the hybrid algorithm gains from the strength of each individual algorithm [112]. Thus, some works have attempted to use hybrid algorithms for the ESN optimization. In [25], a PSO and SA hybrid is proposed for the optimization of the ESN. Testing the method on five benchmarks produced good results. Similarly, Xu *et al.* [113] presented a hybrid of PSO and tabu search (TS) to optimize the ESN for wind power forecasting. Testing on a wind farm data produce showed the competitiveness of the method.

H. SUMMARY TABLE

Table 1 summarizes the works we have reviewed in this paper. The table highlights the type of algorithm employed, the area of the ESN it attempts to optimize, the cost function it employs, the application it was tested upon and with which other methods the technique was compared to. In the table term ‘‘params’’ denote parameters, and the term ‘‘Topo.’’ represents topology. Additionally, the term ‘‘Connect.’’ denotes connectivity, meaning that the connectivities are optimized. Notice that we have added the connectivity (C) in the ‘‘params’’ category not because it is a parameter, but because it is commonly optimized and we want to aim for brevity. Moreover, the term ‘‘ON/OFF’’ denotes presence or absence. Most of the parameters/topology optimized in the table have been explained in Section III.

VI. TRENDS & RESEARCH GAPS

This section presents some of the conclusions drawn from the survey and possible research directions.

A. TRENDS

The trends identified from the reviewed papers include the following:

- From the several papers reviewed in Section V we can confirm the conclusions reached in [8] that most of the reservoir computing optimization and improvements are in the dynamic reservoir itself. Moreover, it also reaffirmed the notion that the construction of a random fixed reservoir is not ‘‘good’’ enough. Therefore, the paradigm shift is that the reservoir computing techniques are methods that have different approach for the reservoir construction from that of the readouts.
- Furthermore, metaheuristics have been greatly used to optimize the ESN’s parameters. The mostly optimized parameters are: reservoir size, reservoir density/connectivity and its spectral radius.
- Additionally, as highlighted in [8] that the condition that the spectral radius be less than one for echo state property is not absolute, particularly for non-linear learning tasks. Thus, few works avoided finding the spectral radius which is computational expensive and they did just find by employing metaheuristics to optimize the reservoir.
- Another interesting finding is the fact that some works is were able to employ metaheuristics to obtain the output weights (training) even in supervised leaning tasks and they found interesting results.
- The PSO is the mostly used metaheuristics to optimize the ESN. Perhaps, this may be a coincidence if we give credit to the ‘‘no free lunch’’ notion [143], which explains that, if every non re-sampling optimization algorithm is used to solve all optimization problems, on average, the performance across all the problems will be the same. Or it is possible that the PSO is indeed versatile enough as the choice optimization algorithm for reservoir computing techniques.

B. RESEARCH GAPS

In this section we present research gaps available on the application of metaheuristic in ESN optimization.

1) ENCODING PROBLEMS

Solution encoding/representation in many of the reviewed research works have been simplified to avoid complications in the algorithm development and achieve a faster convergence due to a smaller search space. This simplification is often achieved by optimizing only few parameters and topologies of the ESN. However, Ferreira *et al.* [20] and its related works have taken the bold step of merging the parameters and topology optimization into a single encoding that is not simplified. This encoding is very versatile as it aims to almost fully optimize the ESN. Nevertheless, is poses additional problems due to the large search space it attempts to explore and exploit. Moreover, perturbation/movement of solutions within the algorithm and combination of two solution as in crossover of GA often leads to infeasible solutions. These invalid solutions must be validated or fixed so that they become feasible solutions. These problem(s) opens an interesting research area of developing of better/new encoding schemes that will avoid these issues.

2) MULTILEVEL ESN

The development of the multilevel ESN by Malik *et al.* [66] has brought about more parameters and topologies to be optimized. One interesting area of optimization is the interconnection between the levels of reservoir. Another fascinating area of research in the multilevel ESN is the optimization procedure to be adopted. The question is whether individual levels of ESN should be optimized first before the

TABLE 1. Summary of selected literature for ESN optimization.

Ref.	Parameter/Topology Optimized	MA	Cost function	Application (Test)	Comparison
Van et al. [40]	Params(SR,N,C)	ES	RMSE	Twin-burger robot motion	Brute force, ARX, FFNN, EA
Ishu et al. [41]	Params(SR,N,C) Topo. (connect.:W^{in},W^{r},W^{back})	ES,EA	RMSE	Twin-burger robot motion	Brute force, ARX, FFNN, [40]
Devert et al. [42]	Train(W^{out})	ES	MSE	Flag image problems Artificial embryogeny model	NEAT [43]
Jiang et al. [44]	Params(SR), Train(W^{out})	CMA-ES	MSE Problem related	Univariate time series Double pole balancing	[11], NEAT [43], AGE [114]
Hartland et al. [46]	Params(N,C,SR)	CMA-ES	MSE, problem related	Tolman maze problem	NEAT [43]
Zhou et al. [49]	Params(SR,W^{in},W^{back},τ)	PSO	Modified MSE	Monkey wrist trajectory	–
Song et al. [50], [51]	Train(W^{out})	PSO	Modified MSE,	Figure-8 trajectory	Classical ESN
Song et al. [52]	Train(W^{out})	PSO	Modified MSE	MSO	Evolino [54], DESN [53]
Zhou et al. [55]	Params(SR,W^{in},C)	PSO	RMSE	Pneumatic artificial muscle modeling	–
Sergio et al. [36]	Params(N,SR,Ra) ON/OFF(W^{inout},W^{back})	PSO	MSE	Database of time series: MEMTES, NARMA Mackey-Glass	Exhaustive search APSO [115], EPUS-PSO [116]
Rabin et al. [56]	Params(N,SR,C, W^{in},W^{back})	PSO	NRMSE, MAPE	Electric load time series	GA-SVM [117], WNN [118] NN-Fuzzy, ARIMA
Yang et al. [58]	Params(N,SR,C,W^{in})	PSO	NRMSE	NARMA Santa Fe laser series Lorenz, Mackey-Glass Simulated spectrum data	Classical PSO-ESN, [60]
Wang et al. [61]	Topo.(connect.:W^{out})	BPSO	NRMSE, NMSE	NARMA Santa Fe laser time series Mackey-Glass time series	Classical ESN, LAR [119]
Basterrech et al. [62]	Params(W^{r})	PSO	MSE	Santa Fe laser time series NARMA-10, NARMA-30	Classical ESN
Chouikhi et al. [63]	Params(W^{in},W^{r},W^{back})	PSO	MSE	Mackey-Glass	Classical ESN

Chouikhi et al. [64]	Params(N,C)
	Params(W^{in},W^{back})
		PSO	MSE, RMSE
			Reservoir complexity
				NARMA, Lorenz	PSO-ESN [62], SVM [120], ABC-BFNN [121]

Chouikhi et al. [65]	Params(N,C)
	Topo.(W^{in},W^{r},W^{inter})
		PSO	RMSE
			Reservoir complexity
				ECG200 data
				Coffee data
				Breast cancer data
				ECG five days data
					DE [122], HS [102], BBO [95] FA [123] ABC [124], FS [125]

Xu et al. [67]	Params(W^{in},W^{r})	PSO	Pearson corr. coeff. RMSE	Temperature & rainfall data	ELM [126], SVESM [127], SCR [68], Classical ESN
Wu et al. [69]	Train(W^{out})	P-PSO	NRSME	NARMA, Lorenz, Smart electric meter data, MNIST	PSO, ADMM [128], CNN [129]
Sheng et al. [70]	Train(W^{out})	PSO	problem related	Mackey-Glass, Noisy MSO Blast furnace generation flow Coke oven consumption flow	Delta MLP, MVE MLP, Bootstrap MLP, MR-based, Bayesian MLP

TABLE 1. ***(Continued.)*** **Summary of selected literature for ESN optimization.**

Ref.	Parameter/Topology Optimized	MA	Cost function	Application (Test)	Comparison
Chouikhi et al. [71]	Params(W^{in},W^{r},W^{back})	PSO	MSE, RMSE	Mackey-Glass, NARMA Henon attractor, Lorenz attractor S & P 500 financial data Joint articulation angles dataset Magnetic levitation data	Classical ESN, ABC, GA, DE, other PSOs [63]
Wei et al. [72]	Params(N,C,SR,W^{in})	PSO	MSE, NRMSE	Power and climate data	SVM, BPNN, classical ESN
Salah et al. [73]	Params(N,SR,W^{in},W^{back})	PSO	Modified NRMSE	Turbofan engine time series	Classical ESN
Abdelbari et al. [74]	Params(N,SR,C,W^{in},W^{back})	PSO, GA & DE	NRMSE + C	Workforce Inventory System Lotka-Volterra System Van der Pol Oscillator System Lorenz	Classical ESN
Xu et al. [77]	Train(W^{out})	GA	Modified MSE	SISO missile model	PID controller
Ferreira et al. [37]	Params(N,SR, Ra) ON/OFF(W^{inout},W^{back})	GA	MSE	Wind speed series	Exhaustive search, [78]
Ferreira et al. [24]	Params(W^{r},W^{in},W^{back},W^{bias},a) Params(readout,Ra,reg.) Topo. connect.(W^{r},W^{in},W^{bias}) ON/OFF(W^{inout},$W^{biasout}$) ON/OFF(W^{bias},W^{back})	GA	Modified MSE, MAE, MAPE	Wind speed time series	Classical search, [37]

Ferreira et al. [20]	Params(W^{r},W^{in},W^{back},W^{bias},a)
	Params(readout,Ra)
	Params(reg.)
	Topo. connect.(W^{r},W^{in},W^{bias})
	ON/OFF(W^{inout},$W^{biasout}$)
	ON/OFF(W^{bias},W^{back})
		GA	Modified MSE	NARMA, Mackey-Glass
				Average hourly wind speed
					AG Search [37]

Deihimi et al. [82]	Params(N,SR)	GA	MSE	Hourly electric load & temperature data	Hybrid: wavelet transform + neuro-evolutionary algorithm [130]

Bianchi et al. [79]	Params(N,C,SR,Noise,W^{r})
	Params(W^{in},W^{back},y_d)
	Params(Related to SVR training)
		GA	NMSE	Orange telephone dataset	ARIMA [131], TES [132],
					ARIMAX [133]

Bianchi et al. [80]	Params(N,SR,C,reg.,Noise) Params(W^{in},W^{back},W^{out}) ,	GA	NRMSE	Electric Load time series	ARIMA [81]

Løkse et al. [83]	Params(N,C,SR,W,W^{r})
	Params(W^{out},W^{in},noise)
		GA	NMSE	NARMA, Mackey-Glass,
				Lorenz, Moore-Spiegel
				MSO
					v-SVR [134], ESN variants

Zhong et al. [84]	Params(N,SR)	GA	RMSE	Turbofan engine- multi-regime time series	PSO-ESN,FA-ESN, SVR,BPNN
Ma et al. [85]	Params(W^{in}, SR, a)	GA	RMSE, NRMSE, MAPE	NARMA, Mackey-Glass Monthly sunspot series Daily minimum temperatures series	ML-ESM [66], [135], R^2P [136], classical ESN
Zhang et al. [88]	Params(N,SR,C,W^{in})	DE	NMSE	Lorenz	Classical ESN
Rigamonti et al. [89]	Params(N,SR,C,W^{in},W^{out},W^{back})	DE	problem related	Turbofan engine time series	FS, ELM [126]
Rigamonti et al. [90]	Params(N,SR,C,W^{in},W^{out},W^{back})	DE	problem related	Turbofan engine time series	Other ESN variants, FFNN
Yang et al. [91]	Params(W^{r})	DE	NRMSE	NARMA, MSO Mackey-Glass	PSO [63], classical ESN, GESN [92], [137]

TABLE 1. ***(Continued.)*** **Summary of selected literature for ESN optimization.**

Ref.	Parameter/Topology Optimized	MA	Cost function	Application (Test)	Comparison
Wang et al. [93]	Params(N,C,SR)	DE	MAPE	Electric energy consumption data	GA [84], ARIMA [131], BPNN [138]
Cui et al. [94]	Train(W^{out})	BBO	MSE	PVC industrial data	Other ESN variants
Amaya et al. [96]	Params(N,C,SR,W^{in},W^{out},)	ABC	Modified NMRSE	Turbofan engine time series	[98]
Duan et al. [99]	Params(N,C,SR,W^{in})	OPIO	NRMSE	Benchmark images	ForWaRD [139], GSR [140] RI [141], FTVd [142] GA, DE, ES, BBO
Morando et al. [106]	Params(N,C,W^{in}) Params(reg.) ON/OFF(W^{back})	BB-BC	MSE	DIAPASON data	–
Han et al. [101]	Params(N,C,SR,W^{in})	QFOA	RMSE, MAE	Traffic data	HS, PSO, BH
Pan et al. [110]	Params(W^{in})	BA	NMSE	Internet traffic data	WNN [118], ELM [126], BPNN SVR, VMD-ESN
Deihimi et al. [108]	Params(N,C,SR)	GWO	MSE	IEEE 37-bus network	MLP, RBF, FFTD-NN, WNN [118]
Bala et al. [104]	Params(N,C,SR)	MCS	MSE	Mackey-Glass	Classical ESN
Sergio et al. [25]	Params(N,a,W^{r},W^{in},W^{back},W^{bias}) Params(reg.,Ra,readout) ON/OFF(W^{inout},$W^{biasout}$,$W^{bias\text{-}r}$) ON/OFF(W^{back}, W^{outout})	(PSO + SA)	Modified MSE	NARMA, Mackey-Glass, MSO, DJIA, STAR	Classical ESN, APSO, [36]
Xu et al. [113]	Train(W^{out})	(PSO + TS)	MSE, MAE, RMSE	Chaotic wind power time series	BPNN, classical ESN

interconnections between the layers are optimized? Or should they be optimized concurrently?

3) FITNESS MEASURE

Since the ESN training is quite fast, most of the cost function employed in the algorithms are those that reduce the error between the predicted and actual outputs. However, another fascinating gap that need to be filled is the development of new cost functions that incorporates both the error minimization and the properties of the reservoir such as eigen value spread in the cost function.

Moreover, few works such as [65] have attempted to formulate the optimization problem as multiobjective, targeting the error function minimization as well as minimal reservoir complexity (reservoir size and connectivity). One interesting question is whether there are other properties of reservoir complexity apart from the reservoir size and its connectivity that need to be considered?

4) NEW METAHEURISTICS

Even at the time of preparing this manuscript, many new MAs are being developed with new interesting characteristics. These new MAs have been proven (using mathematical functions) to have better accuracy and convergence rate than their existing counterparts. Moreover, they are known to have fewer tuning parameters which simplifies their implementations. A large part of the research gap in the ESN optimization is the employment of these new MAs such as the earthworm optimization algorithm (EWA) [144] and pity beetle algorithm (PBA) [145] to optimize the ESN.

5) HYBRID METAHEURISTICS (HAs)

Another fascinating area of further research is the hybridization of existing MAs that have been employed to solve the ESN optimization with new MAs. Since HAs attempt to take from the good qualities of two MAs, we strongly believe that these HAs will further improve the ESN optimization.

VII. CONCLUSIONS

Reservoir computing is still a young field, although a lot have been done over the 20 years of its introduction, there is still a lot of space for development. While it came to ease the training process of recurrent neural networks, however it came with its own baggage. The choice of parameter and topology for the networks is often an optimization issue. Since metaheuristics are known to be good at solving these kind of problems, they are employed. We have surveyed many papers that employ metaheuristics to optimize the echo state network (ESN) which is among the firsts in reservoir computing. Moreover, we have highlighted some deductions from these papers and have pointed research gaps.

REFERENCES

[1] R. J. Schalkoff, *Artificial Neural Networks*, vol. 1. New York, NY, USA: McGraw-Hill, 1997.
[2] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, 1943.
[3] S. O. Haykin, *Neural Networks and Learning Machines*, vol. 3. Upper Saddle River, NJ, USA: Pearson, 2009.
[4] B. Yegnanarayana, *Artificial Neural Networks*. New Delhi, India: PHI Learning Pvt. Ltd., 2009.
[5] J. J. Hopfield, "Hopfield network," *Scholarpedia*, vol. 2, no. 5, p. 1977, 2007.
[6] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," in *Readings in Computer Vision*. Amsterdam, The Netherlands: Elsevier, 1987, pp. 522–533.
[7] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[8] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, 2009.
[9] K. Doya, "Bifurcations in the learning of recurrent neural networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 6, May 1992, pp. 2777–2780.
[10] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," Istituto Dalle Molle di Studi sull'Intell. Artif., Lugano, Switzerland, Tech. Rep. IDSIA-01-99, 1999.
[11] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks—With an Erratum note," German Nat. Res. Center, Inf. Technol. GMD, Bonn, Germany, Tech. Rep. 148, 2001, p. 13.
[12] 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, 2002.
[13] P. F. Dominey, "Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning," *Biol. Cybern.*, vol. 73, no. 3, pp. 265–274, Aug. 1995.
[14] 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.
[15] M. Lukoševičius, H. Jaeger, and B. Schrauwen, "Reservoir computing trends," *KI-Künstliche Intell.*, vol. 26, no. 4, pp. 365–371, Nov. 2012.
[16] B. Schrauwen, D. Verstraeten, and J. Van Campenhout, "An overview of reservoir computing: theory, applications and implementations," in *Proc. 15th Eur. Symp. Artif. Neural Netw.*, 2007, pp. 471–482.
[17] M. Lukoševičius, and H. Jaeger, "Overview of reservoir recipes," School Eng. Sci., Jacobs Univ., Bremen, Germany, Tech. Rep. 11, 2007.
[18] W. Maass, "Liquid state machines: Motivation, theory, and applications," in *Computability in Context: Computation and Logic in the Real World*. Singapore: World Scientific, 2011, pp. 275–296.
[19] Z. Yanduo and W. Kun, "The application of liquid state machines in robot path planning," *J. Comput.*, vol. 4, no. 11, pp. 1182–1186, 2009.
[20] A. A. Ferreira, T. B. Ludermir, and R. R. B. de Aquino, "An approach to reservoir computing design and training," *Expert Syst. Appl.*, vol. 40, no. 10, pp. 4172–4182, Aug. 2013.
[21] I. B. Yildiz, H. Jaeger, and S. J. Kiebel, "Re-visiting the echo state property," *Neural Netw.*, vol. 35, pp. 1–9, Nov. 2012.
[22] H. Jaeger, *Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the 'Echo State Network' Approach*, vol. 5. Bonn, Germany: GMD-Forschungszentrum Informationstechnik Bonn, 2002.
[23] M. Lukoševičius, "A practical guide to applying echo state networks," in *Neural Networks: Tricks of the Trade*. Berlin, Germany: Springer, 2012, pp. 659–686.
[24] A. A. Ferreira and T. B. Ludermir, "Comparing evolutionary methods for reservoir computing pre-training," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jul./Aug. 2011, pp. 283–290.
[25] A. T. Sergio and T. B. Ludermir, "Reservoir computing optimization with a hybrid method," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2014, pp. 2653–2660.
[26] A. B. Owen, "A robust hybrid of lasso and ridge regression," in *Contemporary Mathematics*, vol. 443, no. 7. Providence, RI, USA: AMS, 2007, pp. 59–72.
[27] S. Wang, X.-J. Yang, and C.-J. Wei, "Harnessing non-linearity by sigmoid-wavelet hybrid echo state networks (SWHESN)," in *Proc. IEEE 6th World Congr. Intell. Control Automat. (WCICA)*, vol. 1, Jun. 2006, pp. 3014–3018.
[28] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert, "Optimization and applications of echo state networks with leaky-integrator neurons," *Neural Netw.*, vol. 20, no. 3, pp. 335–352, Apr. 2007.
[29] B. Schrauwen, J. Defour, D. Verstraeten, and J. Van Campenhout, "The introduction of time-scales in reservoir computing, applied to isolated digits recognition," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2007, pp. 471–479.
[30] L. Büsing, B. Schrauwen, and R. Legenstein, "Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons," *Neural Comput.*, vol. 22, no. 5, pp. 1272–1311, 2010.
[31] S. M. Sait and H. Youssef, *Iterative Computer Algorithms With Applications in Engineering: Solving Combinatorial Optimization Problems*. Los Alamitos, CA, USA: IEEE Comput. Soc. Press, 1999.
[32] S. Luke, *Essentials of Metaheuristics*, 2nd ed. Raleigh, NC, USA: Lulu, 2013. [Online]. Available: http://cs.gmu.edu/∼sean/book/metaheuristics/
[33] P. J. van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated Annealing: Theory and Applications*. Dordrecht, The Netherlands: Springer, 1987, pp. 7–15.
[34] D. Whitley, "A genetic algorithm tutorial," *Statist. Comput.*, vol. 4, no. 2, pp. 65–85, Jun. 1994.
[35] K. Khan and A. Sahai, "A comparison of BA, GA, PSO, BP and LM for training feed forward neural networks in e-learning context," *Int. J. Intell. Syst. Appl.*, vol. 4, no. 7, pp. 23–29, 2012.
[36] A. T. Sergio and T. B. Ludermir, "PSO for reservoir computing optimization," in *Proc. Int. Conf. Artif. Neural Netw.* Berlin, Germany: Springer, 2012, pp. 685–692.
[37] A. A. Ferreira and T. B. Ludermir, "Genetic algorithm for reservoir computing optimization," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2009, pp. 811–815.
[38] I. Rechenberg, *Evolutionsstrategie—Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Berlin, Germany: Springer, 1973.
[39] N. Hansen, "The CMA evolution strategy: A comparing review," in *Towards a New Evolutionary Computation*. Berlin, Germany: Springer, 2006, pp. 75–102.
[40] T. van der Zant, V. Bečanović, K. Ishii, H.-U. Kobialka, and P. Plöger, "Finding good echo state networks to control an underwater robot using evolutionary computations," *IFAC Proc. Vols.*, vol. 37, no. 8, pp. 215–220, 2004.
[41] K. Ishu, T. van der Zant, V. Becanovic, and P. Ploger, "Identification of motion with echo state network," in *Proc. MTTS/IEEE TECHNO-OCEAN (OCEANS)*, vol. 3, Aug. 2004, pp. 1205–1210.
[42] A. Devert, N. Bredeche, and M. Schoenauer, "Unsupervised learning of echo state networks: A case study in artificial embryogeny," in *Proc. Int. Conf. Artif. Evol.* Berlin, Germany: Springer, 2007, pp. 278–290.
[43] K. O. Stanley and R. Miikkulainen, "Efficient reinforcement learning through evolving neural network topologies," in *Proc. 4th Annu. Conf. Genetic Evol. Comput.* San Mateo, CA, USA: Morgan Kaufmann, 2002, pp. 569–577.
[44] F. Jiang, H. Berry, and M. Schoenauer, "Supervised and evolutionary learning of echo state networks," in *Proc. Int. Conf. Parallel Problem Solving Nature*. Berlin, Germany: Springer, 2008, pp. 215–224.
[45] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, 2003.
[46] C. Hartland, N. Bredeche, and M. Sebag, "Memory-enhanced evolutionary robotics: The echo state network approach," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2009, pp. 2788–2795.
[47] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. IEEE 6th Int. Symp. Micro Mach. Hum. Sci. (MHS)*, Oct. 1995, pp. 39–43.
[48] S. Alam, G. Dobbie, Y. S. Koh, P. Riddle, and S. Ur Rehman, "Research on particle swarm optimization based clustering: a systematic review of literature and techniques," *Swarm Evol. Comput.*, vol. 17, pp. 1–13, Aug. 2014.
[49] H. Zhou, Y. Wang, and J. Huang, "Modeling of cortical signals using optimized echo state networks with leaky integrator neurons," in *Proc. Int. Conf. Neural Inf. Process.* Berlin, Germany: Springer, 2009, pp. 10–18.

[50] Q. Song and Z. Feng, "Stable trajectory generator-echo state network trained by particle swarm optimization," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Automat. (CIRA)*, Dec. 2009, pp. 21–26.

[51] Q. Song, Z. Feng, and Y. Wang, "Stable training method for echo state networks running in closed-loop based on particle swarm optimization algorithm," in *Proc. Int. Conf. Neural Inf. Process.* Berlin, Germany: Springer, 2009, pp. 253–262.

[52] Q. Song, Z. Feng, and M. Lei, "Stable training method for echo state networks with output feedbacks," in *Proc. IEEE Int. Conf. Netw., Sens. Control (ICNSC)*, Apr. 2010, pp. 159–164.

[53] Y. Xue, L. Yang, and S. Haykin, "Decoupled echo state networks with lateral inhibition," *Neural Netw.*, vol. 20, no. 3, pp. 365–376, Apr. 2007.

[54] D. Wierstra, F. J. Gomez, and J. Schmidhuber, "Modeling systems with internal state using evolino," in *Proc. 7th Annu. Conf. Genetic Evol. Comput.*, 2005, pp. 1795–1802.

[55] H. Zhou, Y. Wang, and K. Xing, "Modeling of McKibben pneumatic artificial muscles using optimized echo state networks," in *Proc. IEEE 8th World Congr. Intell. Control Automat. (WCICA)*, Jul. 2010, pp. 1723–1728.

[56] M. J. A. Rabin, M. S. Hossain, M. S. Ahsan, M. A. S. Mollah, and M. T. Rahman, "Sensitivity learning oriented nonmonotonic multi reservoir echo state network for short-term load forecasting," in *Proc. IEEE Int. Conf. Inform., Electron. Vis. (ICIEV)*, May 2013, pp. 1–6.

[57] E. Castillo, A. S. Hadi, A. Conejo, and A. Fernández-Canteli, "A general method for local sensitivity analysis with application to regression models and other optimization problems," *Technometrics*, vol. 46, no. 4, pp. 430–444, 2004.

[58] L. Yang, X. Liang, T. Ma, and K. Liu, "Spectrum prediction based on echo state network and its improved form," in *Proc. 5th Int. Conf. Intell. Hum.-Mach. Syst. Cybern. (IHMSC)*, vol. 1, Aug. 2013, pp. 172–176.

[59] Z. Wei-Min, L. Shao-Jun, and Q. Feng, "θ-PSO: A new strategy of particle swarm optimization," *J. Zhejiang Univ.-SCIENCE A*, vol. 9, no. 6, pp. 786–790, 2008.

[60] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[61] H. Wang and X. Yan, "Optimizing the echo state network with a binary particle swarm optimization algorithm," *Knowl.-Based Syst.*, vol. 86, pp. 182–193, Sep. 2015.

[62] S. Basterrech, E. Alba, and V. Snášel, "An experimental analysis of the echo state network initialization using the particle swarm optimization," in *Proc. 6th World Congr. Nature Biol. Inspired Comput. (NaBIC)*, Jul./Aug. 2014, pp. 214–219.

[63] N. Chouikhi, B. Ammar, N. Rokbani, A. M. Alimi, and A. Abraham, "A hybrid approach based on particle swarm optimization for echo state network initialization," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2015, pp. 2896–2901.

[64] N. Chouikhi, R. Fdhila, B. Ammar, N. Rokbani, and A. M. Alimi, "Single- and multi-objective particle swarm optimization of reservoir structure in echo state network," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 440–447.

[65] N. Chouikhi, B. Ammar, and A. M. Alimi. (2018). "Hierarchical bilevel multi-objective evolution of single- and multi-layer echo state network autoencoders for data representations." [Online]. Available: https://arxiv.org/abs/1806.01016

[66] Z. K. Malik, A. Hussain, and Q. J. Wu, "Multilayered echo state machine: A novel architecture and algorithm," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 946–959, Apr. 2017.

[67] M. Xu, M. Han, and X. Wang, "Hierarchical neural networks for multivariate time series prediction," in *Proc. IEEE 35th Chin. Control Conf. (CCC)*, Jul. 2016, pp. 6971–6976.

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

[69] K. Wu, Y. Zhu, Q. Li, and G. Han, "Algorithm and implementation of distributed ESN using Spark framework and parallel PSO," *Appl. Sci.*, vol. 7, no. 4, p. 353, 2017.

[70] C. Sheng, J. Zhao, and W. Wang, "Map-reduce framework-based non-iterative granular echo state network for prediction intervals construction," *Neurocomputing*, vol. 222, pp. 116–126, Jan. 2017.

[71] N. Chouikhi, B. Ammar, N. Rokbani, and A. M. Alimi, "PSO-based analysis of echo state network parameters for time series forecasting," *Appl. Soft Comput.*, vol. 55, pp. 211–225, Jun. 2017.

[72] L. Wei and L. Haitian, "Electrical load forecasting using echo state network and optimizing by PSO algorithm," in *Proc. IEEE 10th Int. Conf. Intell. Comput. Technol. Automat. (ICICTA)*, Oct. 2017, pp. 394–397.

[73] S. B. Salah, I. Fliss, and M. Tagina, "Echo state network and particle swarm optimization for prognostics of a complex system," in *Proc. IEEE/ACS 14th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Oct./Nov. 2017, pp. 1027–1034.

[74] H. Abdelbari and K. Shafi, "Learning structures of conceptual models from observed dynamics using evolutionary echo state networks," *J. Artif. Intell. Soft Comput. Res.*, vol. 8, no. 2, pp. 133–154, 2018.

[75] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[76] C. M. Anderson-Cook, *Practical Genetic Algorithms*. Oxfordshire, U.K.: Taylor & Francis, 2005.

[77] D. Xu, J. Lan, and J. C. Principe, "Direct adaptive control: An echo state network and genetic algorithm approach," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 3, Jul./Aug. 2005, pp. 1483–1486.

[78] A. A. Ferreira and T. B. Ludermir, "Using reservoir computing for forecasting time series: Brazilian case study," in *Proc. IEEE 8th Int. Conf. Hybrid Intell. Syst. (HIS)*, Sep. 2008, pp. 602–607.

[79] F. M. Bianchi, "Prediction of telephone calls load using echo state network with exogenous variables," *Neural Netw.*, vol. 71, pp. 204–213, Nov. 2015.

[80] F. M. Bianchi, E. De Santis, A. Rizzi, and A. Sadeghian, "Short-term electric load forecasting using echo state networks and PCA decomposition," *IEEE Access*, vol. 3, pp. 1931–1943, 2015.

[81] S. L. Ho, M. Xie, and T. N. Goh, "A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction," *Comput. Ind. Eng.*, vol. 42, nos. 2–4, pp. 371–375, 2002.

[82] A. Deihimi and H. Showkati, "Application of echo state networks in short-term electric load forecasting," *Energy*, vol. 39, no. 1, pp. 327–340, 2012.

[83] S. Løkse, F. M. Bianchi, and R. Jenssen, "Training echo state networks with regularization through dimensionality reduction," *Cognit. Comput.*, vol. 9, no. 3, pp. 364–378, 2017.

[84] S. Zhong, X. Xie, L. Lin, and F. Wang, "Genetic algorithm optimized double-reservoir echo state network for multi-regime time series prediction," *Neurocomputing*, vol. 238, pp. 191–204, May 2017.

[85] Q. Ma, L. Shen, and G. W. Cottrell. (2017). "Deep-ESN: A multiple projection-encoding hierarchical reservoir computing framework." [Online]. Available: https://arxiv.org/abs/1711.05255

[86] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[87] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

[88] Y. Zhang, Y. Yu, and D. Liu, "The application of modified ESN in chaotic time series prediction," in *Proc. IEEE 25th Chin. Control Decis. Conf. (CCDC)*, May 2013, pp. 2213–2218.

[89] M. Rigamonti *et al.*, "Echo state network for the remaining useful life prediction of a turbofan engine," in *Proc. Annu. Conf. Prognostics Health Manage. Soc.*, 2016, pp. 255–270.

[90] M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury, K. Goebel, and S. Poll, "Ensemble of optimized echo state networks for remaining useful life prediction," *Neurocomputing*, vol. 281, pp. 121–138, Mar. 2018.

[91] C. Yang, J. Qiao, and L. Wang, "A novel echo state network design method based on differential evolution algorithm," in *Proc. IEEE 36th Chin. Control Conf. (CCC)*, Jul. 2017, pp. 3977–3982.

[92] J. Qiao, F. Li, H. Han, and W. Li, "Growing echo-state network with multiple subreservoirs," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 391–404, Feb. 2017.

[93] L. Wang, H. Hu, X.-Y. Ai, and H. Liu, "Effective electricity energy consumption forecasting using echo state network improved by differential evolution algorithm," *Energy*, vol. 153, pp. 801–815, Jun. 2018.

[94] W.-H. Cui, J.-S. Wang, and S.-X. Li, "KPCA-ESN soft-sensor model of polymerization process optimized by biogeography-based optimization algorithm," *Math. Problems Eng.*, vol. 2015, Mar. 2015, Art. no. 493248.

[95] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[96] E. J. Amaya and A. J. Alvares, "Prognostic of RUL based on echo state network optimized by artificial bee colony," *Int. J. Prognostics Health Manage.*, vol. 7, no. 1, May 2016.

[97] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *J. Global Optim.*, vol. 39, no. 3, pp. 459–471, Apr. 2007.

[98] Y. Peng, H. Wang, J. Wang, D. Liu, and X. Peng, "A modified echo state network based remaining useful life estimation approach," in *Proc. IEEE Conf. Prognostics Health Manage. (PHM)*, Jun. 2012, pp. 1–7.

[99] H. Duan and X. Wang, "Echo state networks with orthogonal pigeon-inspired optimization for image restoration," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 11, pp. 2413–2425, Nov. 2016.

[100] H. Duan and P. Qiao, "Pigeon-inspired optimization: A new swarm intelligence optimizer for air robot path planning," *Int. J. Intell. Comput. Cybern.*, vol. 7, no. 1, pp. 24–37, Mar. 2014.

[101] Y. Han, Y. Jing, and K. Li, "Multi-step prediction for the network traffic based on echo state network optimized by quantum-behaved fruit fly optimization algorithm," in *Proc. IEEE 29th Chin. Control Decis. Conf. (CCDC)*, May 2017, pp. 2270–2274.

[102] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *J. Simul.*, vol. 76, no. 2, pp. 60–68, Feb. 2001.

[103] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Inf. Sci.*, vol. 222, pp. 175–184, Feb. 2012.

[104] A. Bala, I. Ismail, and R. Ibrahim, "Cuckoo search based optimization of echo state network for time series prediction," in *Proc. 7th Int. Conf. Intell. Adv. Syst. (ICIAS)*, 2018, pp. 1–6.

[105] S. Walton, O. Hassan, K. Morgan, and M. Brown, "Modified cuckoo search: A new gradient free optimisation algorithm," *Chaos, Solitons Fractals*, vol. 44, no. 9, pp. 710–718, 2011.

[106] S. Morando, M. C. Pera, N. Y. Steiner, S. Jemei, D. Hissel, and L. Larger, "Reservoir computing optimisation for PEM fuel cell fault diagnostic," in *Proc. IEEE Vehicle Power Propuls. Conf. (VPPC)*, Dec. 2017, pp. 1–7.

[107] O. K. Erol and I. Eksin, "A new optimization method: Big Bang–Big Crunch," *Adv. Eng. Softw.*, vol. 37, no. 2, pp. 106–111, 2006.

[108] A. Deihimi and A. Rahmani, "Application of echo state networks for estimating voltage harmonic waveforms in power systems considering a photovoltaic system," *IET Renew. Power Gener.*, vol. 11, no. 13, pp. 1688–1694, 2017.

[109] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[110] L. Pan, J. Cheng, H. Li, Y. Zhang, and X. Chen, "An improved echo state network based on variational mode decomposition and bat optimization for internet traffic forecasting," in *Proc. IEEE 17th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2017, pp. 417–421.

[111] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO)*. Springer, 2010, pp. 65–74.

[112] C. Blum and A. Roli, "Hybrid metaheuristics: An introduction," in *Hybrid Metaheuristics*. Berlin, Germany: Springer, 2008, pp. 1–30.

[113] X. Xu, D. Niu, M. Fu, H. Xia, and H. Wu, "A multi time scale wind power forecasting model of a chaotic echo state network based on a hybrid algorithm of particle swarm optimization and tabu search," *Energies*, vol. 8, no. 11, pp. 12388–12408, 2015.

[114] P. Dürr, C. Mattiussi, and D. Floreano, "Neuroevolution with analog genetic encoding," in *Parallel Problem Solving From Nature*. Berlin, Germany: Springer, 2006, pp. 671–680.

[115] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no 6, pp. 1362–1381, Dec. 2009.

[116] S. T. Hsieh, T. Y. Sun, C. C. Liu, and S. J. Tsai, "Efficient population utilization strategy for particle swarm optimizer," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 444–456, Apr. 2009.

[117] P.-F. Pai and W.-C. Hong, "Forecasting regional electricity load based on recurrent support vector machines with genetic algorithms," *Electr. Power Syst. Res.*, vol. 74, no. 3, pp. 417–425, 2005.

[118] Q. Zhang and T. Liu, "Research on mid-long term load forecasting base on wavelet neural network," in *Proc. IEEE 2nd Int. Conf. Comput. Eng. Appl. (ICCEA)*, vol. 2, Mar. 2010, pp. 217–220.

[119] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, 2004.

[120] S. Hou and Y. Li, "Short-term fault prediction based on support vector machines with parameter optimization by evolution strategy," *Expert Syst. Appl.*, vol. 36, no. 10, pp. 12383–12391, 2009.

[121] H. Dhahri, A. M. Alimi, and A. Abraham, "Designing beta basis function neural network for optimization using artificial bee colony (ABC)," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2012, pp. 1–7.

[122] H. Dhahri and A. M. Alimi, "The modified differential evolution and the RBF (MDE-RBF) neural network for time series prediction," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2006, pp. 2938–2943.

[123] S. Arora and S. Singh, "A conceptual comparison of firefly algorithm, bat algorithm and cuckoo search," in *Proc. IEEE Int. Conf. Control Comput. Commun. Mater. (ICCCCM)*, Aug. 2013, pp. 1–4.

[124] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *Proc. Int. Fuzzy Syst. Assoc. World Congr.* Berlin, Germany: Springer, 2007, pp. 789–798.

[125] W. Zalewski, F. Silva, A. G. Maletzke, and C. A. Ferrero, "Exploring shapelet transformation for time series classification in decision trees," *Knowl.-Based Syst.*, vol. 112, pp. 80–91, Nov. 2016.

[126] G.-B. Huang, "An insight into extreme learning machines: Random neurons, random features and kernels," *Cognit. Comput.*, vol. 6, no. 3, pp. 376–390, 2014.

[127] Z. Shi and M. Han, "Support vector echo-state machine for chaotic time-series prediction," *IEEE Trans. Neural Netw.*, vol. 18, no. 2, pp. 359–372, Mar. 2007.

[128] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[129] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Proc. Workshops 29th AAAI Conf. Artif. Intell.*, vol. 1, 2015, pp. 40–46.

[130] N. Amjady and F. Keynia, "Short-term load forecasting of power systems by combination of wavelet transform and neuro-evolutionary algorithm," *Energy*, vol. 34, no. 1, pp. 46–57, 2009.

[131] Y. Wang, J. Wang, G. Zhao, and Y. Dong, "Application of residual modification approach in seasonal ARIMA for electricity demand forecasting: A case study of China," *Energy Policy*, vol. 48, pp. 284–294, Sep. 2012.

[132] P. S. Kalekar, "Time series forecasting using Holt–Winters exponential smoothing," Kanwal Rekhi School Inf. Technol., Mumbai, India, Tech. Rep. 4329008, Dec. 2004, pp. 1–13.

[133] C.-M. Huang, C.-J. Huang, and M.-L. Wang, "A particle swarm optimization to identifying the ARMAX model for short-term load forecasting," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 1126–1133, May 2005.

[134] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural Comput.*, vol. 12, no. 5, pp. 1207–1245, 2000.

[135] C. Gallicchio and A. Micheli, "Architectural and Markovian factors of echo state networks," *Neural Netw.*, vol. 24, no. 5, pp. 440–456, 2011.

[136] J. B. Butcher, D. Verstraeten, B. Schrauwen, C. R. Day, and P. W. Haycock, "Reservoir computing and extreme learning machines for non-linear time-series data analysis," *Neural Netw.*, vol. 38, pp. 76–89, Feb. 2013.

[137] S. Otte, M. V. Butz, D. Koryakin, F. Becker, M. Liwicki, and A. Zell, "Optimizing recurrent reservoirs with neuro-evolution," *Neurocomputing*, vol. 192, pp. 128–138, Jun. 2016.

[138] C.-J. Chang, J.-Y. Lin, and M.-J. Chang, "Extended modeling procedure based on the projected sample for forecasting short-term electricity consumption," *Adv. Eng. Inform.*, vol. 30, no. 2, pp. 211–217, 2016.

[139] R. N. Neelamani, H. Choi, and R. Baraniuk, "ForWaRD: Fourier-wavelet regularized deconvolution for ill-conditioned systems," *IEEE Trans. Signal Process.*, vol. 52, no. 2, pp. 418–433, Feb. 2004.

[140] J. Zhang, D. Zhao, and W. Gao, "Group-based sparse representation for image restoration," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3336–3351, Aug. 2014.

[141] V. Katkovnik, A. Foi, K. Egiazarian, and J. Astola, "Directional varying scale approximations for anisotropic signal processing," in *Proc. IEEE 12th Eur. Signal Process. Conf.*, Sep. 2004, pp. 101–104.

[142] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, no. 3, pp. 248–272, Aug. 2008.

[143] T. Joyce and J. M. Herrmann, "A review of no free lunch theorems, and their implications for metaheuristic optimisation," in *Nature-Inspired Algorithms Appl. Optim.* Springer, 2018, pp. 27–51.

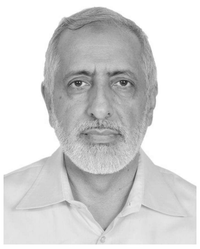
[144] G.-G. Wang, S. Deb, and L. dos Santos Coelho, "Earthworm optimisation algorithm: A bio-inspired metaheuristic algorithm for global optimisation problems," *Int. J. Bio-Inspired Comput.*, vol. 12, no. 1, pp. 1–22, 2018.

[145] N. A. Kallioras, N. D. Lagaros, and D. N. Avtzis, "Pity beetle algorithm—A new metaheuristic inspired by the behavior of bark beetles," *Adv. Eng. Softw.*, vol. 121, pp. 147–166, Jul. 2018.

ABUBAKAR BALA (M'18) received the bachelor's degree in computer engineering from Bayero University Kano (BUK), Nigeria, in 2011, and the master's degree in computer engineering from the King Fahd University of Petroleum and Minerals, Saudi Arabia, in 2015. He is currently pursuing the Ph.D. degree in electrical engineering with Universiti Teknolgi PETRONAS, Malaysia. He is currently a Lecturer with BUK. He is working on developing new algorithms to optimize the echo state network, a type of recurrent neural network. His research interests include optimization, artificial neural networks, fault predictions, cloud computing, and renewable energy.

IDRIS ISMAIL received the bachelor's degree in electrical engineering from Wichita State University, KS, USA, in 1986, the master's degree in control system engineering from The University of Sheffield, U.K., in 2000, and the Ph.D. in electrical and electronics engineering from The University of Manchester, U.K., in 2009. He is currently an Associate Professor with Universiti Teknologi PETRONAS, Malaysia, and also a registered Professional Engineer with the Board of Engineers, Malaysia. His current research interests include data analytics and model predictive control.

ROSDIAZLI IBRAHIM (M'09) received the B.Eng. degree in electrical engineering from Universiti Putra Malaysia, Kembangan, Malaysia, in 1996, the M.Sc. degree in automation and control from Newcastle University, Newcastle upon Tyne, U.K., in 2000, and the Ph.D. degree in electrical and electronic engineering from the University of Glasgow, Glasgow, U.K., in 2008. He is currently an Associate Professor and a Chartered Engineer registered with the U.K. Engineering Council. He is also the Dean of the Center of Postgraduate Studies with Universiti Teknologi PETRONAS. His current research interests include intelligent control and nonlinear multivariable process modeling for process control application.



SADIQ M. SAIT (SM'02) was born in Bengaluru. received the bachelor's degree in electronics engineering from Bangalore University in 1981 and the master's and Ph.D. degrees in electrical engineering from KFUPM in 1983 and 1987, respectively. He is currently a Professor of computer engineering and the Director of the Center for Communications and IT Research, KFUPM. He is also the principle author of two books. He has authored over 200 research papers, contributed chapters to technical books, and lectured in over 25 countries. In 1981, he received the Best Electronic Engineer Award from the Indian Institute of Electrical Engineers, Bengaluru.