Neurocomputing 247 (2017) 39–58

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Evolving multi-dimensional wavelet neural networks for classification using Cartesian Genetic Programming

Maryam Mahsal Khan[a,*], Alexandre Mendes[a], Ping Zhang[b], Stephan K. Chalup[a]

[a] School of Electrical Engineering & Computer Science, Callaghan Campus, The University of Newcastle, NSW 2308, Australia
[b] Menzies Health Institute Queensland, Gold Coast Campus, Griffith University, QLD 4222, Australia

ARTICLE INFO

Article history:
Received 22 July 2016
Revised 5 January 2017
Accepted 11 March 2017
Available online 27 March 2017

Communicated by Prof. H.R. Karimi

Keywords:
Neuroevolution
Wavelet Neural Networks
Classification
Cartesian Genetic Programming

ABSTRACT

Wavelet Neural Networks (WNNs) are complex artificial neural systems and their training can be a challenge. In the past, most common training schemes for WNNs, such as gradient descent, have been restricted to training only a subset of differentiable parameters. In this paper, we propose an evolutionary method to train both differentiable and non-differentiable parameters using the concept of Cartesian Genetic Programming (CGP). The approach was evaluated on the two-spiral task and on real-world datasets for the detection of breast cancer and Parkinson's disease. In our experiments, the performance of the proposed method was comparable to several standard methods of classification. On the breast cancer dataset, the performance was better than other non-ensemble and multistep processing methods. The experimental results show how the performance of WNNs depends on the number of wavelons used. The presented case studies demonstrate that the proposed WNNs perform competitively in comparison to several other methods and results reported in literature.

1. Introduction

The wavelet transform has been used in pattern recognition, signal processing and compression applications for its ability to extract information from signals at either high time or frequency resolutions [1–3]. Wavelet neural networks (WNNs) utilize the concept of the wavelet transform in neural networks.

A combined model of wavelets and neural networks is suitable for function approximation and can be used for prediction and classification. WNNs have been successfully applied in many areas, including signal denoising [4], signal classification and compression [5], short-term electricity load forecasting [6], speech segmentation [7] and speaker recognition [8]. WNNs can provide better function approximation ability than standard multilayer perceptrons (MLPs) and radial basis function (RBF) neural networks over a wide range of applications [9,10].

A WNN is determined by five key parameters. Three of them refer to the activation function (scale, translation, rotation) and two of them to the architecture of the network (weights and number of neurons). More details on the behavior of these parameters is discussed in Section 2.1.4. The standard training procedure of a WNN employs a gradient descent algorithm that can suffer from slow convergence and local optima [11]. In order to optimize the performance of WNNs, a number of research studies have utilized evolutionary algorithms and evolutionary programming techniques [12,13].

Prediction of air and ground traffic flow [14,15], energy consumption [16], large scale function estimation [17], function approximation [18–20], power transformer monitoring [21] and centrifugal compression [22] are a few of the many applications of WNNs which utilize genetic algorithms (GA). WNN evolution via differential evolution (DE) has also been quite successful and includes applications such as load forecasting [23] and bankruptcy prediction [24]. This variety of applications illustrates the adaptability of WNNs to different data domains.

The most common strategy to optimize combinations of WNN parameters is to evolve only the activation function parameters [14–17]. Awad [25] evolved the translation and scale parameters using a GA and trained the weights using the Levenberg Marquardt algorithm. Jinru et al. [26] use a two-stage approach, where a GA is used first for a global search of the parameters, and in the second stage, the optimized parameters are further fine-tuned by using local search algorithms like gradient descent. In [27], the translation parameter is adaptive to the network input and its response to a non-linear function while the remaining attributes are evolved and optimized using particle swarm optimization. Simultaneous evolution of activation function parameters and network structure have

* Corresponding author.
E-mail addresses: maryammahsal.khan@uon.edu.au, c3196821@uon.edu.au (M.M. Khan), alexandre.mendes@newcastle.edu.au (A. Mendes), p.zhang@griffith.edu.au (P. Zhang), stephan.chalup@newcastle.edu.au (S.K. Chalup).

http://dx.doi.org/10.1016/j.neucom.2017.03.048
0925-2312/

also been studied and applied in various domains such as function approximation, Parkinson's disease detection and prediction of hydro-turbine machine condition [20,28,29].

Apart from the existing methods of training wavelet parameters, there are a number of optimization algorithms, including self-adaptive differential evolution [30] and a social emotional algorithm which uses local search function [31] that can be used to optimize the WNN parameters.

In the present paper, a novel algorithm based on the concept of Cartesian Genetic Programming (CGP) is used to evolve a multi-dimensional wavelet neural network, so that its potential application to classification tasks can be evaluated. The paper also aims to contribute to a better understanding of the behaviour of WNNs when their parameters are adjusted.

CGP is an evolutionary programming technique developed by Miller et al. [32]. The concept of CGP has also been used to evolve artificial neural networks [33]. The motivation behind using CGP for evolving parameters is, firstly, that CGP doesn't bloat [34] because the network becomes dominated by redundant genes that have a neutral effect [35–37] on the performance. Secondly, most of the applications evolved via CGP are generic, robust and present good accuracy compared to other methods [38–41].

The computational cost of a WNN increases with the input dimensions of the system. Our objective is to introduce an algorithm that would have the ability to switch features on and off, hence making them either active or inactive during the evolution process. Discarding too many features might result in reduced accuracy. The advantage of using an evolution-based concept to evolve parameters is that features can be pruned during evolution while balancing the need for accuracy, thus efficiently reducing the time to train a network.

Another contribution of this work is the introduction of a rotation parameter R_i, represented as an $n \times n$ matrix where n is the total number of input features. Rotation matrices have not been used in any similar applications yet, due to non-differentiability issues and high computational cost. Our intent is to exploit rotations so that the approximation capability of WNNs can be correctly assessed.

In two of our previous publications [29,42] we have used CGP to evolve wavelet parameters for a one-dimensional WNN. The present manuscript is about a separate study on the concept of multi-dimensional WNNs and the introduction of the rotation parameter for approximating functions. The structure of this paper is as follows. Section 2 describes WNNs, their properties and the tuning parameters, with visual examples. This section also introduces the mechanism used for building wavelet networks via Cartesian Genetic Programming (CGPWNN), constituting the main technical contribution of the paper. Sections 3–5 present the application of WNNs to three test problems: the standard 2D spiral benchmark, breast cancer classification via mammographic images, and Parkinson's disease detection via speech signal analysis. Section 6 incorporates conclusions and possible directions for future research.

2. Background

2.1. Wavelet neural networks

WNNs represent a class of neural networks with wavelets as activation functions; i.e. they combine the theory of wavelet transforms and neural networks [43]. WNNs generally have a feed-forward structure, with one hidden layer, as shown in Fig. 1, and activation functions are drawn from an orthonormal wavelet family. The most common wavelet activation functions are Gaussian, Mexican hat, Morelet and Haar wavelets [44].

Three parameters play a significant role in the tuning of wavelets for function approximation in the context of WNNs. They

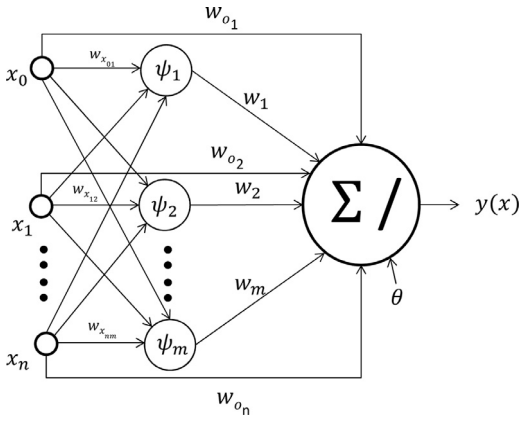


Fig. 1. Structure of a Wavelet Neural Network. The network has m-wavelons in the hidden layer and one-neuron in the output layer. Notice the n direct weighted connections from the inputs to the output neuron.

are dilation α, translation β and rotation R. For a general multi-dimensional WNN, its output can be described mathematically as

$$y(x) = \sum_{i=1}^{m} w_i \psi_i [\alpha_i R_i (w_{x_i} x - \beta_i)] + \sum_{j=1}^{n} w_{o_j} x_j + \theta \quad (1)$$

where w_i represents the weight of each wavelon structure, α_i, β_i and R_i are the dilation, translation and rotation matrices respectively, w_{x_i} is the weight matrix of the collective inputs x connected to the wavelons, w_{o_j} is the weight of the inputs connected to the output neuron and bias θ is used for nonzero mean functions on finite domains [44].

2.1.1. Properties of WNNs

WNNs have been used in several ways and have many variants. Fig. 2 shows a basic decomposition of a WNN and the different combinations that can be found in the literature.

- Feedforward WNNs can only have one hidden layer, as opposed to many in a general ANN system. The major advantage of such a property is that hidden unit dynamics can be understood as percentage contributions of each neuron towards prediction. This property contributes to the network's simplicity and a relatively fast convergence speed.
- Wavelets have an intrinsic ability to analyze input patterns at different resolutions. For one-dimensional data, this corresponds to changing the width of the wavelet function. In any n-dimensional problem, patterns can be enveloped with wavelets of different widths, thus analyzing data at various levels of resolution.
- Zhang [43] argued and proved that wavelet networks preserve the *universal approximation* property that characterizes NNs. The behaviors of network weights and wavelet coefficients are linked and have a good approximation quality even when small networks are used. Moreover, WNNs have been found to achieve the same quality of approximation to that of a neural network, but with a reduced network size [43].
- The basic approach of constructing a WNN is to process the wavelets and the neural network separately. The inputs are decomposed by applying the wavelet transform based on *'wavelet bases'* which are dyadic dilations and translations of the mother wavelet [43,45,46]. The wavelet coefficients are then forwarded to the neural network for training. This approach is called *'wavenet'*. The second approach refers to the wavelet theory and neural network combined into a single process [43] called the *'wavelet network'*. In this approach, dilations, translations and weights are optimized. As the scale and dilation can assume

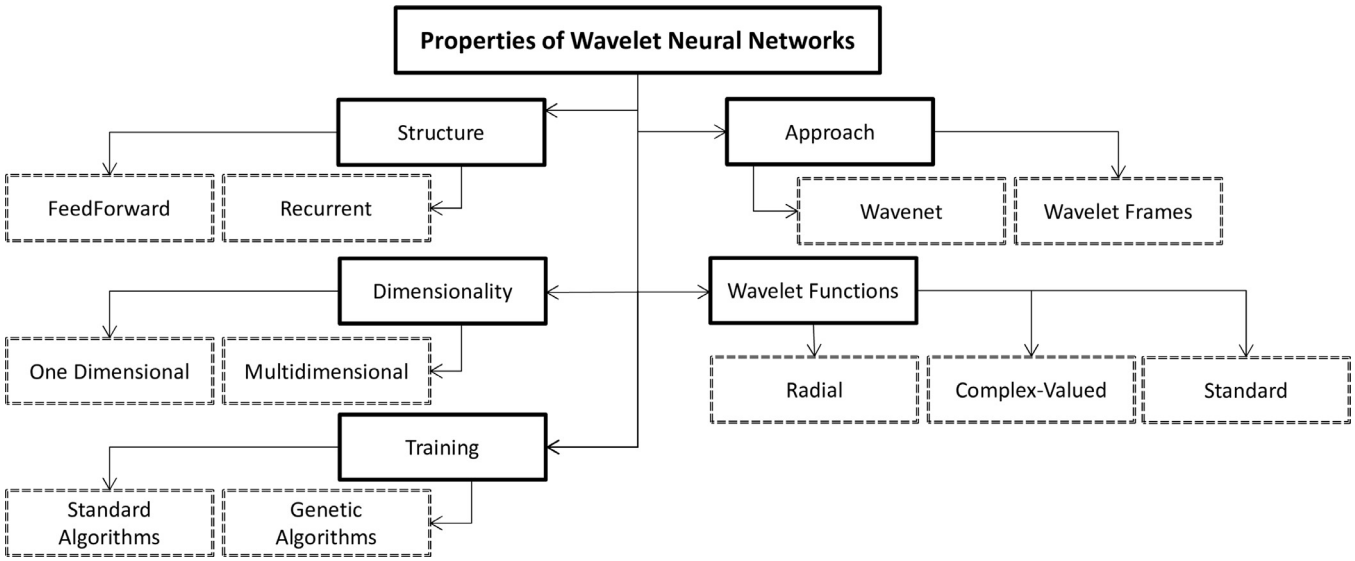


Fig. 2. Properties of a Wavelet Neural Network.

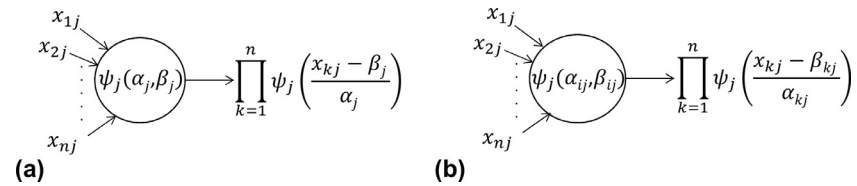


Fig. 3. Different types of a wavelon. (a) One-dimensional wavelon (b) Multi-dimensional wavelon.

any value, the wavelets produced are non-orthogonal and referred to as a *'wavelet frame'* [47]. The main drawback of WNNs is that their performance is affected by high input dimensions, as the construction of the wavelet bases becomes computationally expensive. However, this problem can be circumvented by using the wavelet frame strategy.

- Fig. 3 shows two examples of one- and multi-dimensional wavelons with n-input features. A one-dimensional WNN consists of wavelons with exactly one translate β_j and one dilate α_j parameter. That is, all inputs to the wavelon go through the same transformation. The wavelons produce non-zero outputs when the input lies within a small area of the input domain. In case of a multi-dimensional WNN, each wavelon has n translate β_{ij} and n dilate values α_{ij} and hence each input is transformed differently. A non-zero output is produced when the input vector lies within a small area of the multi-dimensional input domain.
- A WNN can be constructed through wavelet functions that are either standard, radially symmetric or complex-valued. Many applications [4–8] utilize one of these three types of functions for network construction. Examples of a standard and a radially symmetric Morelet wavelet are shown in Fig. 4.
- A standard WNN typically has a feedforward architecture, but in order to approximate dynamical systems, recurrent architectures can also be exploited. Some work about feasible recurrent architectures and their applicability in system identification can be found in [48–51].

2.1.2. Learning in WNNs

A traditional WNN requires the optimization of its weights, and of the translate and dilate parameters. As these parameters are differentiable, standard gradient methods can be used to train the network. In the literature, these parameters are commonly optimized via stochastic gradient [43], conjugate gradient [7], residual base selection [52], orthogonalized residual base selection (ORBS) [53], genetic algorithms [54,55], evolutionary algorithms [28] or evolutionary programming [42]. For an n-dimensional problem, where $n > 1$, a rotation matrix can further control and modify the spatial response of the wavelons but in high dimensions, differentiation becomes difficult and computationally expensive [56]. Several methods have been proposed [43,57] to train the rotation matrix but they are not computationally efficient and hence the rotation parameter of the WNN is often discarded and not used in the training of n-dimensional problem tasks.

2.1.3. Initialization of the WNN

The number of wavelons and their initialization in a WNN is a critical issue for training wavelet networks [56]. As wavelets are rapidly vanishing functions, a dilation value that is too small will make the approximation too local, and similarly, an inappropriate initialization of translation will result in the wavelet being out of the input domain. Hence, based on the distribution of the input domain, appropriate dilation parameters, α_{ij}, and translation parameters, β_{ij}, should be initially assigned. The number of wavelets is highly dependent on the task at hand. Zhang et al. [43] initialized coefficients via the orthogonal least-squares method, while Oussar [58] proposed both a heuristic initialization procedure (shown in Eq. (2)) and a selection method where a library of wavelets was generated and ranked, where P_i and Q_i were the minimum and maximum values of the input x_i from the feature set. In this manner, the wavelet covers the entire span of the feature space:

$$\alpha_{ij} = 0.5(P_i + Q_i)$$
$$\beta_{ij} = 0.2(Q_i - P_i) \quad (2)$$

Echuaz et al. used trigonometric wavelets [59] and a clustering method [60] to initially position the wavelet where the distribution of the input features represented the dilation of the wavelet.

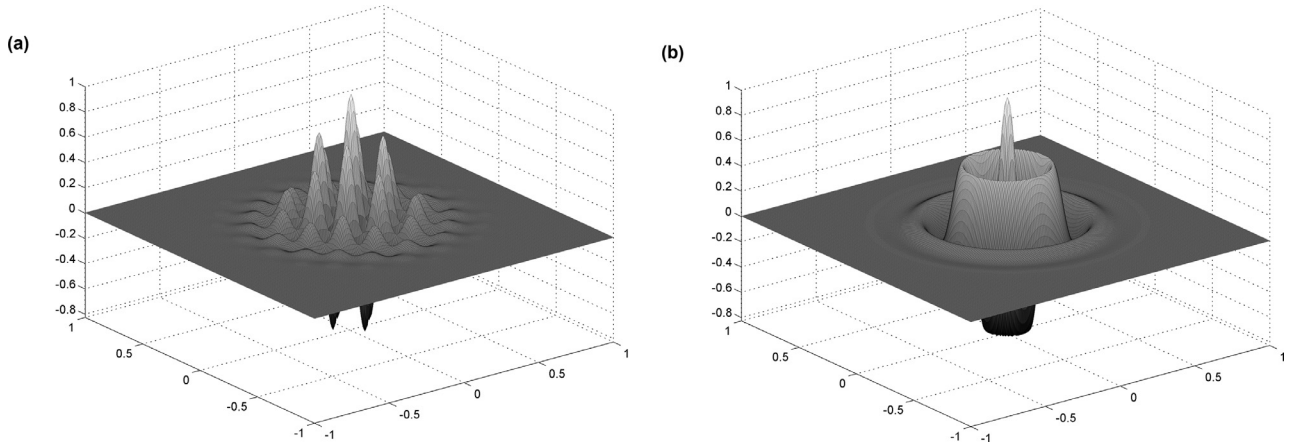


Fig. 4. Different responses of a Morelet wavelet function. (a) A standard 2D Morelet activation function. (b) Radially symmetric 2D Morelet function; where $(x, y) \in [-1, 1]$. Notice symmetrical responses around the origin (0, 0) in (b).

Zainuddin et al. [61] used a hybridized approach of initializing the translation vectors, where a type-2 fuzzy c-means clustering algorithm is combined with a harmony search-based meta-heuristic algorithm. Ling et al. [62] proposed an algorithm called the variable translation based WNN, where a non-linear function is used to assign different translation values based on the sum of inputs of a wavelon.

For the initialization of wavelons, Lin [63] proposed an on-line partition method based on a scattered partitioning of the input data to initialize the number of wavelons. Rao et al. [48] used correlation architecture to train wavelet networks by adding new wavelons at each step, while Yi et al. [64] initialized a large number of wavelons and then reduced it by a shrinkage method. Similarly, in [18,25] wavelons were added one at a time when the estimated approximation error was higher than a predefined threshold.

2.1.4. Understanding WNNs

There are four main parameters of a WNN wavelon: dilation α, translation β, rotation R and the activation function ψ. In this section, we will focus on changing one parameter at a time in order to better understand the response of the wavelon. Along with the wavelon functionality, the role of the activation function in the output layer will also be explored. This procedure, along with a visual representation, will facilitate understanding how WNNs work.

- Dilation (Scale): The dilation parameter α affects the width of the wavelet functions, i.e. increasing α increases the spread of the function, where $\alpha \in (0, 1)$. The response of a WNN with $\beta = 0$ and $\alpha \in \{0.1, 0.2, 0.5\}$ for Gaussian and Mexican hat wavelet functions is shown in Fig. 5. We can see the increase in the width of the function as α grows. Thus, it would be intuitive to assume that higher α values are more desirable in situations where the variance of feature values is high.
- Translation (Shift): The translation parameter β shifts the output of the wavelon by an amount β. This property of wavelons helps to enclose features. Consider a two-input WNN with one wavelon having a Gaussian wavelet function. The two-inputs (x, y) represent the Cartesian coordinates in the 2D Cartesian space where $(x, y) \in [-1, 1]$. Fig. 6 shows the responses of the WNN with translations $\beta \in \{-0.8, 0.2, 0.5\}$. The value of $\alpha = 0.1$ was fixed for all cases. We can see that the translation parameter simply shifted the response of the activation functions by the amount β.
- Rotation R: The parameter R changes the rotation of the input relative to the coordinate axes. Most of the research surrounding WNNs does not incorporate the rotation aspect of the wavelet network. This is because training via gradient algorithms cannot be applied due to differentiability issues, and alternative approaches of training rotation are computationally expensive [56].
  Fig. 7 shows the response of a two-input, one-output WNN with one wavelon. The two inputs represent (x, y) in the Cartesian space. The output neuron has a linear activation function. Fig. 7(a)–(c) show the WNN responses with $(\alpha, \beta, R) = (0.2, 0, \{0, \pi/4, \pi/8\})$, respectively.
  It has to be kept in mind that rotation can only have effect with wavelet functions that are not symmetric. Thus, for a 2D-Gaussian function with the same variance along x- and y-directions, rotation by any factor does not change the WNN response.
- Wavelet Functions ψ: The choice of activation function has a large impact on the performance of the network. The most commonly used activation functions are Morelet, Gaussian and Mexican hat functions. These wavelet functions act as band-pass filters, thus allowing input features within a certain frequency range to pass. Figs. 8(a)–(c) show a 1D representation along with their frequency responses (d)–(f) for the three wavelets aforementioned. Fig. 8(g)–(i) show the 3D surface view of the wavelets.
  In literature, there are two main types of WNNs: one-dimensional and multi-dimensional. In a one-dimensional WNN, each wavelon has only one dilation α and translation β value. In this case, a system with n features has each of them scaled and translated by the same amount. On the other hand, in a multi-dimensional WNN, for a system with n features, each wavelon has n translation and n dilation values. Therefore, each feature is transformed by a different filter and the resulting effect is then calculated.
  In order to visualize the one- and multi-dimensional concepts in the 2D space, first we use an example of a one-dimensional WNN with 2 inputs representing the x- and y-coordinates and a single wavelon with a Morelet activation function. The x- and y-coordinates represent a 2D Cartesian plane within the interval $[-1, 1]$. The value of (α, β) is (0.1, 0). Fig. 9(a) shows the flat view of the output of the one-dimensional WNN.
  Fig. 9(b) represents the same example, but using a multi-dimensional WNN where each x- and y-coordinate have their independent (α, β) values. In the illustrations, β is constant for both inputs and only α is different, with a value of 0.1 and 0.3 for the x- and y-coordinates, respectively. We can see how the shape of the WNN wavelons response has changed. With those two examples, one can conclude that multidimensionality

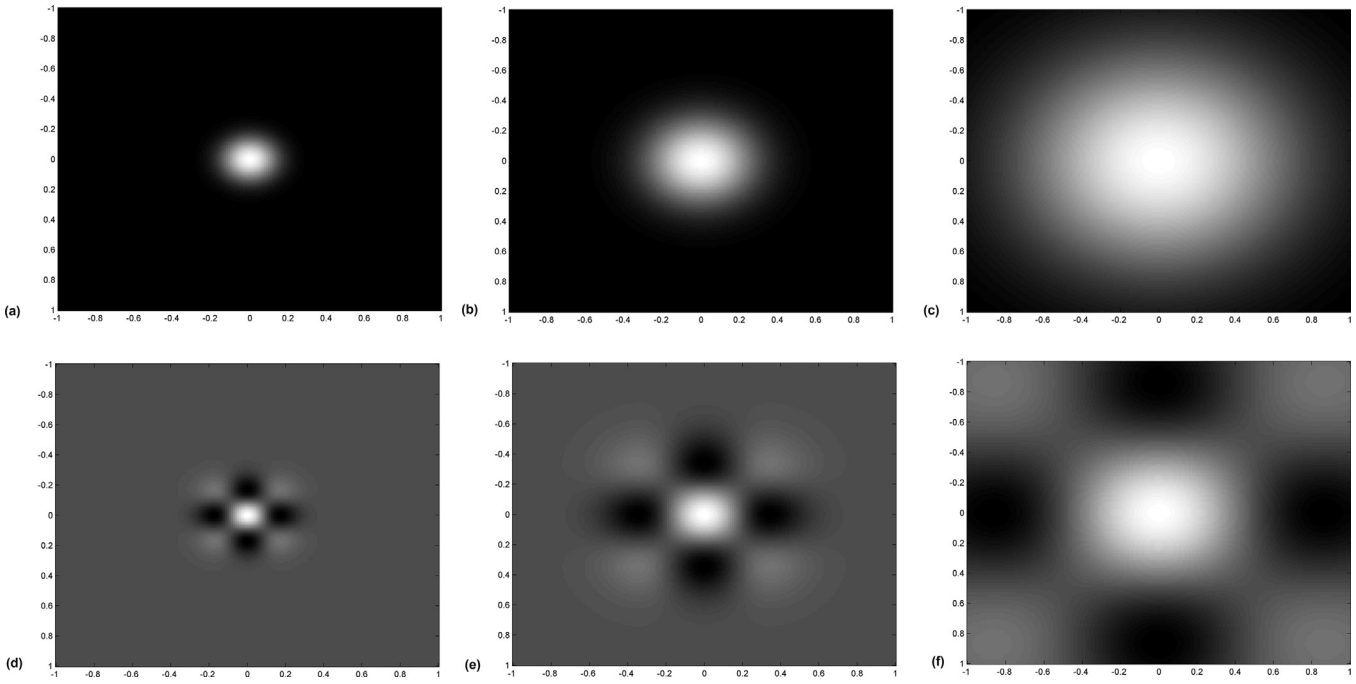


Fig. 5. Effect of changing the dilation parameter α, while $\beta = 0$, on a wavelet function with input $(x, y) \in [-1, 1]$: (a)–(c) show responses for a Gaussian wavelet, (d)–(f) are responses of a Mexican hat wavelet. The outputs for parameters $\alpha = 0.1,\ 0.2,\ 0.5$ are shown in columns 1–3, respectively.

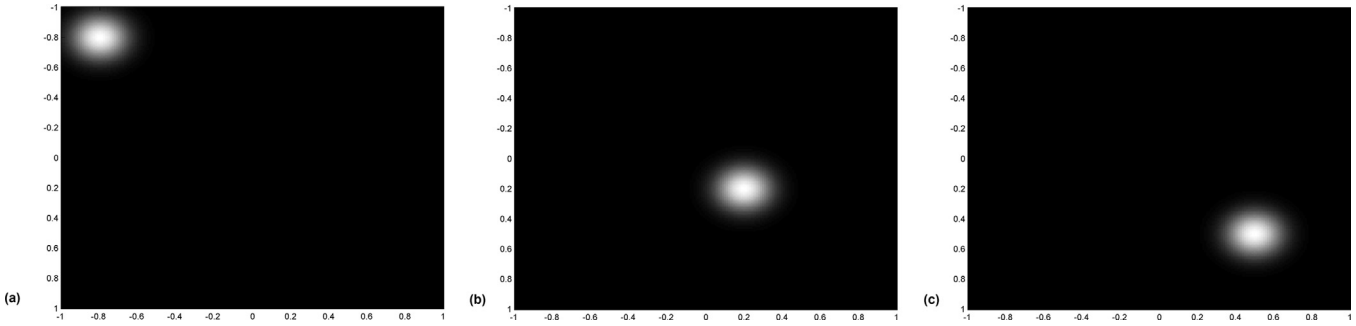


Fig. 6. Effect of changing the translation parameter β, while $\alpha = 0.1$, on a wavelet function with input $(x, y) \in [-1, 1]$: where β = (a) -0.8, (b) 0.2 and (c) 0.5, respectively. Notice the shift of the response center is equal to β simultaneously along the x- and y-axes.

has the strength of generating irregularly shaped responses and the ability to confine features within minimum boundary conditions.

A standard (or conventional) WNN uses standard activation functions (mentioned before). However, there also exists a class of WNN which uses radial functions, called the Radial Wavelet Neural Network (RWNN). In a RWNN, the wavelet function equation is modified to incorporate Euclidean distances. Eq. (3) is an example of a radially symmetric Mexican hat function.

$$\psi(x) = \frac{(1 - \|x - \beta_k\|^2)}{\alpha^2} exp\left(\frac{\|x - \beta_k\|^2}{2\alpha^2}\right) \tag{3}$$

Several research studies focus on using radially symmetric wavelets for approximation and classification problems, e.g., license plate character recognition and classification of heart diseases, among others [65,66]. The radial responses for Gaussian, Morelet and Mexican hat functions are displayed in Fig. 10.

- Output Layer Activation Function O_f: The output layer contains neurons with either linear or non-linear activation functions. The most common activation functions are log-sigmoid and hyperbolic tangent. The output neuron sums up the responses of the wavelons and the final output of the network is significantly changed by the choice of activation function. All examples above show responses of WNNs with a linear activation function in the output neuron. Thus, responses can be manipulated and the accuracy of the system can be controlled by using different activation functions.

It is worth mentioning that all the examples above represent a single wavelon in a WNN. In the case of multiple wavelons, the contribution of each wavelon is weighted by the wavelon's coefficient. In this manner, features are filtered and weighted to produce good approximations or classifications.

2.2. Proposed algorithm

2.2.1. Genome specification

In our study the genome structure of a multi-dimensional Wavelet Neural Network consists of three main sections: (1) a hidden layer with m wavelons, (2) an output layer with n outputs and (3) a bias.

Hidden Layer with wav_m Wavelons: The main and computationally expensive entity of WNNs are the wavelons wav_m where m is

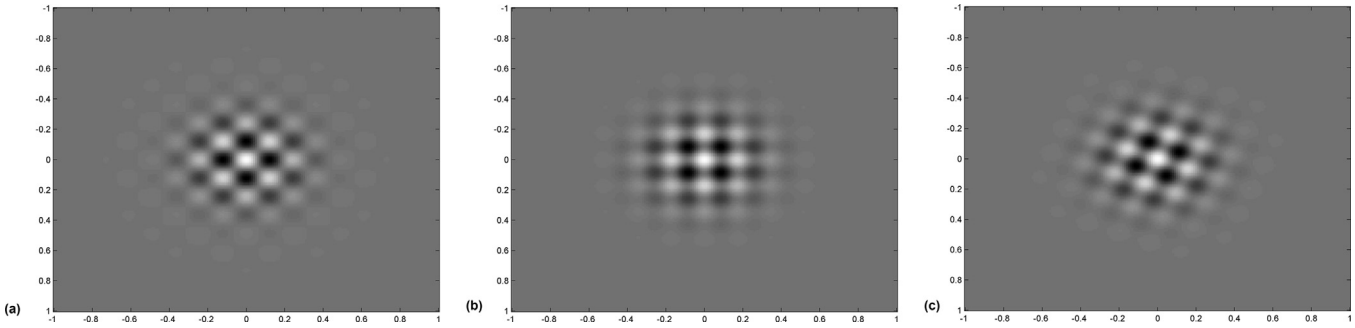


Fig. 7. Effect of changing the rotation parameter R on a Morelet wavelet function with input $(x, y) \in [-1, 1]$: where (α, β, R) = (a) (0.2, 0, 0), (b) (0.2, 0, $\pi/4$), (c) (0.2, 0, $\pi/8$), respectively. Notice how the response is rotated by an amount R.

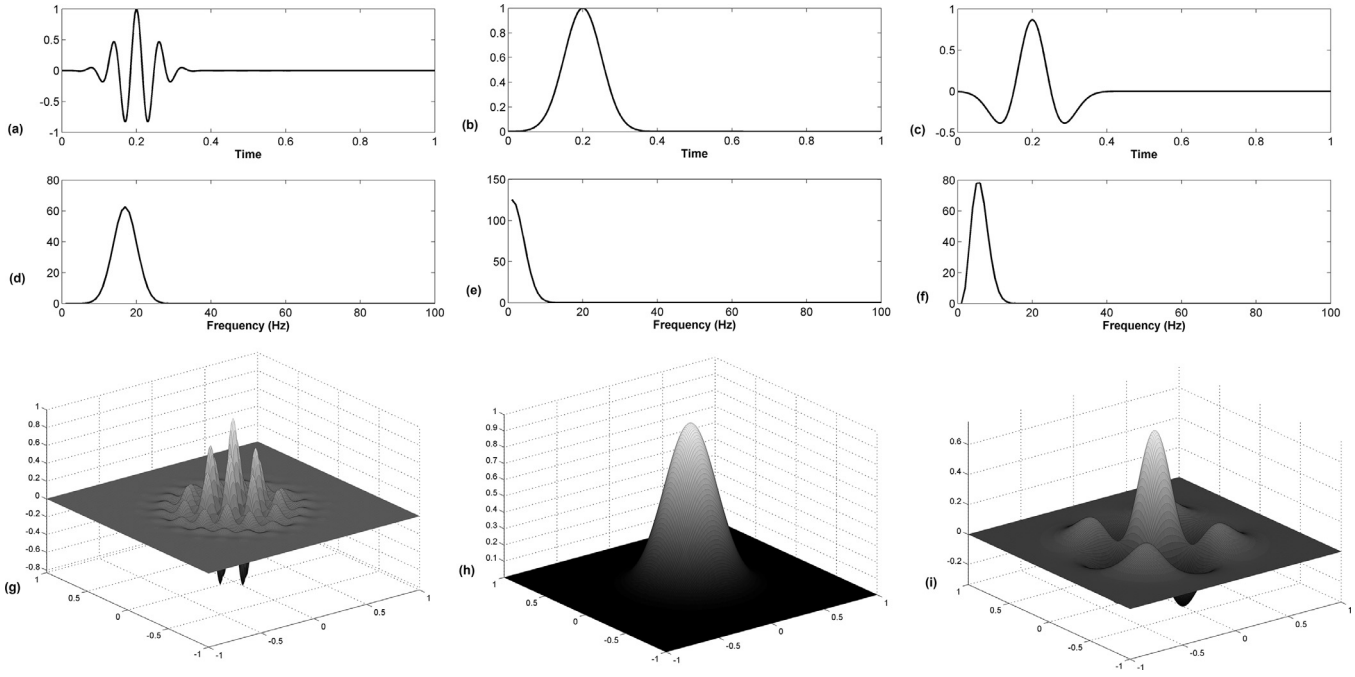


Fig. 8. Three different types of wavelet functions and their frequency responses. (a)–(c) are the time representation of Morelet, Gaussian and Mexican hat wavelets, and (d)–(f) are their corresponding frequency responses (wavelets are sampled with $F_s = 1000$ Hz) with $(\alpha, \beta) = (0.05, 0.2)$. (e–f) 3D surface view of the same wavelets, but with $(\alpha, \beta) = (0.2, 0)$.

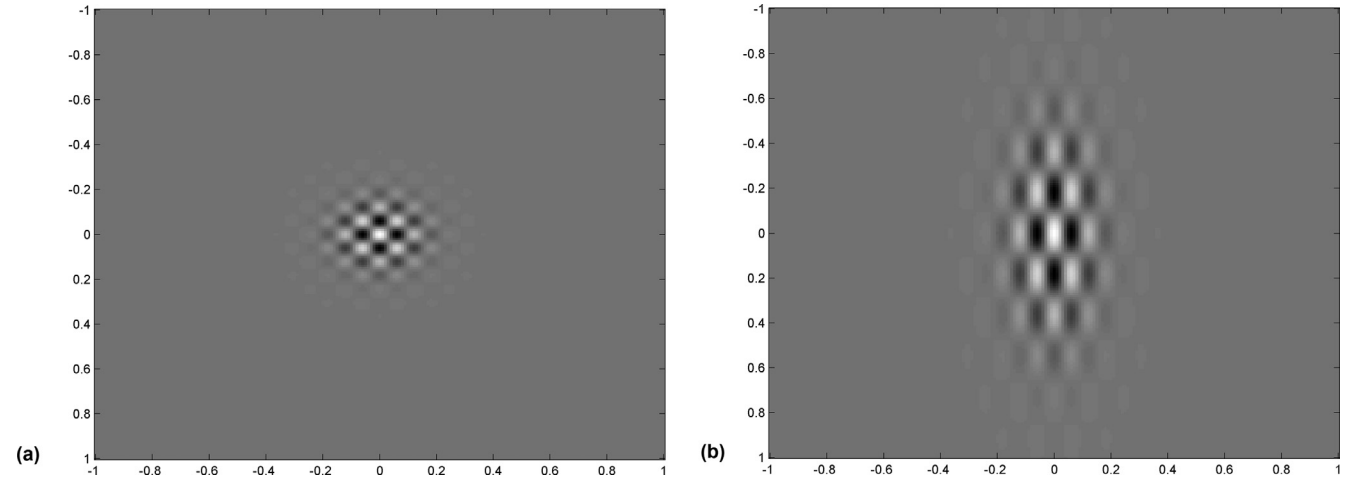


Fig. 9. Given $(x, y) \in [-1, 1]$, response of a Morelet function with (a) one-dimensional dilations and translations with $(\alpha, \beta) = (0.1, 0)$, (b) two-dimensional dilations and translations with $(\alpha_x, \beta_x) = (0.1, 0)$ and $(\alpha_y, \beta_y) = (0.3, 0)$, respectively.

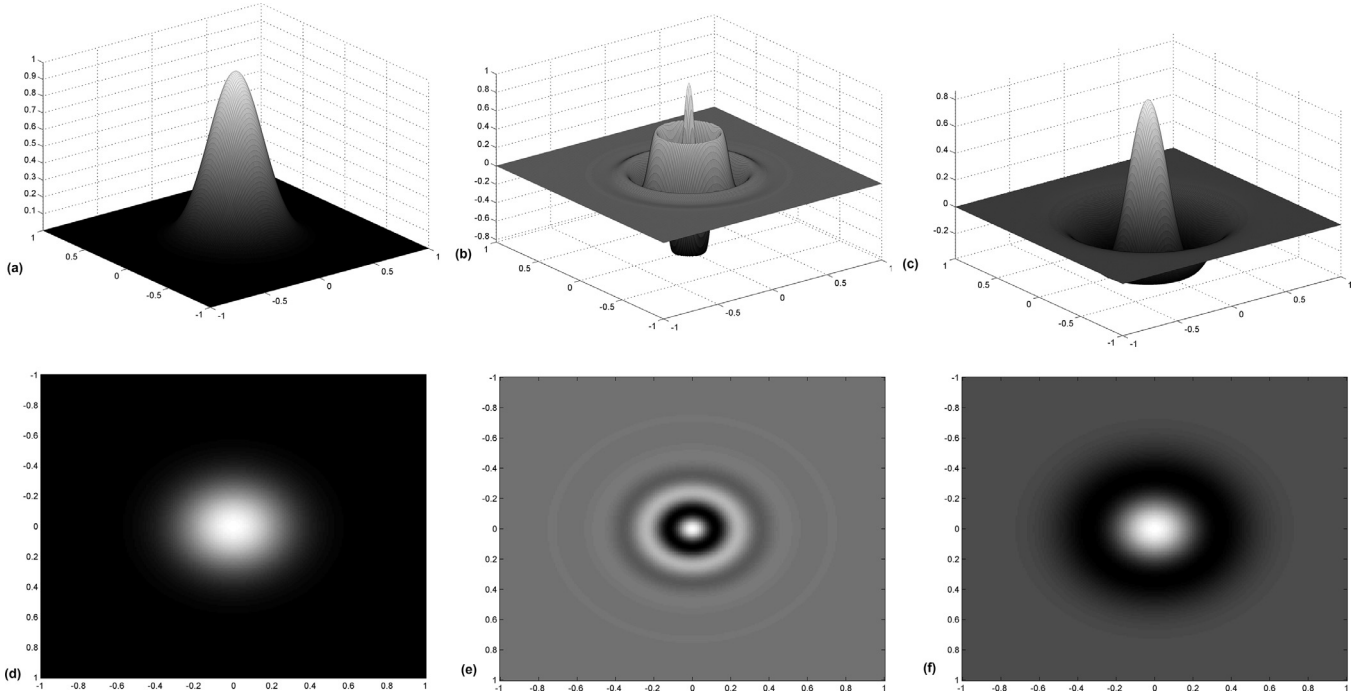


Fig. 10. Radial responses of three common wavelet functions. A 3D surface view of a (a) Gaussian, (b) Morelet and (c) Mexican hat wavelet function, with (d–f) being the flat views of the same functions.

the total number of wavelons in the genome. Each input x_{ij} provided to a wavelon has an associated random weight $w_{x_{ij}} \in$ [-1, 1] for scaling the input x_{ij}, a switch $c_{ij} \in \{0,1\}$ to turn the feature on or off and the scale $\alpha_{ij} \in$ [0, 1], translate $\beta_{ij} \in$ [-1, 1] and rotate $R_i \in$ [-1, 1] parameters to control the behaviour of the wavelet activation functions ψ_i. Each wavelon also carries its own weight $w_{ij} \in$ [-1, 1]. The combination of all the described parameters [x_{ij}, c_{ij}, $w_{x_{ij}}$, α_{ij}, β_{ij}, R_i, ψ_i, w_{ij}] represents a wavelon wav_m in the genome.

The R_i matrix uses singular value decomposition (SVD) to extract an orthogonal matrix for rotating the input features x_{ij}. There are three significant areas where SVD are used: for transforming correlated variables into a set of uncorrelated ones for efficient mapping onto the original data; for ordering the dimensions along which data points exhibit most variation; and to find better approximation by using fewer dimensions useful in data reduction methods. The SVD decomposition of a square matrix R_i of size $n \times n$, where n represents the number of features supplied to each wavelon, decomposes it into three matrices U, S and V:

$$[U, S, V] = SVD(R) \tag{4}$$

U and V are orthogonal matrices of size $n \times n$, and S is a diagonal matrix that contains the singular values. The matrix U is used to rotate the input features x_{ij}. Random orthogonal rotations have been applied to minimize quantization noise in [67,68], with positive results. Computing the SVD of a square matrix is a computational expensive process and takes approximately $O(n^3)$. The number of features supplied to each wavelon controls the computational time of the SVD.

Output Layer with O_n Outputs: The output genome contains either indices of the connected wavelons or consists of direct weighted connections of input ($x_{ij} \times w_{o_{ij}}$). Each of the output neurons has an activation function O_f that represents any function from a predefined lookup table. It has to be kept in mind that an output layer transforms to a perceptron if no wavelons are connected at the output and only the weighted contribution of the input is mapped onto the output layer.

Bias θ: All ANNs have a bias parameter in order to make affine transformations or where systems represent a non-zero mean. In WNN the bias gene $\theta \in$ [-1,1].

2.2.2. Initialization of the network

As mentioned in [58], initializing scale α_{ij} and translation β_{ij} to a random value is not a good strategy because a wavelon's wav_m output would become zero due to its localized properties [44]. Random values may also lead to an increase in training times of the network. The initialization of α_{ij} and β_{ij} is based on the heuristic method proposed by [58] described in Eq. (2).

Apart from scale α_{ij} and translation β_{ij}, the selection of an activation function and the number of wavelons also have a great impact in generating good solutions.

Selection of an activation function depends on the application, but the Mexican hat activation function has been found to perform satisfactorily in many applications [44]. Thus, various activation functions should be tested and chosen accordingly.

Initializing the number of wavelons is critical as too many of wavelons would over-fit and too few would not capture the variability of the data [44]. It has been reported that the number of hidden layer nodes increases to the order of the number of input features to the network [45]. A general rule of thumb is to have the number of wavelons somewhere between the number of input layer size and output layer size [69].

With proper initialization using the above mentioned parameters, the chances of achieving shorter training time and good solutions increases.

2.2.3. Evolution strategy

There are two basic types of evolutionary strategies (μ, λ)-ES and ($\mu + \lambda$)-ES [70]. μ represents the parent population and λ refers to the number of offspring produced in a generation. In (μ, λ)-ES, offspring replaces the parent as the fittest is selected from λ, while in ($\mu + \lambda$)-ES, the fittest is selected from both parents and offspring for the next generation. Cartesian Genetic Programming uses the ($1 + \lambda$)-ES strategy where a single parent is

mutated based on a mutation rate 'τ' to produce λ offspring. The fittest of the genotypes becomes the new parent and is forwarded to the next generation. If a parent and an offspring have the same level of fitness, then the offspring is promoted to the next generation. The algorithm is described in algorithm listing 1.

Algorithm 1 CGPWNN $(\mu + \lambda)$-ES Neuroevolutionary algorithm.

for all j such that $0 \leq j < (\mu + \lambda)$ **do**
  Generate random individual j
end for
Select the top μ individuals as parents
while No solution is found **or** the generation limit is not reached **do**
  for all k such that $0 \leq k < \lambda$ **do**
    Select a random parent from $0 \leq r < \mu$
    Mutate the parent to generate offspring k
  end for
  Select the top μ individuals as parents
end while

2.2.4. Mutation

Like the CGP mutation operator, CGPWNN also uses point based mutation. In this mutation, a randomly chosen location in the genome is changed to a valid value. All random values are pseudo-randomly generated with equal probability. In the current research, the probability of mutation is the same for all genes (alleles) in the genome. A fixed mutation rate of 0.01% [29] is used in all of our experiments. The constraints and the values that each allele can take on are described briefly below:

- $mutate_{w_{x_{ij}}, w_{ij}, \theta, \alpha_{ij}, \beta_{ij}}$: The weight of input $w_{x_{ij}}$, wavelon w_{ij}, bias θ, scale α_{ij} and translate β_{ij} are perturbed by a small percentage of the current real value. An example of the perturbation performed is expressed in Eq. (5):

$$xy = random(-0.1, 0.1)$$
$$w_{ij} = w_{ij} + w_{ij} * xy \tag{5}$$

- $mutate_{func}$: There are two alleles representing functions in the proposed algorithm. One of the genes is a part of a wavelon wav_m - ψ_i and the other is the output neuron function O_f. Thus two functions sets exist; one corresponding to the wavelet activation functions, and the other to the ANN activation functions. The number of functions in each of the sets are user-defined. In our research we have used four wavelet functions Morelet, Mexican hat, Gaussian, and Haar and three activation functions log sigmoid, hyperbolic tangent, and linear. When a function gene is selected, it is randomly changed to another function from the function set.
- $mutate_{c_{ij}}$: A switch gene c_{ij} can either be 0 or 1. So when a switch gene is selected it is simply inverted.
- $mutate_{x_{ij}}$: x_{ij} represents the input features in a system. When an input gene is selected, it is randomly changed to another input feature in the list. For wavelons with input connectivity less than the total number of features, this operation is used, while in a total connected network, only the switch gene operates.
- $mutate_{R_{ij}}$: When a gene selected represents a parameter of the rotation matrix, it is perturbed in a manner similar to weight perturbation.
- $mutate_{o_{ij}}$: An output gene O_n is mutated to a random integer representing an index to either an input x_i or a wavelon ω_m. If the output gene selects an input, then a random weight $w_{ij} \in$ [-1,1] is assigned.

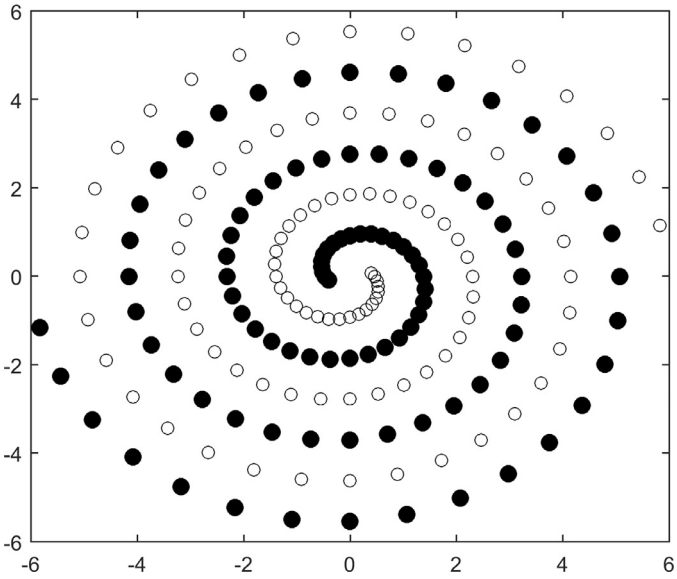


Fig. 11. Training set of points of the two-spiral benchmark task [71].

3. Case study I: Two-spiral task

The two-spiral task is a benchmark task for non-linear classification [71,72]. The dataset consists of two spirals, each with 97 sample data points in a 2D Cartesian space (shown in Fig. 11). The objective is to classify sample points close to each of the spirals by using only the (x, y)-Cartesian coordinates.

In this study, the two-spiral task is investigated under three different configurations of the wavelet neural network.

3.1. Configuration 1: Constructing a WNN using (α, β) with a standard ψ

In this configuration, different types of wavelet functions were tested but only two best scenarios are discussed in detail:

- Two inputs to each wavelon, normalized to $(-1, 1)$; 20 wavelons in the hidden layer using Morelet functions; one neuron in the output layer using a log-sigmoidal function.
- Two inputs to each wavelon, normalized to $(-1, 1)$; 30 wavelons in the hidden layer using a double derivative of Gaussian as the wavelet function; one neuron in the output layer using a log-sigmoidal function.

In both cases, the initial value of (α, β) is (0, 0.1), where β is kept constant throughout the evolutionary process, α is mutated by a random step of $\pm$ 0.05, and input and wavelon weights are perturbed by a maximum of 20% at each mutation step.

Fig. 12 shows snapshots of the response of the CGPWNN with the first scenario. The snapshots are taken at different training accuracies attained across each generation. We can see how the evolutionary process gradually fine-tunes the wavelons to improve classifications.

Fig. 13 shows the response of the CGPWNN with the second scenario. It is clear from the example that the choice of wavelet has a great impact on both training accuracy and classifier response shape.

3.2. Configuration 2: Constructing a WNN using a radial ψ

As the radially symmetric Morelet wavelet (Fig. 10(b)) resembles a spiral, using it as an activation function would intuitively

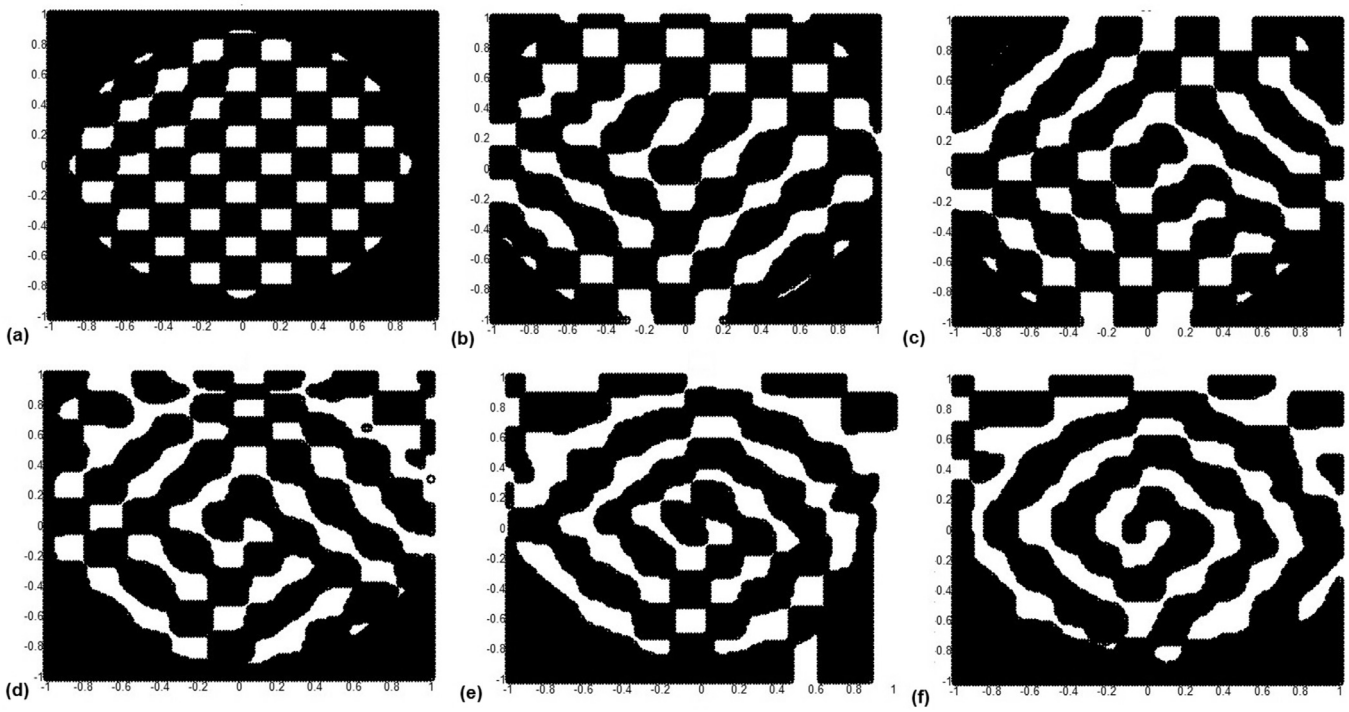


Fig. 12. Responses of a CGPWNN with 20 wavelons in the hidden layer on the two-spiral task with a Morelet wavelet function. (a–f) show the response at different training accuracies with the generation step in brackets: (a) 50% (1) (b) 73.95% (25) (c) 82.81% (50) (d) 89.84% (75) (e) 96.35% (100) (f) 98.95% (109).

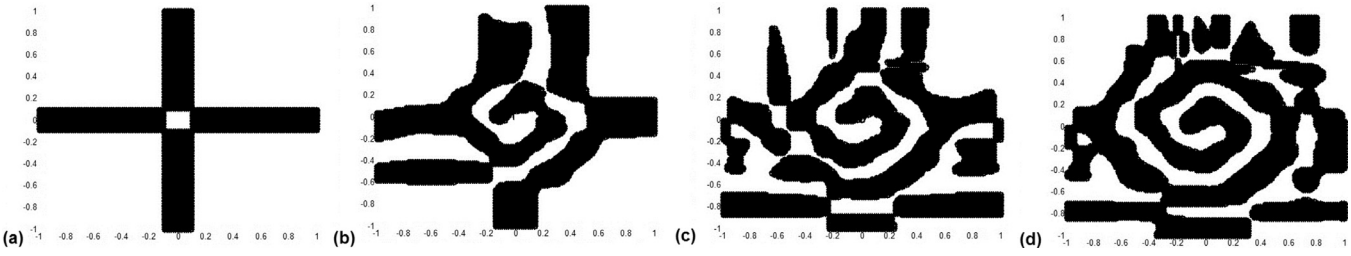


Fig. 13. Responses of a CGPWNN with 30 wavelons in the hidden layer on the two-spiral task with Double Derivative of Gaussian as a wavelet function. (a–d) show the response at different training accuracies with the generation step in brackets: (a) 50% (1) (b) 69% (25) (c) 82% (75) (d) 97% (152).

lead to good quality solutions more quickly. A two input, one output CGPWNN was initialized with two wavelons in the hidden layer. The output neuron had a linear activation function. The training data was normalized to (0, 1). The values of (α, β) were initialized to (0.5, 0.12) and perturbed by a maximum of 20% at each mutation step. The weights of the wavelons were randomly initialized and the input weights were kept constant at 1.0. Fig. 14 displays the outcome of these two wavelons CGPWNN at various stages of the evolutionary process, showing increasing accuracy. The network converged to 100% accuracy in 9 generations.

The WNN had a simple architecture (two wavelons) and was found to converge quickly within 9 generations with high accuracy. In contrast with a standard multilayer perceptron (MLP), Lang and Witbrock [71] were successful in training 2-5-5-5-1 MLPs with additional shortcut connections to succeeding layers. The training of this relatively complex architecture took 10,000 - 20,000 epochs. In [73] a 2-50-1 MLP was trained by employing a second-order Newton optimization method where training took only 650 epochs. Hence, WNNs with powerful activation functions have the ability to approximate functions with a minimum number of wavelons efficiently.

3.3. Configuration 3: Constructing a WNN using (α, β, R) with a standard ψ

As can be seen from Fig. 12, the final spiral image has some artifacts which are due to both choice of the wavelet and the inability to produce smooth curves. Thus, introducing the rotation parameter in the WNN has the possibility to improve the network convergence and quality of the results.

Similar to the first scenario of configuration one 3.1, the CGPWNN had a structure with 20 wavelons in the hidden layer with Morelet activation functions. The wavelon in the output layer had a linear activation function. The (x, y)-coordinates were normalized to the interval $(-1, 1)$. (α, β, R) were initialized to $(0.2, 0.0, \pi/4)$, while the wavelon weights were randomly initialized with values in the interval $(-1, 1)$. The wavelon's weight, α and β were perturbed by 20% during the mutation process while R was randomly mutated to an random angle in the interval $(-\pi, \pi)$. Fig. 15 shows the response of the CGPWNN structure as the evolutionary process progresses, and we can clearly see the improvement in classification.

The case study, demonstrated that WNNs can be effectively applied in general classification tasks, and the performance of the

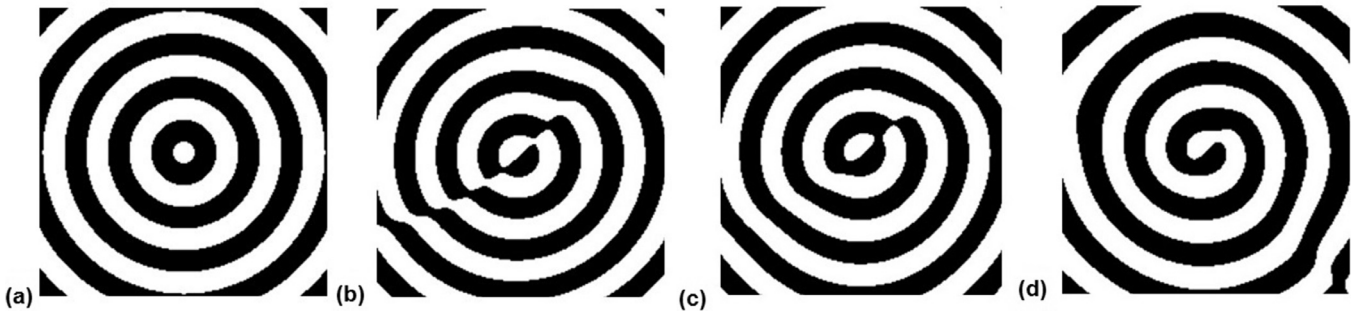


Fig. 14. Responses of a two wavelon CGPWNN on the two-spiral task with radially symmetric Morelet as wavelet functions. (a–d) shows the response at different training accuracies with the generation step in brackets: (a) 50% (1) (b) 96.90% (4) (c) 99.48% (7) and (d) 100% (9).

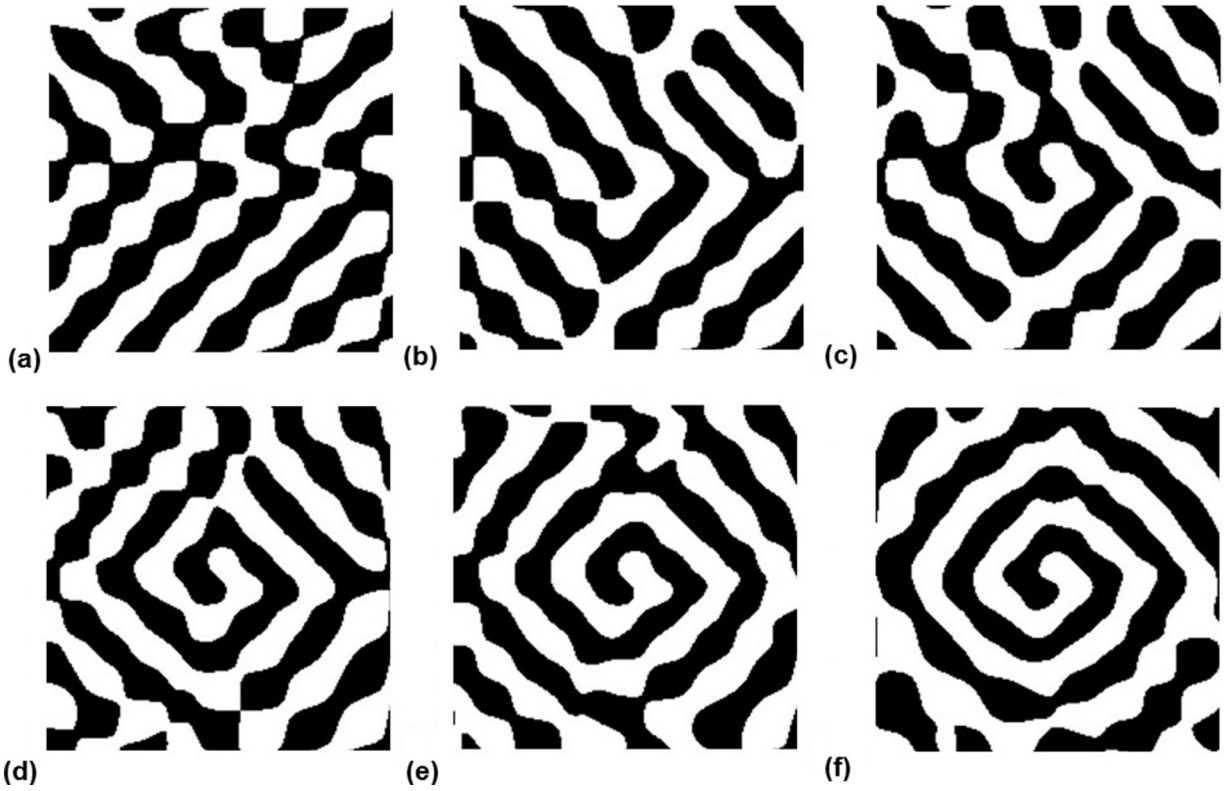


Fig. 15. Responses of a CGPWNN with 20 wavelons in the hidden layer on the two-spiral task with rotation parameter and a Morelet wavelet function. (a-f) shows response at different training accuracies where the generation step is shown in brackets: (a) 59.27% (1) (b) 77.32% (50) (c) 82.47% (100) (d) 98.97% (200) (e) 99.48% (500) and (f) 100% (1030).

WNN can be enhanced by using the rotation parameter for n-dimensional analysis (where $n > 1$).

4. Case study II: Breast cancer classification

The Digital Database for Screening Mammography (DDSM) [74,75] is an online repository of mammographic images of different resolutions obtained from various hospitals. The suspicious areas on the mammograms are manually marked by two experienced radiologists. For analysis, these markings are represented as chain codes and hence can be extracted easily.

In the dataset used by [76], mammographic images scanned by a HOWTEK scanner at 43.5 microns per pixel spatial resolution were downloaded and extracted via the chain code. A total of 25 features were derived from the extracted regions. Details on those features can be found in [77]. For the purpose of fair comparison of the algorithms with other researchers using the same dataset, only six features out of the 25 were used for this study. They are 4 Breast Imaging Reporting and Data System (BIRADS) lexicon features specified by an expert radiologist, and 2 features, namely *patient age* and *subtlety* values, extracted from each mammographic record. The list of these features along with their descriptions is shown in Table 1.

A large number of studies involve the use of DDSM for cancer diagnosis and classification. The literature discussed below focuses on research using the same dataset distribution and features for the purpose of comparing results.

Zhang et al. [78] combined probabilities that were outcomes of statistical classifiers, including Logistic Regression (LR) and Discriminant Analysis (DA) with 14 grey level and 6 human extracted features from the feature set. The updated features were then tested using neural networks (including genetic neural networks) with 3 random splits of the dataset. The combined LRDA-GNN approach obtained a maximum accuracy of 91%. Also, Zhang et al. in [79] used the SPSS software [80] to analyse and extract significant features. Four of these were identified as statistically rel-

Table 1
Features and description of the Digital Database for the Screening Mammography (DDSM) dataset [77].

Features	Description
Human Interpreted Features - BIRADS	
Breast density	Density of breast tissue; rated 1–4
Abnormality assessment rank	Seriousness of abnormality; rated 1–5
Mass shape	Morphological descriptor, e.g. round, oval, lobulated, irregular etc.; rated 1–9
Mass margin	Morphological descriptor, e.g. circumscribed, microlobulated, obscured etc.; rated 1–5
Human Interpreted Features - Others	
Subtlety	Subjective abnormality measure; rated 1–5
Patient age	Age of patient at the time of mammography

evant: assessment, age, margin and shape. These features were then used for classification and tested using neural networks, CART [81–83] and C5.0 [84,85], and an improved accuracy over [78] was obtained. An area under the ROC curve of 0.979 was achieved using 7 human extracted features (6 of which are the same as listed in Table 1) via Logistic Regression [86]. In [76], decision trees were used at different cost ratios on the whole feature set with a 50/50 data split. A maximum of 91% accuracy was obtained with higher variance for a 1:1 cost ratio using CART.

Panchal et al. in [87] divided the dataset based on 6 different feature combinations, and with a 50/50 data split. The main purpose for doing so was to improve performance by detecting key features. The research study identified the grey level features and BIRADS as significant, and a 92% classification accuracy was attained.

Different algorithms were devised by Verma [88–90] for classification and were tested using the dataset. Two soft cluster-based neural networks, namely a Soft Cluster Based Direct Learning (SCBDL) network and a Soft Cluster Neural Network (SCNN), were introduced. A maximum accuracy of 94% in SCNN and 95% in SCBDL was obtained from a 10-fold cross-validation experiment. Other clustering algorithms obtained lower accuracies: SVM = 86.5%, k-means = 84.5% and SOM = 76%. With those other algorithms, the use of an add-on neuron in the hidden layer was proposed to improve both the memorization and generalization ability of the network, with an alternative learning mechanism. The classifier was trained on 6 human extracted features, with a 50/50 split experiment setting, and attained a 94% accuracy.

Mc Leod et al. [91] also tried different mechanisms to augment the classification accuracy for the dataset. They proposed a multi-cluster support vector machine in which a k-means algorithm was used to generate clusters for benign and malignant classes, which were then used for classification by a standard SVM. The method obtained an accuracy of 94.5% using 10-fold cross-validation on 6 human extracted features, while standard SVM alone attained only 87.5%. Also, a 3% rise in the accuracy was achieved when combining the approach with other classifiers including neural networks and radial basis function networks [92]. The accuracy was finally elevated to 99% by using ensemble-based classifiers with 127 classifiers [93–96]. Similar improvement was also observed in an LCA ensemble (94%), compared to LCA alone (87%) in [97].

4.1. Training and testing sets

A subset of suspicious areas was manually extracted from the mammograms in the DDSM dataset. The subset consisted of 200 areas representing 50% benign and 50% malignant tumors. The dataset was then split in two different ways (see below), in order to investigate the effect of division on the performance of the algorithm. Although as discussed earlier, a total of 25 features were extracted from these suspicious areas, in a similar method to Verma's and Mc Leod's studies, 6 human-interpreted features were used in all of our experiments (breast density, mass shape, mass margin, assessment, subtlety and patient age).

- Training on 50% of the data: In this strategy, the dataset was split into 50% training and 50% testing, the same as one of the splits used in [76] and [78]. Each of the sets had an equal contribution of benign and malignant samples.
- 10-fold cross-validation: The dataset is divided into 10 subsets with a random contribution of benign and malignant samples. Training uses 9 of the subsets and testing uses the remaining one. This is repeated 10 times, always with a different selection of subset for testing, and the average accuracy is reported.

4.2. Performance measures

The performance of the classifiers is evaluated based on the following metrics:

1. Training Accuracy (Tr_{Acc}): fraction of correctly trained samples.
2. Testing Accuracy (Te_{Acc}): fraction of correctly classified samples as expressed in Eq. (6), also known as the classification accuracy. The higher the percentage, the better is the classifier performance.

$$Te_{Acc} = \frac{TP + TN}{P + N} \tag{6}$$

where TP represents true positive cases, i.e. accurate classification of benign samples; TN represents true negative cases, i.e. accurate classification of malignant samples; and $(P + N)$ is the total number of positive and negative test samples.

4.3. Experimental setup

CGPWNN structures with different numbers of wavelons (4, 5, 10, 15, 20 & 25) were evolved using the proposed algorithm. Similar to the two-spiral task, the dataset was trained and tested with the rotation parameter enabled and disabled. The number of inputs to each wavelon was equivalent to the number of features, corresponding to 6. There is one output wavelon and a thresholded activation function was used: if the output of the classifier is above zero, the sample is classified as malignant; otherwise it is benign. A $(1 + \lambda)$-ES, with $\lambda = 25$, and a mutation rate of 0.01% was used in all simulations. Each network was evolved for 2000 generations. The wavelet activation functions used were Gaussian, Mexican hat, Morelet and first derivative of Gaussian [29]. Table 2 shows the average performance of the different structures for 50 independent evolutionary runs for all experimental settings. The table shows the figures for training accuracy Tr_{Acc}, testing accuracy Te_{Acc}, active neurons and the number of selected features for the six different structures (S1–S6) evolved.

4.4. Results and discussions

In Table 2, we can see an increase in the training accuracies for all strategies when the number of wavelons in the hidden layer is increased. This indicates that the wavelet network approximates well in the presence of greater numbers of wavelons. On the other hand, we see that no such conclusion can be reached for the testing accuracy.

With the rotation parameter enabled, a maximum of 91.18% in 50/50 split and 89.03% in 10-fold cross-validation strategies was observed, while with the rotation parameter disabled, a maximum of 89.42% in 50/50 split and 88.24% in 10-fold cross-validation was observed. This indicates a possible overfitting of the network in the 10-fold cross-validation scenario in both cases.

Table 2
Performance of a multi-dimensional CGPWNN on the DDSM dataset using (a) a 50/50 split and (b) 10-fold cross-validation strategies. Six different structures (S1–S6) are investigated and their accuracies reported. The dataset is trained and tested with the rotation parameter enabled and disabled respectively. Maximum values are indicated in boldface.

(a) 50/50 Split

Structure:	With rotation				Without rotation			
Wavelons	Accuracy %		Activeparameters		Accuracy %		Active parameters	
	Tr_{Acc}	Te_{Acc}	Neurons	Features	Tr_{Acc}	Te_{Acc}	Neurons	Features
S1:4	93.58	88.42	4	4.98	94.40	86.98	4	5.02
S2:5	94.68	88.60	5	5.18	95.34	88.04	5	5.14
S3:10	96.78	90.30	10	5.96	96.90	88.80	10	5.94
S4:15	97.12	90.46	15	5.98	97.72	88.70	15	5.96
S5:20	97.80	**91.18**	20	6.00	98.08	**89.42**	20	6.00
S6:25	**98.48**	89.74	25	6.00	**98.42**	88.98	25	6.00

(b) 10-fold cross-validation

Structure:	With rotation				Without rotation			
Wavelons	Accuracy %		Active parameters		Accuracy %		Active parameters	
	Tr_{Acc}	Te_{Acc}	Neurons	Features	Tr_{Acc}	Te_{Acc}	Neurons	Features
S1:4	93.74	88.12	4	5.82	93.85	87.99	4	4.82
S2:5	94.16	88.33	5	5.92	94.73	87.70	5.00	5.25
S3:10	95.23	**89.03**	10	6.00	96.60	88.13	10	5.90
S4:15	95.66	88.94	15	6.00	97.06	87.98	15	5.98
S5:20	95.93	88.77	20	6.00	97.37	**88.24**	20	5.98
S6:25	**97.76**	88.63	25	6.00	**97.53**	87.78	25	6.00

For comparison, the numbers of wavelons utilized in [42] were 50 and 100, while in the current study there is a maximum of 25 wavelons. In addition, the study [42] generated only one-dimensional WNN solutions whereas the present study shows an effective utilization of resources in the multi-dimensional strategy for generating WNNs. In addition, unlike the one-dimensional WNN in [42], we can observe that all of the wavelons and features were actively utilized.

Fig. 16 shows the average fitness graph of the two structures *S*4 and *S*5 for both training strategies with rotation parameter enabled. Graphs (a) and (c) relate to structure *S*4, and graphs (b) and (d) to structure *S*5. In the 50/50 split strategy, the network converges to a maximum comparatively quickly. One obvious reason could be the different number of training samples. Also, it took approximately twice the number of generations to reach the same training accuracy for a structure trained with 10-fold cross-validation.

Fig. 17 displays the average of the various point-based parameter mutations of the 50 final genotypes from each structure. We can observe that mutation of rotation had a predominant role in approximating the classifier response. This may also indicate that a proper perturbation rate should have been set in order to avoid multiple mutations because perturbing the rotation matrix is a very sensitive process; a minor change in the matrix translates to huge rotations in the *n*-dimensional space.

Fig. 18 displays the average histogram of the activation functions utilized in the best structures found with both training strategies. The histogram is averaged over 50 independent runs. It can be seen that the Mexican hat, which is the second derivative of Gaussian, has been the most widely used in both strategies. This indicates that Mexican hat is the most effective function for this particular dataset.

For an eight core CPU @3.4 GHz with Windows-7 64-bit and a RAM of 16 GBytes, the average CPU times for evaluating 50 genotypes with rotation enabled and disabled for the largest structure S6 and using the 50/50 split were 15 and 5 h, respectively. This was somewhat expected, as in the rotation enabled case, computing the SVD of λ-offsprings in each generation is a computationally expensive process. Future work will focus on improving the speed by finding efficient strategies for computing rotation matrices.

Table 3
Comparison of different classifiers on the DDSM using 6 features. CGPWNN performs comparably to standard methods of classification.

Algorithm/Reference	Testing acc. (%)
50/50 Data split	
AANN [87]	91.00
CART [76]	91.00
C5.0 [76]	89.00
GANN [76]	89.00
BPNN [76]	88.00
CGPWNN	91.18
10-fold cross-validation	
LCA [97]	87.00
SVM [91]	87.50
K-means [89]	84.50
SOM [89]	76.00
NN [97]	90.00
SCBDL [90]	97.50
SCNN [89]	94.00
MCSVM [91]	94.50
NN Ensemble [95]	99.00
LCA Ensemble [97]	94.00
CGPWNN	89.03

Table 3 shows the performance of different classifiers in classifying the breast mass dataset. We can see that in the 50/50 split case, the proposed CGPWNN classifier has obtained the best result. For the 10-fold cross-validation, the result of the CGPWNN was similar to other non-ensemble methods, including the traditional NN. Further studies focusing on the best structure to be used could improve these results. It has to be kept in mind that with *CGPWNN* the classification has a single step. In other non-ensemble approaches, e.g., SCNN and SCBDL, a two-step process is followed, such as cluster and classify. The current results indicate that wavelet-based neuro-evolutionary algorithms can achieve comparable accuracies to other methods without the use of more complex techniques like pre-processing or ensemble classification. Therefore, there is a clear potential

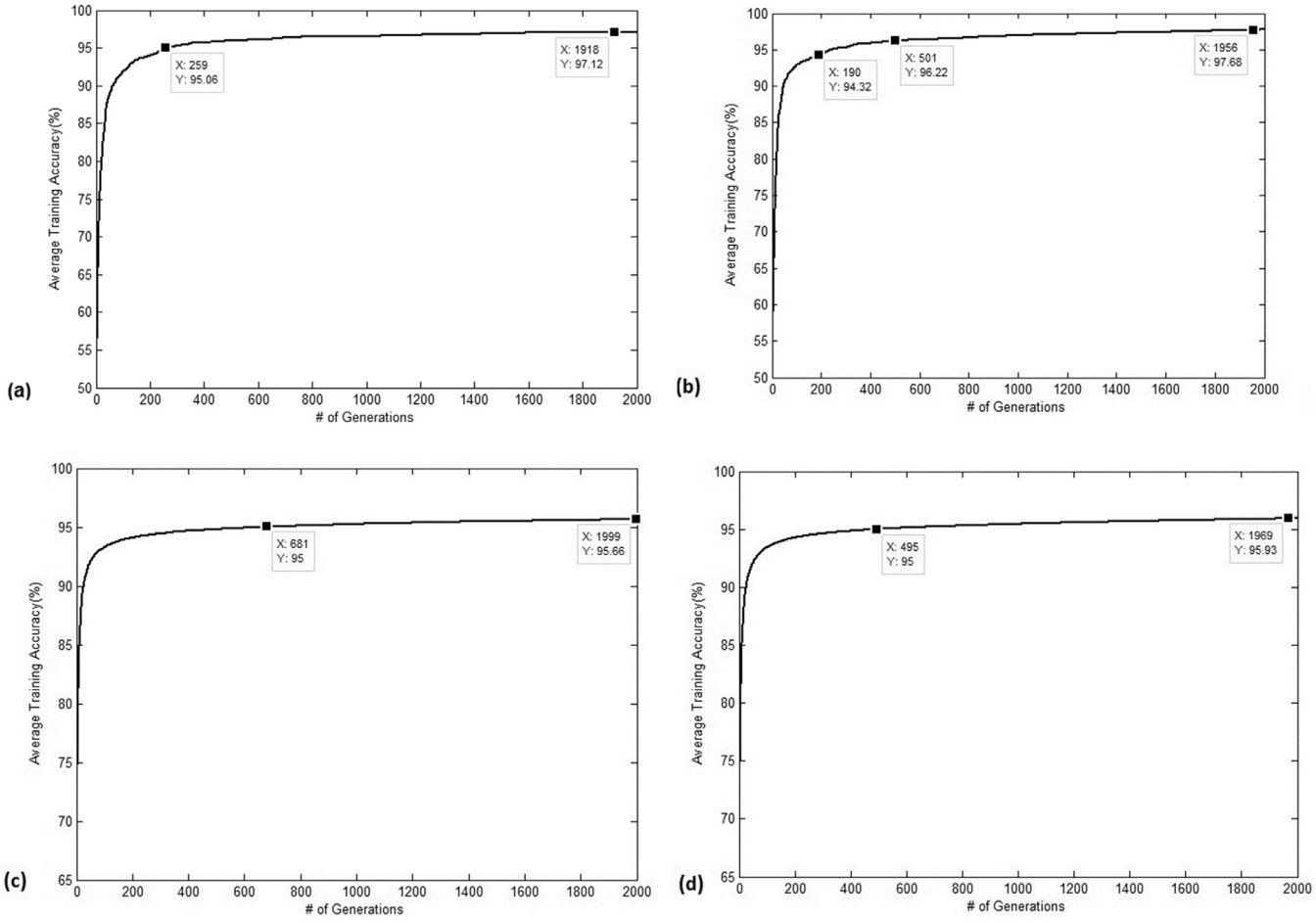


Fig. 16. Fitness graph of two structures of WNNs on DDSM dataset with rotation enabled. (a,b) are the graphs for structures *S*4 and *S*5 with the 50/50 split training set while (c,d) are the graphs for structures *S*4 and *S*5 with the 10-fold cross-validation strategy. Notice the slowed rise in the 10-fold cross-validation strategy, where it took 681 generations to get 95% accuracy while attaining the same at 259 generations in (a).

for further improvement if those strategies are adopted in the future.

5. Case study III: Sakar's Parkinson's disease dataset classification

This part of the research uses a recent, publicly available dataset from an online machine learning data repository from the University of California at Irvine (UCI) [98,99]. The dataset represents features extracted from speech signals of Parkinson's disease (PD) of affected and healthy individuals.

5.1. Database and features

The dataset comprises multiple speech recordings, which include sustained vowels (a, o, u), numbers from 1 to 10, four short rhyming sentences and nine Turkish words from 40 individuals. These recordings comprise 26 records per individual. Half of the individuals are diagnosed with Parkinson's and the other half represents healthy subjects. The software Praat [100], for acoustic analysis, is used to extract twenty six features from the speech signals. The time-frequency and linear-based features extracted from the speech signals can be found in detail in [99].

In [99], Sakar et al. used k-NN and SVM to classify the dataset, using a leave-one-subject-out (LOSO) cross-validation strategy and a summarized leave-one-out (s-LOO) strategy. In the LOSO strategy, a maximum 54.42% classification accuracy was obtained when using k-NN with $k = 5$, while a maximum of 55% was obtained when using an SVM classifier with an RBF kernel. In s-LOO, a new dataset was derived from the existing one. The new dataset represents the central tendency (mean, median, alpha-trimmed (10% and 25%)) and dispersion metrics (standard deviation, interquartile range, mean absolute deviation) of the 26 recordings per individual, thus shrinking the dataset to 40 datarows and increasing the number of features in the feature space by 6×26. Different combinations of feature vectors were tried, e.g., metrics (1–4), (2–5), (3–6) and (1–6). The s-LOO was found to obtain a maximum accuracy of 65% with a k-NN classifier ($k = 7$), and a maximum of 75% when using a SVM classifier with linear kernel and 4×26 features, i.e. mean, median, alpha-trimmed and standard deviation.

To our knowledge, at the time our study was conducted there was only one paper [101] published using the Sakar's Parkinson's disease dataset. We are unable to compare the results with the said author's results, as the number of individuals within the dataset is not consistent with what is available in UCI's repository. The authors of [101] used a dataset with 41 PD and 52 healthy individuals, each with 26 records. The training dataset consisted of 83% of the samples, while the testing set comprised the remaining 17%. Naive–Bayes and k-NN classifiers were used to train and test the individual sample records. The k-NN produced a classification accuracy of 80% while the Naive–Bayes classifier obtained a maximum accuracy of 93.3%.

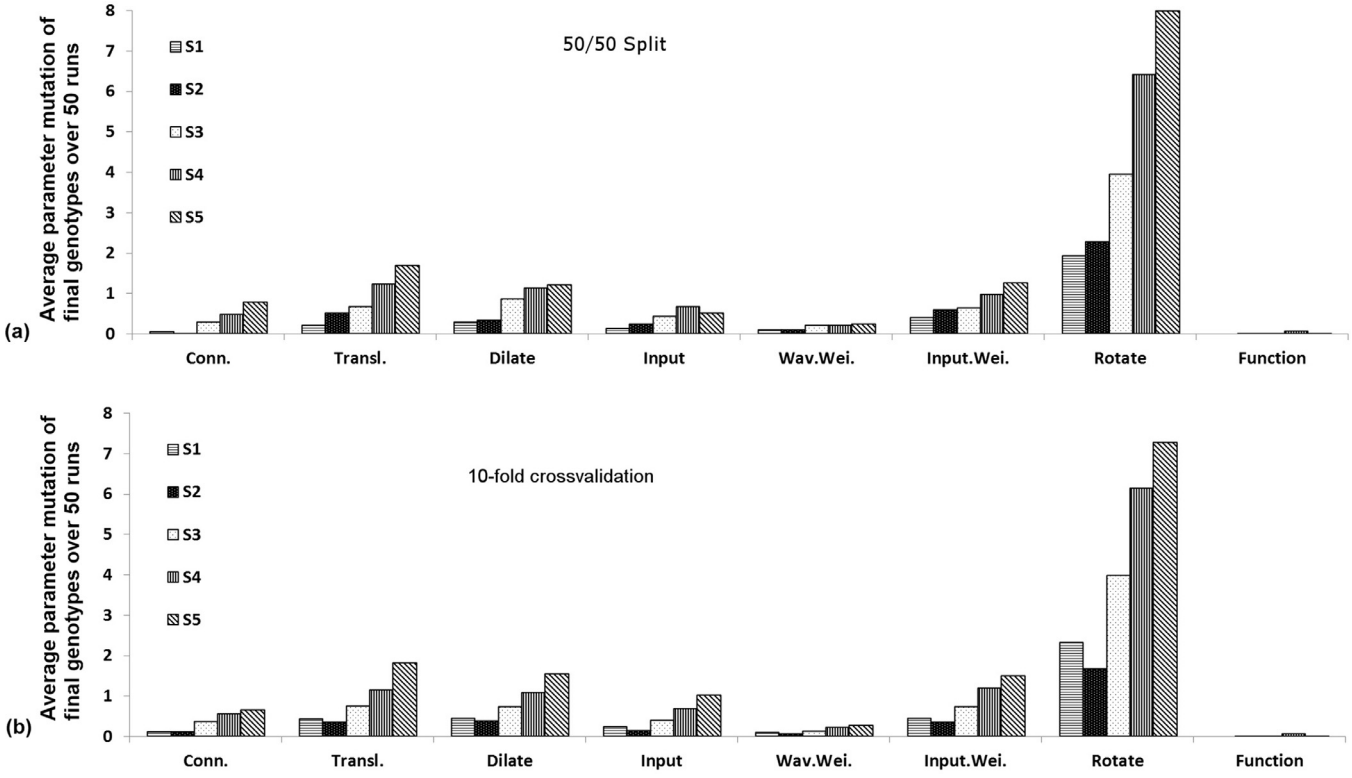


Fig. 17. The two graphs show the average over 50 independent evolutionary runs of the number of mutations (y-axis) of each parameter (x-axis) of the final genotype. This is shown for all structures S1-S5 in different colors. (a) 50/50 split training strategy and (b) 10-fold cross-validation training strategy. We can see that the mutation of rotation had a main role in approximating the classifier response.

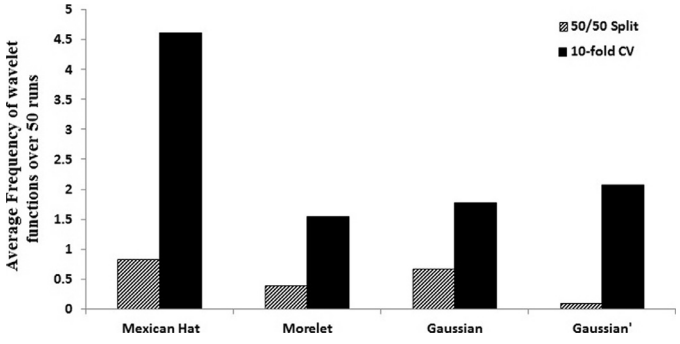


Fig. 18. Frequency of the wavelet function ψ used in the final genotypes of the winning structures. The graph shows the histogram for the 50/50 split and 10-fold cross-validation strategies. The values are the average of 50 independent evolutionary runs. We can see that Mexican hat is the most effective function for this particular dataset.

5.2. Training and testing sets

Two different training and testing sets were generated for the dataset, in order to investigate the performance of the algorithm and compare it with [99].

- Leave-one-subject-out strategy: The dataset consisted of 26 records for each of the 40 individuals, thus making a total of 1040 datarows. In this strategy, all records of a single individual are left out for testing, and those remaining (i.e. 39×26) are used for training the classifier. In this manner, a total of 40 training and testing subsets are generated.
- 10-fold cross-validation strategy: The dataset is divided into 10 training sets with a random individual's record. Each training set consists of the records of 36 individuals while the testing set consists of records of the remaining 4 individuals. The average accuracy for the 10 training/testing sets is then reported. Due to the fact that each individual has 26 records, if more than half of an individual's records are classified as PD, then the individual itself is classified as PD (unhealthy). This approach was adopted from [99,102] in order to avoid over-fitting, as the frequency response of the records of the same patient are potentially very similar.

5.3. Experimental Setup

For Parkinson's disease, given the relative lack of alternative approaches, we tested a traditional technique, namely Support Vector Machine (SVM), and compared those results to the ones from our method. We describe the experimental settings next:

- CGPWNN Parameters: Five different structures $S1$, $S2$, $S3$, $S4$, $S5$, with $1-5$ wavelons in the hidden layer, were used to simulate the wavelet networks, with and without rotation-enabled scenarios. The number of inputs to each wavelon is equal to the number of features, i.e. 26. The output layer of the network has one neuron with a threshold activation function – any value below 0 is categorized as unhealthy. A $(1+\lambda)$-ES, with $\lambda = 25$, and a mutation rate of 0.01% was used in all simulations. Each network was evolved for 1000 generations. The wavelet activation function used was Morelet. Table 4 shows the average performance of the different structures for all the experiment strategies. The table shows the figures for training accuracy Tr_{Acc}, testing accuracy Te_{Acc}, standard deviation std, ac-

Table 4
Performance of multi-dimensional CGPWNN on the Sakar's Parkinson's dataset using (a) 10-fold cross-validation (b) leave-one-subject-out strategies. Five different structures (S1-S5) are investigated and their accuracies reported. The classifiers are trained and tested with rotation parameter enabled and disabled respectively. Maximum values are indicated in boldface.

	(a) 10-fold cross-validation							
Structure:	**With rotation**				**Without Rotation**			
Wavelons	**Accuracy %**		**Active parameters**		**Accuracy %**		**Active parameters**	
	Tr_{Acc}	Te_{Acc}(std)	**Features**	**Neurons**	Tr_{Acc}	Te_{Acc} (std)	**Features**	**Neurons**
S1:1	81.64	65.30 (1.63)	23.41	1.00	84.25	57.60(2.97)	16.24	1.00
S2:2	81.78	66.00 (2.28)	25.75	2.00	**84.27**	57.35(2.07)	21.79	2.00
S3:3	81.75	65.25 (1.74)	25.97	3.00	84.01	56.30(2.45)	24.22	3.00
S4:4	81.67	**66.25** (1.59)	25.99	4.00	83.48	59.20(1.67)	25.21	4.00
S5:5	**81.97**	65.75 (2.91)	26.00	5.00	83.78	**59.60**(2.48)	25.60	5.00
	(b) Leave-one-subject-out							
Structure:	**With rotation**				**Without rotation**			
Wavelons	**Accuracy %**		**Active parameters**		**Accuracy %**		**Active parameters**	
	Tr_{Acc}	Te_{Acc}(std)	**Features**	**Neurons**	Tr_{Acc}	Te_{Acc} (std)	**Features**	**Neurons**
S1:1	80.57	71.85 (4.11)	23.35	1.00	83.03	68.30(2.52)	16.58	1.00
S2:2	80.83	72.40 (3.40)	25.71	2.00	**83.69**	69.10(2.97)	21.70	2.00
S3:3	80.86	70.25 (3.32)	25.97	3.00	83.08	68.75(2.96)	24.23	3.00
S4:4	**81.15**	70.45 (3.17)	26.00	4.00	83.17	68.90(2.57)	25.22	4.00
S5:5	81.02	**73.05** (2.92)	26.00	5.00	82.71	**71.30**(3.52)	25.62	5.00

Table 5
Performance of SVM classifier on leave-one-subject-out and 10-fold cross-validation test strategies with the Sakar's Parkinson's dataset.

Kernel	**LOSO**			**10-fold CV**		
	Parameters	**Accuracy %**		**Parameters**	**Accuracy %**	
		Tr_{Acc}	Te_{Acc}		Tr_{Acc}	Te_{Acc}
Linear	(C=0.0005)	66.73	60.00	(C=0.0005)	68.05	52.50
RBF	(C,σ)=(13.12,3.75)	81.60	65.00	(C,σ)=(13.12,3.75)	79.44	57.50

tive neurons and the number of selected features. The values correspond to the average of '50 × datasubsets' independent evolutionary runs.

- SVM Parameters: An SVM-based classifier was used in this section of the study. The normalized features of the dataset were used, similarly to the CGPWNN. The SVM classifier was implemented using Matlab's toolbox of statistics and machine learning. In our tests, both 'radial basis function (RBF)' and 'linear' kernel were used. A one-dimensional grid search was performed for the linear kernel where the cost parameter C was the search attribute, while a two-dimensional grid search was performed for the RBF kernel where both C and *kernelwidth* σ were searched. C and σ were initialized to [0.001, 15] and [0.001, 10.5], respectively. Table 5 shows the results of the SVM classifier with the maximum classification accuracy for both kernels. The SVM results were also reported by [99], but no grid search technique was employed, and the accuracy obtained was for fixed C and σ values.

5.4. Results and discussion

Table 4 displays the performance of the evolutionary multi-dimensional wavelet neural network with both leave-one-subject-out (LOSO) and 10-fold cross-validation test strategies for rotation enabled and disabled cases, respectively. The LOSO strategy proved to be an effective strategy for classification with a maximum classification accuracy of 73.05% for rotation enabled and 71.30% for rotation disabled, respectively. The 10-fold cross-validation, on the other hand, obtained lower classification rates of 66.25% and 59.60% with rotation enabled and disabled respectively. Similar results were obtained with SVMs, with a maximum accuracy of 65.00% for the LOSO strategy. The grid search was effective in locating the optimal C and σ values and produced better results than those already reported in the literature [99] as shown in Table 6.

Table 6
Comparison of techniques using leave-one-subject-out and 10-fold cross-validation strategies with Sakar's Parkinson's Disease classification. CGPWNN is found to outperform other methods reported in literature.

Algorithm / Reference	Training acc. (%)	Testing acc. (%)
Leave-one-subject-out		
k-NN [99]	–	54.42
SVM Linear (C=10) [99]	–	52.50
SVM RBF (C,σ)=(10,0.005) [99]	–	55.00
SVM Linear *Grid* (C=0.0005)	66.73	60.00
SVM RBF *Grid* (C,σ)=(13.12,3.75)	81.60	65.00
CGPWNN	81.02	73.05
10-fold cross-validation		
SVM Linear *Grid*	68.05	52.50
SVM RBF *Grid*	78.85	57.50
CGPWNN	81.67	66.25

We can see that all wavelons in both structures were active and thus participated in the approximation of the datasets. Unlike the one-dimensional WNN [29], for the multi-dimensional approach almost all features were utilized and active. A minimum average of 23.35 and 23.41 features over 50 independent evolutionary runs were used for the LOSO and the 10-fold cross-validation strategies, with rotation enabled.

From Table 4, we also clearly see an increase in variance of the genotypic solutions for the LOSO strategy as compared to the 10-fold cross-validation strategy. Fig. 19 shows the individual response of the genotypes generated for both strategies and displayed in

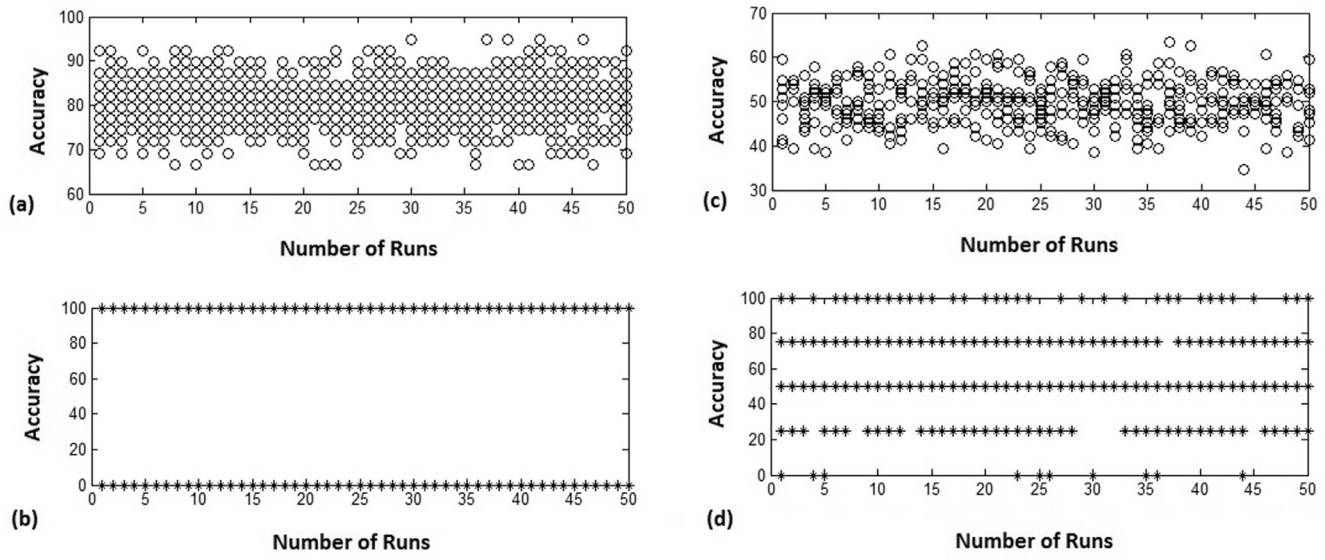


Fig. 19. CGPWNN response (rotation enabled) of the best genotypes for both training strategies on the Sakar's Parkinson's dataset. (a,c) shows the training accuracies and (b,d) testing accuracies of the genotypes for leave-one-subject-out and 10-fold cross-validation strategies, respectively. We can see the different level of classification accuracy attained for both strategies, 2 levels for LOSO and 5 levels for 10-fold cross-validation strategies.

terms of training (a,c) and testing (b,d) accuracies for the rotation enabled case only. The high variance in the genotypic solutions is attributed to the level of accuracy, 2 levels for LOSO and 5 levels for 10-fold cross-validation it can achieve within a strategy, and is depicted in Fig. 19(b,d).

The trend of the average number of mutations of the final genotypes was similar to what was obtained in Section 4: Fig. 17 where the rotation parameter had been found to be mutated a significant number of times.

For a 24 core CPU @3.4*GHz* with Windows-7 64-bit and a RAM of 16 GBytes, the average CPU times for evaluating 50 genotypes with rotation enabled and disabled for the largest structure S5 were 24.72 h and 5.83 h, respectively for the 10-fold cross-validation strategy.

Table 6 includes, for comparison, the results from [99]. The SVM grid search proved to be a better strategy, with a maximum of 65.00% obtained using an 'RBF' kernel. The multi-dimensional wavelet neural network (CGPWNN) has the best classification accuracy among all methods, reaching 73.05% with the leave-one-subject-out strategy.

6. Case study IV: Little's Parkinson's disease dataset classification

This case study uses a Parkinson's disease dataset donated by Max Little to the University of California Irvine's machine learning repository [103,104]. This dataset consists of multiple recordings of the same speech from 31 individuals. Each individual has 6 or 7 speech records. A total of 22 features were extracted from each speech sample using acoustic analysis software. Details of the dataset and the research literature surrounding its usage can be found in [29].

6.1. Experimental setup

Similar to the other case studies above, Table 7 shows four different wavelet neural network classifiers evolved via CGP for both with and without rotation enabled cases, using the 10-fold cross-validation strategy. The number of hidden layer wavelons were 5, 10, 15 and 20, respectively. The number of inputs to each wavelon was equivalent to the number of features, which was 22. There is one output wavelon and a thresholded activation function was used – if the output of the classifier is above zero, the sample is classified as unhealthy and vice versa. A $(1+\lambda)$-ES, with $\lambda = 25$, and a mutation rate of 0.01% was used in all simulations. Each network evolved for 1000 generations. The wavelet activation function used was Mexican hat. Table 7 shows the average performance of the different structures for 50 independent evolutionary runs for the 10-fold cross-validation sets. The table shows the average figures for training accuracy Tr_{Acc}, testing accuracy Te_{Acc}, active neurons and the number of selected features, where the best configurations are marked in bold.

6.2. Results and discussion

Table 8 displays the performance of the evolutionary multi-dimensional wavelet neural network using a 10-fold cross-validation test strategy for both rotation enabled and disabled cases. The rotation enabled WNN produced a maximum classification accuracy of 90.13%, while for the rotation disabled WNN, a maximum accuracy of 88.80% was obtained.

The simulations were performed on a homogeneous WNN with only one activation function (*Mexican hat*) while for a heterogeneous multi-dimensional wavelet neural network [29], an accuracy of 92.93% was obtained. This demonstrates that the use of heterogeneous activation functions can further improve the approximation ability and overall classification rate of the network.

The trend of the average number of mutations of the final genotypes was similar to what was obtained in Section 4:Fig. 17 where the rotation parameter had been found to be mutated a significant number of times.

Table 8 presents the results of different methods of classifying the dataset features. Ensemble methods deliver fairly good performance compared with standalone classifiers. As compared to other algorithms that required pre-processing and filtering, CGPWNN algorithm was found to perform competitively without any additional steps.

Table 7
Performance of multi-dimensional CGPWNN on Little's Parkinson's dataset, using a 10-fold cross-validation strategy. Four different structures (S1–S4) are investigated and their accuracies reported. The classifiers are trained and tested with rotation parameter enabled and disabled, respectively. Maximum values are indicated in bold.

Structure:	Without rotation				With rotation			
Wavelons	Accuracy %		Active parameters		Accuracy %		Active parameters	
	Tr_{Acc}	Te_{Acc}	Neurons	Features	Tr_{Acc}	Te_{Acc}	Neurons	Features
S1:5	91.12	85.33	5	20.19	91.30	85.80	5	18.65
S2:10	95.04	87.93	10	21.94	95.49	88.33	10	21.72
S3:15	96.87	87.40	15	22.00	97.96	89.87	15	21.99
S4:20	**96.97**	**88.80**	20	22.00	**98.21**	**90.13**	20	21.99

Table 8
Comparison of techniques using 10-fold cross-validation strategies in Little's Parkinson's Disease classification. CGPWNN is found to perform competitively in comparison to other classification methods.

Algorithm	Testing acc. (%)
Preselection filter+exhaustive search+SVM [104]	91.4 ± 4.4
mRMR+SVM [105]	92.75 ± 1.21
SVM+MLP-ensemble [106]	90.8
SVM+RBF-ensemble [106]	88.71
SVM+LADTree-ensemble [106]	92.82
SVM+J48-ensemble [106]	92.3
SVM+KSTAR-ensemble [106]	96.41
SVM+IBk-ensemble [106]	96.93
GP-EM [107]	93.12
PCA+FKNN [108]	96.07
GMM+LDA+LS-SVM or PNN or GRNN [109]	100
CGPWNN	90.13

7. Conclusion and future work

Wavelet neural networks (WNNs) combine the characteristics of wavelet transforms and neural networks. They have been the focus of many studies, including studies on time-series prediction and approximation of 1D functions. One of the contributions of our study is the introduction of a rotation parameter for multi-dimensional networks. In addition, we have proposed a genetic algorithm to evolve all of the parameters of the network to obtain better classification accuracies. We have applied these to a benchmark two-spiral classification task and three biomedical datasets for breast cancer and Parkinson's disease data classification.

The two-spiral task was tested on three different settings of the wavelet neural network: a standard WNN with no rotation; a standard WNN with rotation enabled; and a radial WNN. It was found that the WNN with rotation enabled provided more flexibility in generating spiral responses compared to that with rotation disabled. WNNs using radially symmetric activation functions were found to converge to a more accurate solution within fewer network evaluations.

Diagnosing breast cancer was the first task involving the use of a multi-dimensional WNN. It was found that the technique performed similarly to other non-ensemble techniques in literature, achieving a maximum accuracy of 91.18%.

The second task utilizing multi-dimensional wavelet neural networks was the classification of Sakar's Parkinson's disease dataset. The approach achieved a maximum accuracy of 73.05%, with only five wavelons which outperformed other techniques reported in the literature.

In all of the real datasets, rotation was the parameter that mutated the most during the evolutionary process (Fig. 17), and it appears to be critical to the quality of the classification. All features in the datasets were used in the evaluation of the WNNs – which indicates their significance in diagnosis.

The computational time involved in the evolution of the network is relatively high, as the rotation aspect of learning involves the manipulation of an $n \times n$ matrix. Strategies for improving the computational time of the network could be addressed in future studies.

It was also observed that perturbing the rotation matrix is a very sensitive process, where a minor change in the matrix can translate to large rotations in n-dimensional space. Further work could focus on how to control such phenomena more effectively.

Another possible future research direction could be to investigate the behavior of homogeneous and heterogeneous networks in the different case studies.

Acknowledgment

The first author would like to acknowledge the support through an Australian Government Research Training Program Scholarship.

References

[1] S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Trans. Pattern Anal. Mach. Intell. 11 (7) (1989) 674–693, doi:10.1109/34.192463.
[2] A.-G. S. Bao-Guo Xu, Pattern recognition of motor imagery EEG using wavelet transform, J. Biomed. Sci. Eng. 1 (1) (2008) 64–67.
[3] H. Szu, B. Telfer, J. Garcia, Wavelet transforms and neural networks for compression and recognition, Neural Netw. 9 (4) (1996) 695–708, doi:10.1016/0893-6080(95)00051-8.
[4] Z. Zhang, Learning algorithm of wavelet network based on sampling theory., Neurocomputing 71 (1–3) (2007) 244–269. URL http://dblp.uni-trier.de/db/journals/ijon/ijon71.html#Zhang07.
[5] S. Kadambe, P. Srinivasan, Adaptive wavelets for signal classification and compression, Int. J. Electron. Commun. 60 (1) (2006) 45–55, doi:10.1016/j.aeue.2005.01.006. URL http://www.sciencedirect.com/science/article/pii/S1434841105000099.
[6] Z. Bashir, M. El-Hawary, Short term load forecasting by using wavelet neural networks, in: Proceedings of the Canadian Conference on Electrical and Computer Engineering, vol. 1, 2000, pp. 163–166.
[7] H.H. Szu, B.A. Telfer, S.L. Kadambe, Neural network adaptive wavelets for signal representation and classification, Opt. Eng. 31 (9) (1992) 1907–1916, doi:10.1117/12.59918.
[8] H. Szu, B. Telfer, J. Garcia, Wavelet transforms and neural networks for compression and recognition, Neural Netw. 9 (4) (1996) 695–708, doi:10.1016/0893-6080(95)00051-8.
[9] J. Zhang, G.G. Walter, Y. Miao, W.N.W. Lee, Wavelet neural networks for function learning, IEEE Trans. Signal Process. 43 (6) (1995) 1485–1497, doi:10.1109/78.388860.
[10] N. Jin, D. Liu, Z. Pang, T. Huang, Wavelet basis function neural networks, in: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN2007), 2007, pp. 500–505, doi:10.1109/IJCNN.2007.4371007.
[11] H.-J. Yang, X. Hu, Wavelet neural network with improved genetic algorithm for traffic flow time series prediction, Opt. - Int. J. Light Electron. Opt. 127 (19) (2016) 8103–8110, doi:10.1016/j.ijleo.2016.06.017. URL http://www.sciencedirect.com/science/article/pii/S0030402616306386.
[12] S. Yao, C. Wei, Z. He, Evolving wavelet neural networks, in: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, 1995, pp. 1851–1854, doi:10.1109/ICNN.1995.488903.
[13] S. Yao, C. Wei, Z. He, Evolving wavelet neural networks for function approximation, Electron. Lett. 32 (4) (1996) 360–361.
[14] F. Qiu, Y. Li, Air traffic flow of genetic algorithm to optimize wavelet neural network prediction, in: Proceedings of the IEEE International Conference on Software Engineering and Service Science (ICSESS'2014), 2014, pp. 1162–1165, doi:10.1109/ICSESS.2014.6933773.

[15] H. jun Yang, X. Hu, Wavelet neural network with improved genetic algorithm for traffic flow time series prediction, Opt. - Int. J. Light Electron. Opt. 127 (19) (2016) 8103–8110, doi:10.1016/j.ijleo.2016.06.017. URL http://www.sciencedirect.com/science/article/pii/S0030402616306386.

[16] H. Zhao, R. Liu, Z. Zhao, C. Fan, Analysis of energy consumption prediction model based on genetic algorithm and wavelet neural network, in: Proceedings of the 3rd International Workshop on Intelligent Systems and Applications (ISA'2011), 2011, pp. 1–4, doi:10.1109/ISA.2011.5873468.

[17] D. Sahoo, G. Dulikravich, Evolutionary wavelet neural network for large scale function estimation in optimization, in: Proceedings of the 11th Multidisciplinary Analysis and Optimization Conference (AIAA/ISSMO), 2006, pp. 1–11, doi:10.2514/6.2006-6955.

[18] J. Xu, A genetic algorithm for constructing wavelet neural networks, in: Proceedings of the International Conference on Intelligent Computing (ICIC'2006), vol. 4113, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, 2006, pp. 286–291, doi:10.1007/11816157_29.

[19] Y. Luo, G. Nie, A niche hierarchy genetic algorithms for learning wavelet neural networks, in: Proceedings of the 2nd IEEE Conference on Industrial Electronics and Applications, 2007, pp. 960–964, doi:10.1109/ICIEA.2007.4318550.

[20] M. Huang, B. Cui, A novel learning algorithm for wavelet neural networks, in: L. Wang, K. Chen, Y.S. Ong (Eds.), Proceedings of the First International Conference in Advances in Natural Computation, Part I, Springer, Berlin Heidelberg, 2005, pp. 1–7, doi:10.1007/11539087_1.

[21] Y.-C. Huang, C.-M. Huang, Evolving wavelet networks for power transformer condition monitoring, IEEE Trans. Power Deliv. 17 (2) (2002) 412–416, doi:10.1109/61.997908.

[22] H. Liang-yong, H. Sheng-zhong, Immune evolutionary algorithm of wavelet neural network to predict the performance in the centrifugal compressor and research, in: Proceedings of the Third International Conference on Measuring Technology and Mechatronics Automation (ICMTMA'2011), vol. 02, IEEE Computer Society, Washington, DC, USA, 2011, pp. 366–369, doi:10.1109/ICMTMA.2011.378.

[23] G.C. Liao, Application a novel evolutionary computation algorithm for load forecasting of air conditioning, in: Proceedings of the Asia-Pacific Power and Energy Engineering Conference, 2012, pp. 1–4, doi:10.1109/APPEEC.2012.6307573.

[24] N. Chauhan, V. Ravi, D.K. Chandra, Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks, Expert Syst. Appl. 36 (4) (2009) 7659–7665, doi:10.1016/j.eswa.2008.09.019. URL http://www.sciencedirect.com/science/article/pii/S0957417408006702.

[25] M. Awad, Using genetic algorithms to optimize wavelet neural networks parameters for function approximation, Int. J. Comp. Sci. Issues 11 (2) (2014) 256–257.

[26] L. Jinru, L. Yibing, Y. Keguo, Fault diagnosis of piston compressor based on wavelet neural network and genetic algorithm, in: Proceedings of the 7th World Congress on Intelligent Control and Automation (WCICA'2008), 2008, pp. 6006–6010, doi:10.1109/WCICA.2008.4592852.

[27] S. Ling, H. Iu, F. Leung, K. Chan, Improved hybrid particle swarm optimized wavelet neural network for modeling the development of fluid dispensing for electronic packaging, IEEE Trans. Ind. Electron. 55 (9) (2008) 3447–3460, doi:10.1109/TIE.2008.922599.

[28] Y. He, F. Chu, B. Zhong, A hierarchical evolutionary algorithm for constructing and training wavelet networks, Neural Comput. Appl. 10 (4) (2002) 357–366, doi:10.1007/s005210200008.

[29] M.M. Khan, S.K. Chalup, A. Mendes, Parkinson's disease data classification using evolvable wavelet neural networks, in: Proceedings of the Second Australasian Conference on Artificial Life and Computational Intelligence (ACALCI'2016), vol. 9592, Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 113–124. doi:10.1007/978-3-319-28270-1_10.

[30] Z. Guo, G. Liu, D. Li, S. Wang, Self-adaptive differential evolution with global neighborhood search, Soft Comput. (2016) 1–10, doi:10.1007/s00500-016-2029-x.

[31] Z. Guo, X. Yue, H. Yang, K. Liu, X. Liu, Enhancing social emotional optimization algorithm using local search, Soft Comput. (2016) 1–12, doi:10.1007/s00500-016-2282-z.

[32] J.F. Miller, P. Thomson, Cartesian genetic programming, in: Proceedings of the European Conference on GeneticProgramming: EuroGP 2000, Edinburgh, Scotland, UK April 15–16, 2000, vol. 1802, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 121–132. doi:10.1007/978-3-540-46239-2_9.

[33] M. Khan, G. Khan, A. Ahmad, J. Miller, Fast learning neural networks using cartesian genetic programming, Neurocomputing 121 (2013) 274–289.

[34] J. Miller, What bloat? Cartesian genetic programming on boolean problems, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO2001) - Late Breaking Papers, 2001, pp. 295–302.

[35] V.K. Vassilev, J.F. Miller, The advantages of landscape neutrality in digital circuit evolution, in: Proceedings of the Third International Conference Evolvable Systems: From Biology to Hardware:, ICES 2000 Edinburgh, Scotland, UK, April 17–19, 2000, vol. 1801, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 252–263. doi:10.1007/3-540-46406-9_25.

[36] T. Yu, J. Miller, Neutrality and the evolvability of Boolean function landscape, in: Proceedings of the 4th European Conference Genetic Programming:, EuroGP 2001 Lake Como, Italy, April 18–20, 2001, vol. 2038, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 204–217. doi:10.1007/3-540-45355-5_16.

[37] T. Yu, J. Miller, Finding needles in haystacks is not hard with neutrality, in: Proceedings of the 5th European Conference Genetic Programming: EuroGP 2002 Kinsale, Ireland, April 3–5, 2002, vol. 2278, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 13–25. doi:10.1007/3-540-45984-7_2.

[38] M. Khan, G. Khan, J. Miller, Efficient representation of recurrent neural networks for Markovian/non-Markovian non-linear control problems, in: Proceedings of the International Conference on System Design and Applications (ISDA2010), 2010, pp. 615–620.

[39] A. Walker, J.F. Miller, Solving real-valued optimisation problems using cartesian genetic programming, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO2007), 2007, pp. 1724–1730.

[40] J.A. Walker, J.F. Miller, Predicting prime numbers using cartesian genetic programming, in: Proceedings of the 10th European Conference Genetic Programming: EuroGP 2007, Valencia, Spain, April 11–13, 2007., vol. 4445, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 205–216. doi:10.1007/978-3-540-71605-1_19.

[41] J.A. Walker, J.F. Miller, Changing the genospace: Solving GA problems with cartesian genetic programming, in: Proceedings of the 10th European Conference Genetic Programming: EuroGP 2007, Valencia, Spain, April 11–13, 2007, vol. 4445, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 261–270. doi:10.1007/978-3-540-71605-1_24.

[42] M. Khan, S. Chalup, A. Mendes, Evolving wavelet neural networks for breast cancer classification, in: Proceedings of the Twelfth Australasian Data Mining Conference (AUSDM'2014), vol. 158, ACM Digital Library, Brisbane, Australia, 2014, pp. 121–130.

[43] Q. Zhang, A. Benveniste, Wavelet networks, IEEE Trans. Neural Netw. 3 (6) (1992) 889–898.

[44] A. Alexandridis, A. Zapranis, Wavelet neural networks: a practical guide, Neural Netw. 42 (2013) 1–27.

[45] J. Zhang, G. Walter, Y. Miao, W.N.W. Lee, Wavelet neural networks for function learning, IEEE Trans. Signal Process. 43 (6) (1995) 1485–1497, doi:10.1109/78.388860.

[46] X.D.Y.D. Song, Q. Cao, H.R. Karimi, Control strategy based on wavelet transform and neural network for hybrid power system, J. Appl. Math. 2013 (2013) 1–8, doi:10.1155/2013/375840.

[47] Y. Pati, P. Krishnaprasad, Analysis and synthesis of feedforward neural networks using discrete affine wavelet transformations, IEEE Trans. Neural Netw. 4 (1) (1993) 73–85, doi:10.1109/72.182697.

[48] S. Rao, B. Kumthekar, Recurrent wavelet networks, in: Proceedings of the IEEE International Conference on Neural Networks (ICNN1994), vol. 5, 1994, pp. 3143–3147, doi:10.1109/ICNN.1994.374736.

[49] R. Zhang, Y. Chai, X. an Fu, A network traffic prediction model based on recurrent wavelet neural network, in: Proceedings of the 2nd International Conference on Computer Science and Network Technology (ICCSNT2012), 2012, pp. 1630–1633, doi:10.1109/ICCSNT.2012.6526232.

[50] C.-J. Lin, C.-C. Chin, Recurrent wavelet-based neuro fuzzy networks for dynamic system identification, Math. Comput. Model 41 (23) (2005) 226–239, doi:10.1016/j.mcm.2004.05.004. URL http://www.sciencedirect.com/science/article/pii/S0895717705000762.

[51] J.-Y.S. Jung-Heum Yon, Y.-T. Kim, H.-T. Jeon, Dynamic multidimensional wavelet neural network and its applications, J. Adv. Comput. Intell. 4 (5) (2000) 336–340.

[52] K.-C. Kan, K.-W. Wong, Self-construction algorithm for synthesis of wavelet networks, Electron. Lett. 34 (20) (1998) 1953–1955, doi:10.1049/el:19981364.

[53] J. Xu, D.W. Ho, A basis selection algorithm for wavelet neural networks, Neurocomputing 48 (1–4) (2002) 681–689, doi:10.1016/S0925-2312(01)00638-5. URL http://www.sciencedirect.com/science/article/pii/S0925231201006385.

[54] A. Prochazka, V. Sys, Time series prediction using genetically trained wavelet networks, in: Proceedings of the IEEE Workshop on Neural Networks for Signal Processing, 1994, pp. 195–203, doi:10.1109/NNSP.1994.366048.

[55] P. Cristea, R. Tuduce, A. Cristea, Time series prediction with wavelet neural networks, in: Proceedings of the 5th Seminar on Neural Network Applications in Electrical Engineering (NEUREL2000), 2000, pp. 5–10, doi:10.1109/NEUREL.2000.902374.

[56] G.H. Bakir, Foundations of wavelet networks and applications: S. sitharama iyengar, E. C. Cho, Vir V. Phoho; Chapman and Hall/CRC., Neural Netw. 17 (3) (2004) 459. URL http://dblp.uni-trier.de/db/journals/nn/nn17.html#Bakir04.

[57] K. Mak, P. Peng, K. Yiu, L. Li, Multi dimensional complex-valued Gabor wavelet networks, Math. Comput. Model. 58 (11–12) (2013) 1755–1768, doi:10.1016/j.mcm.2013.02.011.

[58] Y. Oussar, G. Dreyfus, Initialization by selection for wavelet network training, Neurocomputing 34 (1–4) (2000) 131–143.

[59] J. Echauz, V. G., Elliptic and radial wavelet neural networks, in: Proceedings of the Second World Automation Congress (WAC1996), vol. 5, 1996, pp. 173–179.

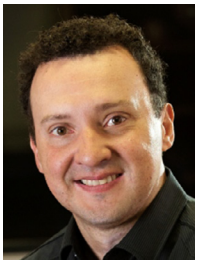
[60] J. Echauz, Strategies for fast training of wavelet neural networks, in: Proceedings of the 2nd International Symposium on Soft Computing for Industry, 3rd World Automation Congress (WAC98), 1998, pp. 1–6.

[61] Z. Zainuddin, K.H. Lai, P. Ong, A hybrid algorithm for the initialization of wavelet neural networks: application in epileptic seizure classification, Int. J. Appl. Phys. Math. 3 (5) (2013) 352–358.

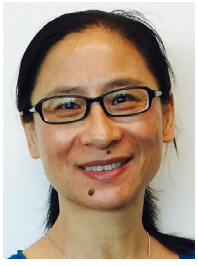
[62] S. Ling, F. Leung, Genetic algorithm-based variable translation wavelet neural network and its application, in: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN2005), vol. 3, 2005, pp. 1365–1370, doi:10.1109/IJCNN.2005.1556073.

[63] C.-J. Lin, Wavelet neural networks with a hybrid learning approach, J. Inf. Sci. Eng. 22 (2006) 1367–1387.
[64] Y. Yu, S. Tan, J. Vaderwalle, E. Deprettere, Near-optimal construction of wavelet networks for nonlinear system modeling, in: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS1996), vol. 3, 1996, pp. 48–51, doi:10.1109/ISCAS.1996.541477.
[65] R. Cheng, Y. Bai, H. Hu, X. Tan, Radial wavelet neural network with a novel self-creating disk-cell-splitting algorithm for license plate character recognition, Entropy (Basel) 17 (6) (2015) 3857–3876, doi:10.3390/e17063857.
[66] J. Guillermo, E. Sanchez, B.C.A. Ricalde L.J. and Cruz, Intelligent classification of real heart diseases based on radial wavelet neural network, in: Proceedings of the Cairo International Biomedical Engineering Conference (CIBEC2014), 2014, pp. 162–165.
[67] Y. Gong, S. Lazebnik, Iterative quantization: a procrustean approach to learning binary codes, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR2011), 2011, pp. 817–824, doi:10.1109/CVPR.2011.5995432.
[68] H. Jegou, M. Douze, C. Schmid, P. Perez, Aggregating local descriptors into a compact image representation, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR2010), 2010, pp. 3304–3311, doi:10.1109/CVPR.2010.5540039.
[69] A. Blum, Neural Networks in C++: An Object-Oriented Framework for Building Connectionist Systems, John Wiley & Sons, Inc., New York, NY, USA, 1992.
[70] H. Beyer, H. Schwefel, Evolution strategies: a comprehensive introduction, Nat. Comput. 1(1) (2002) 3–52.
[71] K.J. Lang, M.J. Witbrock, Learning to tell two spirals apart, in: D. Touretzky, G. Hinton, Sejnowski (Eds.), Proceedings of the 1988 Connectionist Models Summer School, Morgan Kaufmann, Los Altos, CA, 1988, pp. 52–59.
[72] S.K. Chalup, L. Wiklendt, Variations of the two-spiral task, Connect. Sci. 19 (2) (2007) 183–199, doi:10.1080/09540090701398017.
[73] S. Osowski, P. Bojarczak, M. Stodolski, Fast second order learning algorithm for feedforward multilayer neural networks and its applications, Neural Netw. 9 (9) (1996) 1583–1596, doi:10.1016/S0893-6080(96)00029-9.
[74] K. Bowyer, D. Kopans, W.P. Kegelmeyer, R. Moore, K. Chang, S. MunishKumaran, Current status of the digital database for screening mammography, in: Proceedings of the Fourth International Workshop on Digital Mammography, Kluwer Academic Publishers, 1998, pp. 457–460.
[75] M. Heath, K. Bowyer, D. Kopans, R. Moore, W.P. Kegelmeyer, The digital database for screening mammography, in: Proceedings of the Fifth International Workshop on Digital Mammography, Medical Physics Publishing, 2001, pp. 212–218.
[76] K. Kumar, P. Zhang, B. Verma, Application of decision trees for mass classification in mammography, in: Proceedings of the 2nd International Conference on Natural Computation, Advances in Natural Computation and Data Mining, Xidian University Press, China, 2006, pp. 365–375.
[77] P. Zhang, B. Verma, K. Kumar, Neural vs. statistical classifier in conjunction with genetic algorithm based feature selection, Pattern Recognit. Lett. vol. 26 (7) (2005) 909–919.
[78] P. Zhang, K. Kumar, B. Verma, A hybrid classifier for mass classification with different kinds of features in mammography, in: Proceedings of the Second International Conference on Fuzzy Systems and Knowledge Discovery (FSKD'2005), vol. 3614, Springer, Changsha, China, 2005, pp. 316–319.
[79] P. Zhang, K. Kumar, Analyzing feature significance from various systems for mass diagnosis, in: Proceedings of the IEEE International Conference on Computational Intelligence for Modelling Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMA-IAWTIC'2006), 2006, pp. 141–146, doi:10.1109/CIMCA.2006.46.
[80] SPSS, IBM SPSS Software, 2006, Accessed March 2017, URL http://www-01.ibm.com/software/au/analytics/spss/.
[81] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Pacific Grove:Wadsworth, Belmont, CA, 1984.
[82] D. Steinberg, P. Colla, in: CART - Classification and Regression Trees, Salford Systems, San Diego, CA, 1997.
[83] D. Steinberg, M. Golovnya, CART 6.0 User's Manual, in: San Diego CA: Salford Systems, 2006. URL https://www.salford-systems.com/resources/publications/cart-6-0-user-guide.
[84] J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, ISBN 1-55860-238-0, 1993.
[85] RuleQuest Research, C5.0: An informal tutorial, 2010, Accessed March 2017, URL http://www.rulequest.com/see5-unix.html.
[86] P. Zhang, J. Doust, K. Kumar, Presenting a simplified assistant tool for breast cancer diagnosis in mammography to radiologists, in: Proceedings of the Second International Conference on Medical Biometrics (ICMB2010), vol. 6165, Springer, Hong Kong China, 2010, pp. 363–372.
[87] R. Panchal, B. Verma, Neural classification of mass abnormalities with different types of features in digital mammograms, Int. J. Comput. Intell. Appl. 6 (2006) 61–75.
[88] B. Verma, Novel network architecture and learning algorithm for the classification of mass abnormalities in digitized mammograms, Artif. Intell. Med. 42 (2008) 67–79.
[89] B. Verma, P. Mc Leod, A. Klevansky, A novel soft cluster neural network for the classification of suspicious areas in digital mammograms, Pattern Recognit. 42 (2009) 1845–1852.
[90] B. Verma, P. Mc Leod, A. Klevansky, Classification of benign and malignant patterns in digital mammograms for the diagnosis of breast cancer, Expert Syst. Appl. 37 (2009) 3344–3351.
[91] P. Mc Leod, B. Verma, Multi-cluster support vector machine classifier for the classification of suspicious areas in digital mammograms, Int. J. Comput. Intell. Appl. 10 (2011) 481–494.
[92] P. Mc Leod, B. Verma, A classifier with clustered sub classes for the classification of suspicious areas in digital mammograms, in: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN2010), Barcelona, 2010, pp. 1–8, doi:10.1109/IJCNN.2010.5596832.
[93] P. Mc Leod, B. Verma, Clustered ensemble neural network for breast mass classification in digital mammography, in: Proceedings of the World Congress on Computational Intelligence (WCCI2012), IEEE, Brisbane Australia, 2012, pp. 1–6.
[94] P. Mc Leod, B. Verma, A multilayered ensemble architecture for the classification of masses in digital mammograms, in: Proceedings of the 25th Australasian Joint Conference Advances in Artificial Intelligence AI 2012: Sydney, Australia, December 4–7, 2012, vol. 7691, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 85–94. doi:10.1007/978-3-642-35101-3_8.
[95] P. Mc Leod, B. Verma, Effects of large constituent size in variable neural ensemble classifier for breast mass classification, in: Proceedings of the 20th International Conference on Neural Information Processing (ICONIP'2013), vol. 8228, Springer Berlin Heidelberg, Daegu, Korea, 2013, pp. 525–532.
[96] P. Mc Leod, B. Verma, Variable hidden neuron ensemble for mass classification in digital mammograms [application notes], IEEE Comput. Intel. Mag. 8 (1) (2013) 68–76, doi:10.1109/MCI.2012.2228598.
[97] S. Pour, P. Mc Leod, B. Verma, A. Maeder, Comparing data mining with ensemble classification of breast cancer masses in digital mammograms, in: Proceedings of the Second Australian Workshop on Artificial Intelligence in Health: AIH 2012, 2012, pp. 55–63.
[98] UCI machine learning repository; parkinson speech dataset with multiple types of sound recordings data set, URL http://archive.ics.uci.edu/ml/datasets/Parkinson+Speech+Dataset+with++Multiple+Types+of+Sound+Recordings. (Accessed May 2014).
[99] B. Sakar, M. Isenkul, C. Sakar, A. Sertbas, F. Gurgen, S. Delil, H. Apaydin, O. Kursun, Collection and analysis of a parkinson speech dataset with multiple types of sound recordings, IEEE J. Biomed. Health Inform. 17 (4) (2013) 828–834, doi:10.1109/JBHI.2013.2245674.
[100] P. Boersma, D. Weenink, Praat: doing phonetics by computer, URL http://www.fon.hum.uva.nl/praat/. (Accessed April 2015)
[101] Y. Alemami, L. Almazaydeh, Detection of parkinson disease through voice signal features, J. Am. Sci. 10 (2014) 44–47.
[102] M. Caglar, B. Cetisli, I. Toprak, Automatic recognition of Parkinson's disease from sustained phonation tests using ann and adaptive neuro-fuzzy classifier, J. Eng. Sci. Des. 1 (2) (2010) 59–64.
[103] M. Little, P. McSharry, R. S.J., D. Costello, I. Moroz, Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection, Biomedical Eng. Online 6 (23) (2007), doi:10.1186/1475-925X-6-23.
[104] M. Little, P. McSharry, E. Hunter, J. Spielman, L. Ramig, Suitability of dysphonia measurements for telemonitoring of Parkinson's disease, IEEE Trans. Biomed. Eng. 56 (4) (2009) 1015–1022.
[105] C. Sakar, O. Kursun, Telediagnosis of Parkinson's disease using measurements of dysphonia, J. Med. Syst. 34 (4) (2010) 591–599.
[106] A. Ozcift, SVM feature selection based rotation forest ensemble classifiers to improve computer-aided diagnosis of Parkinson disease, J. Med. Syst. 36 (4) (2012) 2141–2147.
[107] P. Guo, P. Bhattacharya, N. Kharma, Advances in detecting Parkinson's disease, in: Lecture Notes in Computer Science (LNCS), vol. 6165, Springer, 2010, pp. 306–314.
[108] H.-L. Chen, C.-C. Huang, X.-G. Yu, X. Xu, X. Sun, G. Wang, S.-J. Wang, An efficient diagnosis system for detection of Parkinson's disease using fuzzy k-nearest neighbor approach, Expert Syst. Appl. 40 (1) (2013) 263–271, doi:10.1016/j.eswa.2012.07.014.
[109] M. Hariharan, K. Polat, R. Sindhu, A new hybrid intelligent system for accurate detection of Parkinson's disease, Comput. Methods Programs Biomed. 113 (3) (2014) 904–913, doi:10.1016/j.cmpb.2014.01.004. URL http://www.sciencedirect.com/science/article/pii/S0169260714000054.

Maryam Mahsal Khan did her B.Sc. Computer System Engineering from University of Engineering & Technology Peshawar, Pakistan in 2005 and Masters in Electrical & Electronic Engineering from Universti Teknologi Petronas, Malaysia in 2008. Before commencing her Ph.D. at the University of Newcastle, she worked as an Assistant Professor at UET Peshawar, Pakistan and later as a Research Engineer at LMKR Pvt. Ltd, Islamabad, Pakistan. She has a keen interest in Non-Linear Control, Genetic Algorithms and Genetic Programming, Artificial Neural Networks, Pattern Recognition, Image Processing, Signal Processing, Time-frequency decomposition. She has a range of publications in these fields in the conferences of repute.



Alexandre Mendes received his Ph.D. degree in Electrical Engineering from the State University of Campinas, Brazil, in 2003. He is a Senior Lecturer with the School of Electrical Engineering and Computer Science at The University of Newcastle, Australia. His research interests include optimization and data mining, with applications in bioinformatics, robotics and operations research.



Dr. Ping Zhang is a Research Fellow at Menzies Health Institute Queensland, Griffith University Australia. She has worked in bioinformatics and health informatics research area in the last 10 years. Her research interests include pattern recognition, biomarkers discovery, vaccine target identification and applying machine learning and statistical techniques for medical decision making.

Stephan Chalup is an Associate Professor in Computer Science and Software Engineering at the University of Newcastle, Australia, where he leads the Interdisciplinary Machine Learning Research Group. He received his Ph.D. (Machine Learning) in 2002 from Queensland University of Technology in Brisbane, Australia. His research interests include manifold learning, kernel machines, humanoid robots, computer vision, and neural information processing systems.