Soft Comput (2017) 21:4635–4659
DOI 10.1007/s00500-016-2074-5

A modular ridge randomized neural network with differential evolutionary distributor applied to the estimation of sea ice thickness

Ahmad Mozaffari[1] · K. Andrea Scott[1] · Shojaeddin Chenouri[2] · Nasser L. Azad[1]

Published online: 18 February 2016

Abstract In this paper, a sequential intelligent methodology is implemented to estimate the sea-ice thickness along the Labrador coast of Canada based on spatio-temporal information from the moderate resolution imaging spectro-radiometer, and the advanced microwave scanning radiometer-earth sensors. The proposed intelligent model comprises two separate sub-systems. In the first part of the model, clustering is performed to divide the studied region into a set of sub-regions, based on a number of features. Thereafter, this learning system serves as a distributor to dispatch the proper information to a set of estimation modules. The estimation modules utilize ridge randomized neural network to create a map between a set of features and sea-ice thickness. The proposed modular intelligent system is best suited for the considered case study as the amount of collected spatio-temporal information is large. To ascertain the veracity of the proposed technique, two different spatio-temporal databases are considered, which include the remotely sensed brightness temperature data at two different frequencies (low frequency, 6.9 GHz, and high frequency, 36.5 GHz) in addition to both atmospheric and oceanic variables coming from validated forecasting models. To numerically prove the accuracy and computational robustness of the designed sequential learning system, two different sets of comparative tests are conducted. In the first phase, the emphasis is put on evaluating the efficacy of the proposed modular framework using different clustering methods and using different types of estimators at the heart of the estimation modules. Thereafter, the modular estimator is prepared with standard neural identifiers to prove to what extent the modular estimator can increase the accuracy and robustness of the estimation.

Keywords Sea-ice thickness · Ridge randomized neural network · Unsupervised information clustering · Differential evolution

1 Introduction

Estimation of sea-ice thickness in the Arctic is a very crucial task. Improved estimates of sea-ice thickness should lead to safer navigation in ice-infested waters, as well as improved estimates of ocean-atmosphere heat transfer, and improved weather forecasting. It has been reported that there is an increasing need for developing accurate and comprehensive forecasting tools that can estimate crucial sea-ice parameters, such as sea-ice thickness, in different regions (Eicken 2013). It is clearly necessary to conduct research on novel tools that can estimate sea ice thickness from observational data, as well as on improving the accuracy of the existing prognostic forecast models. The present contribution is on the first of these two objectives and proposes an intelligent method to retrieve sea-ice thickness from geophysical data. The method is different from those commonly used in the sea ice community (Iwamoto et al. 2013; Nihashi et al. 2009; Kaleschke et al. 2012), in that it is a black box approach, meaning that it is not based on any physical principles. While the method presented here uses brightness temperatures from passive microwave

Communicated by V. Loia.

Nasser L. Azad
nlashgar@uwaterloo.ca

K. Andrea Scott
ka3scott@uwaterloo.ca

[1] Department of Systems Design Engineering, University of Waterloo, Waterloo, Canada

[2] Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, Canada

radiometers in addition to data from forecast models, it could in principle be modified to accommodate other types of data.

Data from passive microwave sensors are widely used for sea ice monitoring because these sensors are capable of returning surface information during both dark and cloudy periods. In addition to providing indispensable ice concentration estimates to the scientific community, it has been observed that ice thickness information can be obtained from these data by exploiting the correlation between the horizontally and vertically polarized emissions measured by passive sensors and the thickness of thin ice (Nihashi et al. 2009; Iwamoto et al. 2013). It has been found that this correlation is stronger for the lower frequency channels on a passive microwave sensor than for the higher frequency channels, even when the data from the higher frequency channels have been screened for atmospheric contamination (Scott et al. 2014). This is due to the fact that the emitting layer is farther below the surface for lower frequencies (Weeks 2010).

Research conducted on how to select the most influential observational data, or on how to use this information for developing estimation tools, is often based on the underlying physical/environmental characteristics of the remote sensing measurement and the region of study. However, when developing physical laws, several important issues may arise. First, if the objective is to develop high-fidelity physical models, and if it is optimistically assumed that most of the important elements are taken into account in these models, there is a possibility that the resulting model possesses a complicated structure. Significant effort may be required to identify the unknown structural parameters of the model (Pratama et al. 2014b). The model may also be computationally very expensive.

The difficulties associated with developing accurate and efficient physical models have led to alternative and complementary research on establishing new surrogate models. Indeed, as the quantity of available data is increasing, there is a possibility to develop accurate black-boxes, which are ignorant of the physical situation. Such a trend is, in fact, in line with a global interest towards concentrating on developing automated machines for reducing the amount of human inference in data analysis (Pratama et al. 2014a).

Over the past decade, enormous progress has been made to foster the applicability of intelligent techniques as black boxes capable of creating an accurate nonlinear map between a set of input and output data pairs (Meireles et al. 2003). In this context, most of the proposed learning systems are designed to prepare intelligent systems for a wide range of applications in which there is need for incremental and recurrent learning (Pratama et al. 2015a), evolving classification and regression (Pratama et al. 2015b, c). However, there exist rare reports in the literature of sea-ice thickness estimation focusing on the use of intelligent identifiers. By surveying the archived literature, a limited number of relevant research papers have been found with a meaningful contribution toward advocating the use of intelligent identifiers for modeling of sea-ice thickness. Haverkamp et al. (1995) proposed an intelligent technique for estimation of sea-ice thickness from a synthetic aperture radar (SAR) database. The simulation results clearly verified that the developed intelligent system can accurately identify the variations in ice thickness to regional and seasonal changes. Soh et al. (2004) developed a hybrid intelligent system for satellite sea-ice image analysis. The developed intelligent framework was able to perform both feature selection and rule-base classification using the data coming from sensors. The simulation results endorse the efficiency of the proposed intelligent paradigm. Lin and Yang (2012) proposed a hybrid algorithm based on chaotic immune genetic algorithm and back-propagation neural network (BP-NN) to predict the sea-ice thickness in the Bohai Sea and the northern region of the Yellow Sea. The results of the conducted simulations clearly demonstrated both the accuracy and robustness of the proposed estimator. Belchansky et al. (2008) developed an in-situ learned and empirically derived neural network model to estimate the fluctuation of Artic sea ice thickness. One of the important characteristics of the developed neural network was that it could accurately predict the sea-ice thickness under different conditions with a very high accuracy.

In the pursuit of addressing the main open concerns on the potential of intelligent methods for the estimation of geophysical varaibles, here, the authors would like to propose a novel technique based on the concept of modular neurocomputing (Rojas 1996). To the author's best knowledge, this is the first time such an intelligent system has been used for the estimation of a geophysical variable, such as sea-ice thickness. Furthermore, this research intends to clearly demonstrate how a relatively sophisticated intelligent tool can be used to (1) process spatio-temporal data coming from two satellite sensors, the moderate resolution imaging spectro-radiometer (MODIS), and advanced microwave scanning radiometer-earth (AMSR-E) observing system,(2) develop a modular intelligent tool that can separate the information coming from different regions, and (3) estimate the sea-ice thickness using a set of independent identification modules allocated to each of those independent regions.

The rest of the paper is organized as follows: Section 2 is devoted to a brief review of the background of modular identification systems. Also it is discussed why a modular system is potentially a good approach for estimation of sea-ice thickness over a given geographic region, which is chosen here as the Labrador coast, along the east coast of Canada. The description of the studied region along with the traits of the collected data from MODIS and ASMR-E sensors are presented in Sect. 3. The details of the developed model are given in Sect. 4. Section 5 is devoted to the description of experimental setup. Finally, conclusions are given in Sect. 6.

2 Modular neural networks: a concise review

In the field of data mining, one often encounters complicated databases comprising several independent sub databases each having its own characteristics and features. To process such complicated information, the human brain uses an automatic procedure that first tries to discriminate the sub-processes and then processes the data corresponding to each sub-process. Inspired by the approach taken by the human mind, a novel computing field has emerged within the realm of intelligent computing, known as modular data processing (Fodor 1983). Modular data processing can be used for developing both modular regression and modular classification tools. The main idea behind such a scheme is to use some similarity and discrimination criteria to first divide the database into a set of sub databases that has similar characteristics, and thereafter allocate independent regression/classification methods to each of those independent databases (Melin 2012). Modular computational frameworks can have any type of identifiers, e.g. neural networks (NNs), neuro-fuzzy systems, and fuzzy inference systems, in their architecture (Melin 2012). Each of these identifiers has its own pros and cons, and thus, can give specific features to the resulting modular system. Due to the straightforward and beneficial computational characteristics of neurons in neural networks, most of the existing modular architectures use such identifiers in their structure. A modular NN has a relatively complex architecture including a series of independent NNs working together in a systematic fashion to predict the same output (Rojas 1996).

The key point lies in the fact that each of those sole NNs try to fulfill a specific task by focusing on separate portions of the input space from the database. So, the systemic integration of the prediction of those independent NNs can result in a modular NN capable of performing a unique task. Such an approach has an obvious advantage from the data mining viewpoint, especially when dealing with complicated information systems with multiple data streams. In fact, by doing so, a complex large data-mining task is reduced to a number of smaller and manageable data mining tasks, and a higher level of manipulation can be achieved to have a much more robust and accurate result. The other advantage of modular NNs lies in their intrinsic parallel architecture enabling them to perform the entire task in a parallel fashion. This intuitively increases the speed and efficacy of computation (Farooq 2000).

An enormous amount of applied and theoretical research has been carried out to demonstrate the applicability of such systems for different tasks. Providing a detailed chronological review on the progress of modular intelligent data processing is clearly beyond the scope of the current investigation. The interested readers are referred to some special issues and seminal books published regarding the advances in modular data processing (Rojas 1996; Farooq 2000). In general, the present research clearly indicates that modular NNs not only can increase the robustness and accuracy of the data mining task, but also can do this job very efficiently. The same observation has been reported by the scientists working within various realms, such as medicine, systems sciences, and manufacturing (Ding et al. 2014; Javadi et al. 2013). Such promising reports have instigated the authors to evaluate the potential of modular intelligent computing for estimating sea-ice thickness. The main reasons behind such a motivation can be listed as follows:

(a) The quantity of data collected from satellites is quite large, and are often combined with data from other sources (e.g., forecast model output) to produce several streams of spatio-temporal information. It is desirable to develop a modular data mining tool capable of reducing the complexity of the resulting database, while achieving accurate and robust estimates of geophysical variables.

(b) Modular neural networks have clearly demonstrated their high potential in various fields of engineering. However, there are fewer applications of such systems for geoscience tasks, in particular estimation of sea-ice thickness. Therefore, the current investigation tries to excavate the computational potential of modular identifiers for the estimation of sea-ice thickness.

(c) The current investigation also contributes to the field of modular computing by proposing a novel sequential system that uses a differential evolutionary algorithm for clustering, and ridge randomized neural network for estimation. To demonstrate the computational advantages of the proposed modular architecture, here, the authors use some well-known rival modular neural networks for the same estimation task.

The detailed procedures required for the implementation of the proposed modular estimator will be given in the next sections.

3 Description of collected database

In this section, the details of the studied region located along the east coast of Canada as well as the characteristics of data from the two considered remote sensors, i.e. ASMR-E and MODIS, are discussed.

3.1 Region of Study

As mentioned, in this study, the emphasis is put on measuring the sea-ice thickness along the east coast of Canada including sea-ice along the Labrador coast, and the northern coast of Newfoundland, as indicated in Fig. 1.

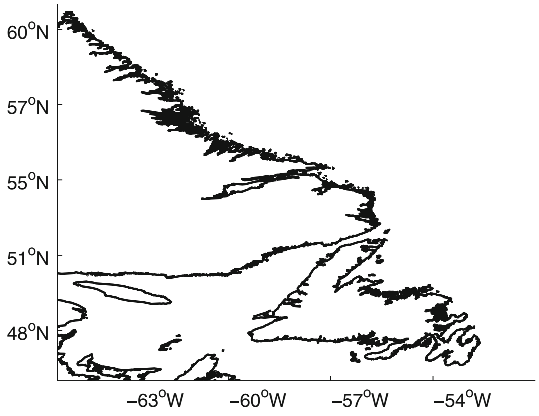


Fig. 1 Region of study is the Labrador Coast and Newfoundland

Table 1 Characteristics of the collected features used for designing M-RRNN

Notation	Features
x_1	Longitudinal position
x_2	Latitudinal position
x_3	Brightness temperature
x_4	Wind speed
x_5	Water vapour
x_6	Cloud water
x_7	Ice temperature
x_8	Sea surface temperature
x_9	Ice concentration
x_{10}	Ice thickness

The database used in this study covers the period of February 2, 2007, to February 20, 2007. For this portion of the year, the ice cover along the Labrador coast is bounded to the west and south by land and to the east by the Labrador current. The ice starts to appear along the Labrador coast in December and gradually becomes thicker through January and February. The ice cover contains a marginal ice zone composed of small ice floes near the open water, with the ice becoming thicker toward the land boundaries. In addition, sometimes, there may be some coastal polynyas between the consolidated ice region and the landfast ice.

3.2 AMSR-E data

To capture the radiation within the passive microwave range of electromagnetic spectrum, the AMSR-E sensor uses six different frequencies, 6.9, 10.7, 18.7, 23.8, 36.5, and 89 GHz. The footprint of each of these frequencies is approximately elliptical, with sizes ranging from 74 km × 43–6 km × 4 km. Swath data are used in this study to mitigate the effects of uncertainties that can arise when data are averaged or resampled. Furthermore, due to the land contamination associated with the sensor footprints, in this study, the information coming from pixels a distance of up to half of the sensor footprint from the land boundaries has been neglected. Brightness temperatures are used from both a low-frequency channel (6.9 GHz) and a high-frequency channel (36.5 GHz). The latitude, longitude, and brightness temperature values are entries x_1 to x_3 in the database (see Table 1).

3.3 MODIS data

The MODIS sensor measures radiation in the VIS/IR range of electromagnetic spectrum. To calculate the sea-ice thickness, the surface temperature calculated from the MODIS infrared channels (Hall et al. 2004) is used in a heat balance equation (Yu and Lindsay 2003). In this study, the heat balance equation uses the atmospheric variables from the Global Environmental Multiscale (GEM) model and the MOD29 ice surface temperature product prepared by the National Snow and Ice Data Center (Hall et al. 2007). The MODIS data are swath data at a 1-km resolution in which each pixel has been screened for cloud contamination. The observed data include surface temperatures from 243 to 271K. To reduce the rate of uncertainty of the collected database, nighttime images are used as those are not affected by uncertainties associated with the surface albedo and shortwave radiation (Wang et al. 2010). Figure 2 depicts a sample ice temperature image obtained by the MODIS sensor. The ice thickness from MODIS is entry x_{10} in the database (see Table 1).

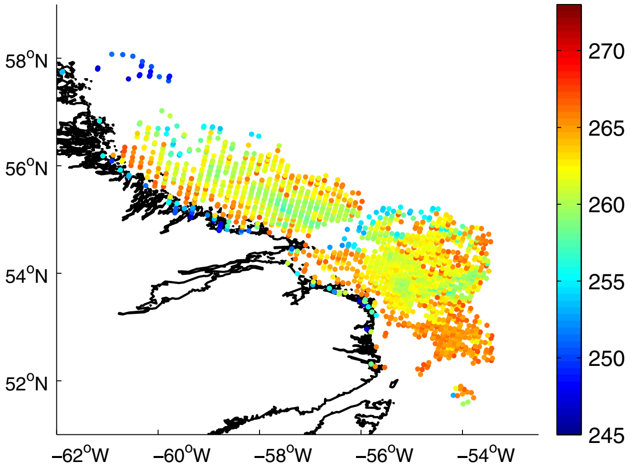


Fig. 2 Ice temperature from MODIS, January 24, 2007

3.4 Data from the forecasting system

Variables from an atmospheric weather forecasting model (the GEM model) and a coupled ice-ocean model are used in addition to the AMSR-E and MODIS data. The variables from these models are those that impact brightness temperature, such as surface temperature, windspeed, water vapour, and cloud liquid water. The dataset has been described in a previous study (Scott et al. 2012). The variables from the forecasting system used in the present study are listed as x_4 to x_9 in Table 1.

4 Methodology

In this section, the details of the architecture of modular ridge randomized neural network (M-RRNN) is presented. The considered M-RRNN is comprised of two parts: a distributor with a modified differential evolutionary algorithm at its heart and a modular estimation phase with ridge randomized neural network (RRNN) at each of the considered modules. In the first sub-section, the algorithmic description of the modified differential evolutionary algorithm is presented, and in the second sub-section, the steps required for the implementation of the RRNN and for creating a modular architecture are proposed.

4.1 Implementation of distributor: differential evolutionary algorithm

For the implementation of the clustering methodology, a modified version of differential evolutionary algorithm (DEA), called scale factor local search differential evolution (SFLSDE) (Neri and Tirronen 2009), is taken into account. This method has been proven to have a very robust performance and is not sensitive to the dimensionality of the problem. In previous theoretical and numerical investigations, it has been observed that the local search operators used at the heart of SFLSDE can result in a very fast convergence towards the optimum solution regardless of the characteristics of the landscape of the objective function (Neri and Tirronen 2009; Mozaffari et al. 2014). These local search operators are known as golden sectioning search (GSS) and hill-climbing (Neri and Tirronen 2009), which are used to update the scale factor parameter and crossover rate of the standard DEA. The salient asset of SFLSDE lies in its capability to allocate an independent scale factor to each of the potential solutions in the solution space. This enables SFLSDE to devise an independent searching strategy for each of the agents that in turn increases the diversity of the algorithm. One of the other salient assets of SFLSDE compared to other variants of DEA lies in its capability to update the scaling factor value in a self-organizing fashion. To the best knowledge of the authors, SFLSDE has not been applied to clustering so far. However, it has the following computational advantages that clearly indicate it may have high potential for the clustering task:

(a) It has been proven that SFLSDE has an efficient performance, independent from the dimensionality of the optimization problem. This is due to the insensitivity of the local searches to the scale of the solution landscape (Neri and Tirronen 2009).

(b) The algorithmic structure of SFLSDE not only results in a high diversity of the search over the solution space, but also enables it to neatly balance the exploration and exploitation capabilities to ensure the convergence to the optimum regions within the objective landscape (Neri and Tirronen 2009).

(c) One of the main features of SFLSDE lies in its capability to fix a major flaw associated with a large number of modified DEAs, namely the phenomenon of stagnation. Indeed, a large portion of proposed DEAs may become trapped in a local minimum during the optimization procedure. This is known as stagnation. However, diversified refreshing of the characteristics of genotypes participating in the optimization procedure helps SFLSDE to escape from local minima, which in turn impedes stagnation (Neri and Tirronen 2009).

(d) SFLSDE is indeed a memetic algorithm based on Lamarckian learning which is the integration of DEA and two local search operators (Neri and Tirronen 2009). However, despite most of the existing memetic methods, the local searches used at the heart of SFLSDE does not increase the computational complexity of the resulting architecture, as both hill climbing and golden sectioning search mechanisms have quite simple architectures.

The abovementioned remarks have motivated us to adopt SFLSDE to develop a clustering methodology, which will be used as a distributor in the considered modular ridge randomized neural network. By inspecting the literature of heuristic clustering, the authors realized that adding a general operator known as acceleration can effectively boost the performance of clustering metaheuristics (Chuang et al. 2011). Thus, in this study, the authors embed the acceleration strategy into the algorithmic structure of SFLSDE to make sure it has an acceptable clustering behavior. Let us call the resulting clustering method as SFLSDE-clust. The following steps are taken to implement the method:

Step 1: Set the controlling parameters of SFLSDE-*clust* according to those given in Table 2.

Step 2: Uniformly spread the position of S chromosomes [denoted by vectors $\mathbf{s}(1), \mathbf{s}(2), \ldots, \mathbf{s}(S)$] through the solution space. Each solution in this space represents the centroids of a set of clusters (different centroid points in a 2D space are shown in the first panel of Fig. 3). Assume that K clusters are going to be used for partitioning the collected information, and the destined information possesses d features. Then, each chromosome has $K \times d$ bits that should be optimized. In a previous work by the authors, it was observed that the optimum number of chromosomes for clustering is $10 \times K \times d$ (Chuang et al. 2011).

Table 2 Controlling parameters of SFLSDE$-$*clust* for the current simulations

Parameter	Notation	Value
Number of chromosomes	P_s	40
Minimum scale factor value	F_ℓ	0.1
Maximum scale factor value	F_u	0.9
Random number	τ_1	0.1
Random number	τ_2	0.1
Random number	τ_3	0.03
Random number	τ_4	0.07
Initial climbing value	h	0.6
Lower bound of GSS	a	-1
Upper bound of GSS	b	1
Crossover rate	CR	0.8
Golden value	Φ	1.6181
Golden search iterations	$\text{iter}_{\text{SFGSS}}$	10
Hill climbing iterations	$\text{iter}_{\text{SFHC}}$	10

Step 3: Prior to starting the clustering procedure, proceed with the acceleration process. In this way, select the one third of the chromosomes randomly and use the standard K-means clustering (Chuang et al. 2011) to update their position. Each chromosome will cluster the information in the database to minimize the Euclidean distance

$$\|\mathbf{x}_p - \mathbf{c}_j\| = \sqrt{\sum_{i=1}^{d} \left(x_{p,i} - c_{j,i}\right)^2},$$

where $\mathbf{c}_j$ is the centroid of the jth cluster $\mathcal{C}_j$, $c_{j,i}$ indicates the ith coordinate of the centroid $\mathbf{c}_j$, $x_{p,i}$ is the ith coordinate of the pth data in the database, and d is the dimensionality of the data points $\mathbf{x}_p$. The centroid of the jth cluster $\mathcal{C}_j$ is determined using the set of data points in cluster j according to

$$\mathbf{c}_j = \frac{1}{n_j} \sum_{\mathbf{x}_p \in \mathcal{C}_j} \mathbf{x}_p,$$

where n_j represents the number of data points in cluster $\mathcal{C}_j$.
Step 4: After the termination of acceleration phase, calculate the objective function of all chromosomes using the sum of intra-cluster distance, given as

$$J = J(\mathbf{c}_1, \ldots, \mathbf{c}_K) = \sum_{j=1}^{K} \sum_{\mathbf{x}_p \in \mathcal{C}_j} \|\mathbf{x}_p - \mathbf{c}_j\|. \tag{1}$$

The above objective function is minimized to find the $\mathbf{c}_j$ corresponding to optimal clustering and is referred to as the fitness.

Step 5: For each chromosome select 5 random numbers, u_1, u_2, u_3, u_4, and u_5, from a uniform distribution (Neri and Tirronen 2009).
Step 6: For each chromosome $\mathbf{s}(i)$, randomly select three individuals, i.e. $\mathbf{s}_1(i)$, $\mathbf{s}_2(i)$ and $\mathbf{s}_3(i)$, from the existing population pool.
Step 7: For chromosome $\mathbf{s}(i)$, if the selected solution has the best fitness value and also u_5 is less than a given threshold τ_3, proceed with golden sectioning search (GSS) strategy to update the scale factor of the agent $F(i)$ (the steps required for GSS is given in Appendix 1). Archive the resulting off-spring as $\mathbf{u}(i)$.
Step 8: For chromosome $\mathbf{s}(i)$, if the selected solution has the best fitness value and also u_5 is greater than the threshold τ_3 and at the same time less than the threshold τ_4, proceed with the hill-climbing search to update the scale factor of the agent, $F(i)$ (the steps required for GSS is given in Appendix 2). Archive the resulting off-spring as $\mathbf{u}(i)$.
Step 9: For the chromosome $\mathbf{s}(i)$, if u_5 is greater than the threshold τ_4, proceed with the conventional evolutionary steps of standard DEA (the steps required for GSS is given in Appendix 3). Archive the resulting off-spring as $\mathbf{u}(i)$.
Step 10: Calculate the fitness of the calculated solution. If the resulting solution $\mathbf{u}(i)$ has a higher fitness than the parent $\mathbf{s}(i)$, then replace the old solution with the new one.
Step 11: Check the stopping criteria. If the stopping criteria are satisfied terminate the procedure; otherwise, return to Step 2.

A schematic illustration of the procedures taken by SFLSDE-clust to partition the information to a number of clusters is shown in Fig. 3. At this point, K partitions have been created, which can be used to develop a modular estimator with K modules.

4.2 Implementation of the modular architecture

As it was pointed out previously, a modular architecture comprises of K modules where each of them has an estimator at their heart. In this sub-section, we first concentrate on explaining the mathematical structure of each estimator, i.e. RRNN. Thereafter, we explain how those independent modules should be put together to form the modular framework, i.e. M-RRNN.

In the proposed modular frame, random based neural networks are used for function approximation. Through theoretical studies, it has been proven that a multi-layer neural network with a single hidden layer including bounded and nonconstant activation functions can serve as a universal approximator (Hornik 1991). In parallel, in the same year, Park and Sandberg (1991) demonstrated that radial-basis networks with the same width for all radial-basis neurons or different widths for different RBF neurons in the network can also serve as universal approximators. Such findings

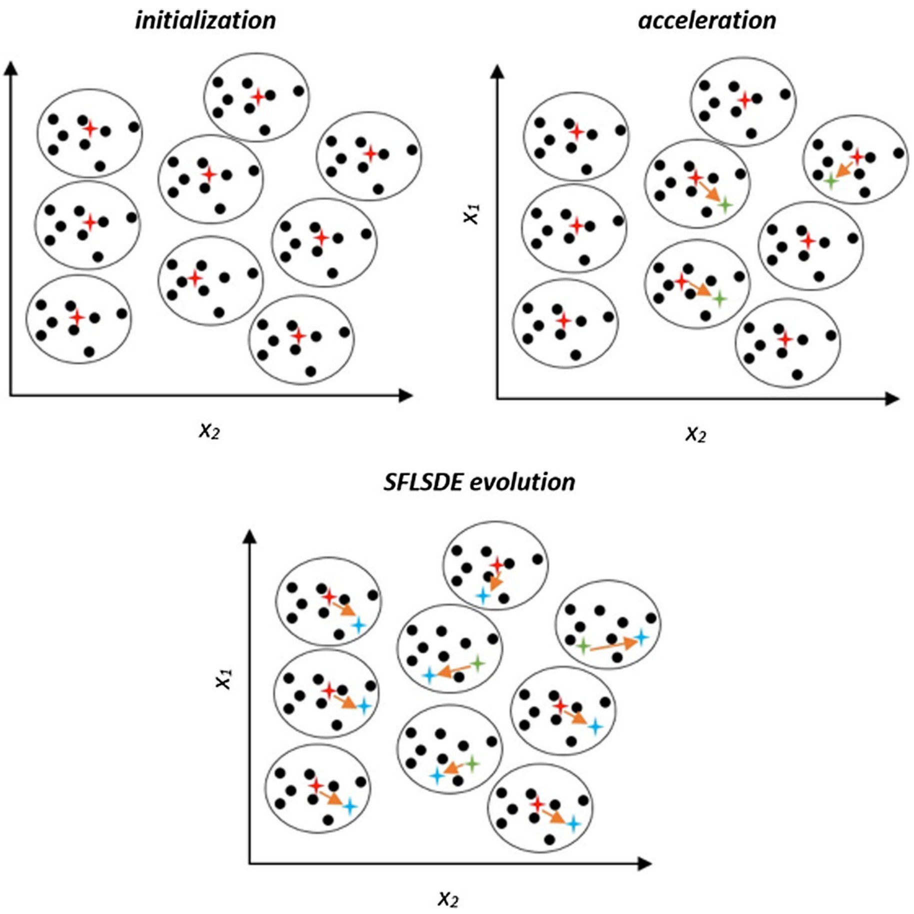


Fig. 3 Schematic illustration of the proposed clustering procedure in 2D space: (1) initialization of centroids in the information space, (2) selection and updating of one third of the clusters, (3) proceed with the operators of SFLSDE to update all of the clusters

have enabled the researchers of neural computation society to search for much flexible learning schemes which reduce the complexity of optimization process required for training the network's computational units (neurons). In line with such activities, over the past decades, a comprehensive investigation has been carried out which clearly demonstrated the potentials of random based learning systems for designing feed-forward neural networks (Schmidt et al. 1992), radial basis neural networks (Broomhead and Lowe 1988; Lowe 1989), and functional link nets (Pao et al. 1994). Based on the promising reports on the computational power of feed-forward randomized neural networks (RNN) (Schmidt et al. 1992), which is in a good agreement with the authors own experiments, this network is used at the heart of the proposed modular network to form the approximator.

It should be mentioned that, nowadays, a large number of research groups are trying to improve the performance of RNNs and modified versions of random neural networks are being proposed. In this context, one can find different architectures of RNNs in the literature which are known by specific names. Among the existing vairants of RNNs methods such as extreme learning machines (ELMs) (Huang et al. 2006), random vector functional link nets (RVFLNs) (Zhang and Suganthan 2015), random kitchen sinks (RKSs) (Rahimi and Recht 2007), fast foods (Le et al. 2013), convex network (Huang et al. 2013), no-prop algorithm (Widrow et al. 2013), liquid state machines (LSMs) (Yamazaki and Tanaka 2007), echo state networks (ESNs) (Rodan and Tiňo 2011), and reservior computing machines (LukošEvičIus and Jaeger 2009), have relatively found their reputation in commputational intelligence society.

Proposed in Wu and Moody (1996), ridge neural network is a modified version of standard RNN which tries to tame the numerical difficulties associated with the analytical solving methodology used in analytically trained least square based neural networks. To be more precise, the modification is made by applying the concept of Tikhonov regularization instead of simple least square solution to tune the synaptic weights of RNN (Burger and Neubauer 2003). Assume that the collected database has n training samples $\mathfrak{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$ in which $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,d})^{\mathrm{T}}$ represents the d-dimensional input vector and y_i represents the response value for the ith observation, respectively. Assume that neural network has N hidden nodes in its architecture. Then, the mathematical formulation below is used to create a map between the input vectors, $\mathbf{x}_i$'s, and the target values, $f(\mathbf{x}_i)$

$$\sum_{j=1}^{N} w_j g\left(\boldsymbol{\alpha}_j^{\mathrm{T}}\mathbf{x}_i + b_j\right) = f(\mathbf{x}_i), \quad i = 1, \ldots, n, \tag{2}$$

where $\boldsymbol{\alpha}_j = (\alpha_{j,1}, \ldots, \alpha_{j,d})^{\mathrm{T}}$ are the synaptic weight vectors connecting the input nodes to the jth hidden node, w_j indicates the weight connecting the jth hidden node to the output nodes, and g is a continuous activation function, which is the sigmoid function in this paper (Pao et al. 1994), i.e.

$$g(x) = \frac{\mathrm{e}^x}{1+\mathrm{e}^x} \quad \text{for } x \in \mathbb{R}.$$

Let

$$\mathbf{H} = \begin{pmatrix} g(\boldsymbol{\alpha}_1^{\mathrm{T}}\mathbf{x}_1 + b_1) & \ldots & g(\boldsymbol{\alpha}_N^{\mathrm{T}}\mathbf{x}_1 + b_N) \\ \vdots & \vdots & \vdots \\ g(\boldsymbol{\alpha}_1^{\mathrm{T}}\mathbf{x}_n + b_1) & \ldots & g(\boldsymbol{\alpha}_N^{\mathrm{T}}\mathbf{x}_n + b_N) \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix},$$

$$\mathbf{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_N \end{pmatrix}.$$

To estimate the function f defined in Eq. (2), the RNN algorithm discussed in Schmidt et al. (1992) allows the user to choose $\boldsymbol{\alpha}_j$'s and b_j arbitrarily at random and apply the least square method to estimate the hidden output weight vector $\mathbf{w}$, that is

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{H}\mathbf{w}\|_2^2,$$

where $\|\mathbf{a}\|_2$ represents the Euclidean norm of an arbitrary vector $\mathbf{a}$. See also Wu and Moody (1996). It is known that if the matrix $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ is invertible the least square solution is

$$\widehat{\mathbf{w}} = \left(\mathbf{H}^{\mathrm{T}}\mathbf{H}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{y}.$$

In practice though, often the condition value of the matrix $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ is close to zero and, therefore, the solution $\widehat{\mathbf{w}}$ is not stable. To resolve this problem, Schmidt et al. (1992) suggested using the Moore–Penrose generalized inverse. A better and more numerically stable solution is obtained by a penalized least square problem known as Tikhonov regularization or the ridge regression. Therefore, we do the following optimization (Hastie et al. 2009):

$$\min_{\mathbf{w},\lambda_2} \left\{\|\mathbf{y} - \mathbf{H}\mathbf{w})\|_2^2 + \lambda_2\|\mathbf{w}\|_2^2\right\}.$$

The solution to this optimization problem is given by the ridge regression estimate

$$\widehat{\mathbf{w}} = (\mathbf{H}^{\mathrm{T}}\mathbf{H} + \lambda_2\mathbf{I})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{y}, \tag{3}$$

where $\lambda_2 \geq 0$ is the ridge or Tikhonov regularization parameter. In the present study λ_2 was obtained by means of Bayesian information criterion (BIC) (Hastie et al. 2009).

At this point, the steps required for the implementation of an estimation module, i.e. RRNN, is completed. Now it should be discussed as to how those independent modules are concatenated to form the final M-RRNN.

As mentioned earlier, the number of independent modules at the heart of M-RRNN is directly related to the number of clusters formed by the SFLSDE-clust distributor. Let us assume that the observed input vectors are given by $\mathcal{C} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ and is clustered into K separated partitions $\mathcal{C}_1, \ldots, \mathcal{C}_K$. For $j = 1, \ldots, K$ and $\ell = 1, \ldots, d$, let

$$a_{j,\ell} = \min\{x_{j,\ell} \mid \mathbf{x}_j = (x_{j,1}, \ldots, x_{j,\ell}, \ldots, x_{j,d}) \in \mathcal{C}_j\},$$
$$b_{j,\ell} = \max\{x_{j,\ell} \mid \mathbf{x}_j = (x_{j,1}, \ldots, x_{j,\ell}, \ldots, x_{j,d}) \in \mathcal{C}_j\}.$$

Then, for each cluster $j = 1, \ldots, K$

$$\mathcal{C}_j \subset [a_{j,1}, b_{j,1}] \times [a_{j,2}, b_{j,2}] \times \cdots \times [a_{j,d}, b_{j,d}],$$
$$\mathcal{C} = \bigcup_{j=1}^{K} \mathcal{C}_j.$$

Then, all of the training samples in cluster j are used to train jth module of M-RRNN. It can be clearly understood that by assigning an independent module to each of the above partitions, the following can be assured: (1) whole $\mathcal{C}$ is covered by the resulting M-RRNN, and at the same time; (2) a precise estimator is developed for each of the separated clusters, as the information in each cluster shares relatively similar characteristics. Besides, by decomposing the complexity of the dataset to a number of separated clusters with independent characteristics, it can be ensured that simpler estimators with smaller sizes of hidden layers can be used for estimating each cluster, and also the accuracy of estimation is increased.

4.3 Architecture of M-RRNN

By integrating the distributor with the modular estimator, the architecture of M-RRNN is formed. The architecture of M-RRNN is presented in Fig. 4.

5 Experimental setup

In this section, the steps required for setting the parameters for the simulations are outlined. As discussed, the experiments are performed in two different stages. As the proposed modular scheme has two different sections, i.e. distributor and estimator, different sorts of rival techniques should be considered to test the efficacy of the proposed technique.

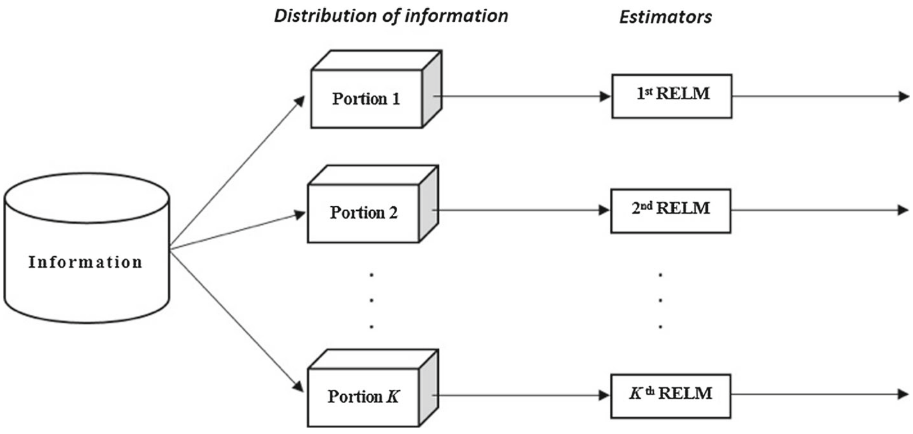


Fig. 4 Architecture of the designed modular ridge extreme learning machine

5.1 Distributor performance

To check the performance of distributor, which is based on a differential evolutionary algorithm, several well-known nature-inspired clustering methods, genetic algorithm (GA) (Sheikh et al. 2008), artificial bee colony (ABC) (Karaboga and Ozturk 2011), chaotic particle swarm optimization (CPSO) (Chuang et al. 2011), and firefly algorithm (FA) (Senthilnath et al. 2011), are taken into account. It is worth mentioning that all of the considered clustering techniques are equipped with an acceleration phase that can improve the performance of any given stochastic/metaheuristic clustering algorithm (Chuang et al. 2011). To tune GA, the number of chromosomes of 40, the crossover probability (P_c) of 0.8, the mutation probability (P_m) of 0.02, and the number of elite chromosomes (e) of 1, are considered. Besides, the arithmetic graphical search (AGS), tournament selection and simulated binary crossover (SBX) operators are adopted from the literature to form the algorithmic structure of GA. For optimal implementation of ABC, the number of onlooker and employed bees of 20 and the limit number of 10 are taken into account. The limit size of 10 implies that the first bee that fails to update its position after 10 trials is sent to scout bee search phase to update its solution vector. For CPSO, the particle size of 40, inertia weight of 0.9, and cognitive and social coefficients of 1.4 are taken into account. To prepare FA, the number of fireflies of 40, the maximum attraction ($\beta_{\max}$) of 1, and the absorption rate (Υ) of 1 are selected for the sake of optimization. All of the above mentioned parameters are set based on the recommendations given in the seminal cited papers as well as the author's own assessments. Besides, all of the rival algorithms start the optimization using the same population seeding (distribution) to impede obtaining any biased results. All of the considered clustering techniques perform the optimization for 10,000 function evaluations, and each algorithm is executed for ten independent runs. Well-known statistical measures, i.e. standard deviation (std.), accuracy (mean), best (min), and worst (max), are also taken into account, which are defined as follows:

$$\text{Mean fitness} = \frac{1}{10}\sum_{i=1}^{10} J_i$$

$$\text{Best fitness} = \min\{J_i | i = 1, \ldots, 10\}$$

$$\text{Worst fitness} = \max\{J_i | i = 1, \ldots, 10\}$$

$$\text{Robustness} = \frac{1}{9}\sum_{i=1}^{10}(J_i - \text{Mean fitness})^2$$

It is also worth pointing out that the clustering process is conducted using four different numbers of clusters ($K = 5, 6, 7$ and 8) to find the optimal number of clusters.

It is expected that either the ice surface temperature or the ice thickness can be used together with the spatial information (latitude and longitude values) to optimally partition the database. In this way, two different sets of clustering experiments are carried out to evaluate which method leads to the best results. The two sets of data used for the two clustering scenarios are

$$\text{Experiment 1 : features } x_1, x_2, x_7$$

$$\text{Experiment 2 : features } x_1, x_2, x_{10},$$

which correspond to latitude/longitude/ice temperature and latitude/longitude/ice thickness, respectively.

5.2 Performance analysis

To evaluate both the accuracy and robustness of the estimation part of the proposed M-RRNN, several well-known variants of neural networks, back-propagation neural network (BP-NN) with steepest descent optimization algorithm (Mozaffari and Fathi 2012), optimally pruned RNN (OPRNN), randomized neural network (RNN) (Schmidt et al. 1992), and multi-layer feed-forward neural network

Table 3 Statistical characteristics of the collected databases for low- and high-frequency databases

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Low frequency										
Mean	300.9	57.9	228.2	8.91	1.15	2E−4	251.8	272.0	0.75	0.35
Min	295.4	53.3	161.7	2.23	0.88	−1E−4	244.0	271.1	0.001	6E−4
Max	305.0	66.0	252.6	16.0	1.73	1E−3	267.1	274.6	0.999	1.26
Std.	2.21	3.65	23.49	3.18	0.16	4E−4	4.52	0.86	0.322	0.261
High frequency										
Mean	300.8	57.3	239.1	8.29	1.14	1E−4	249.8	271.7	0.862	0.431
Min	295.1	53.3	203.2	1.42	0.85	−4E−4	242.1	271.1	0.002	7E−4
Max	304.8	66.1	249.7	14.5	1.57	0.001	263.0	274.8	1.000	1.924
Std.	2.27	3.47	7.24	2.83	0.16	3E−4	3.42	0.64	0.234	0.278

For the low-frequency database there are 14,639 data pairs, while for the high-frequency database there are 17,162 data pairs

with interior point optimization (MLFF-IPO) (Raja and Samar 2014), are replaced with RRNN modules of the modular architecture. All of the identification modules have 10 neurons in their hidden layer. For the steepest descent optimization method, a learning rate of 0.1 is used. The iterative optimizers perform the learning for 100 iterations. It should be pointed out that OPRNN comprises of a standard RNN in which the most influential neurons are retained in the architecture through multiresponse sparse regression neuron pruning. Also, the pruning procedure takes advantage of leave-one-out validation criterion to ensure the optimal selection of active neurons (Hastie et al. 2009).

The database used for the analysis has eight features. The considered features are listed in Table 1. As it was mentioned, the considered databases contain ice thickness from the MODIS sensor, brightness temperatures from the ASMR-E sensor, and geophysical variables from forecast models. The database covers the period from 2 February to 20 February and has 14,639 and 17,162 temporal data-pairs for low (6.9 GHz) and high (36.5 GHz) frequency AMSR-E channels, respectively. In the rest of this paper, these two datasets are referred to as the low- and high-frequency databases. The collected databases have been normalized within the range of unity for increasing the efficiency of the computation. Table 3 lists the statistical characteristics of the gathered features for both low and high frequencies.

5.3 Verification of method complexity

It is necessary to verify that the complexity of the final modular network is not greater than the complexity of non-modular systems. In this regard, after finding a modular network with K modules, single models of each of the above identifiers with $K \times 10$ neurons are taken into account for estimation of the sea-ice thickness. This is mainly due to the fact that the complexity of the resulting networks in terms of O notation strictly depends on the number of hidden nodes (which is 10 in this study).

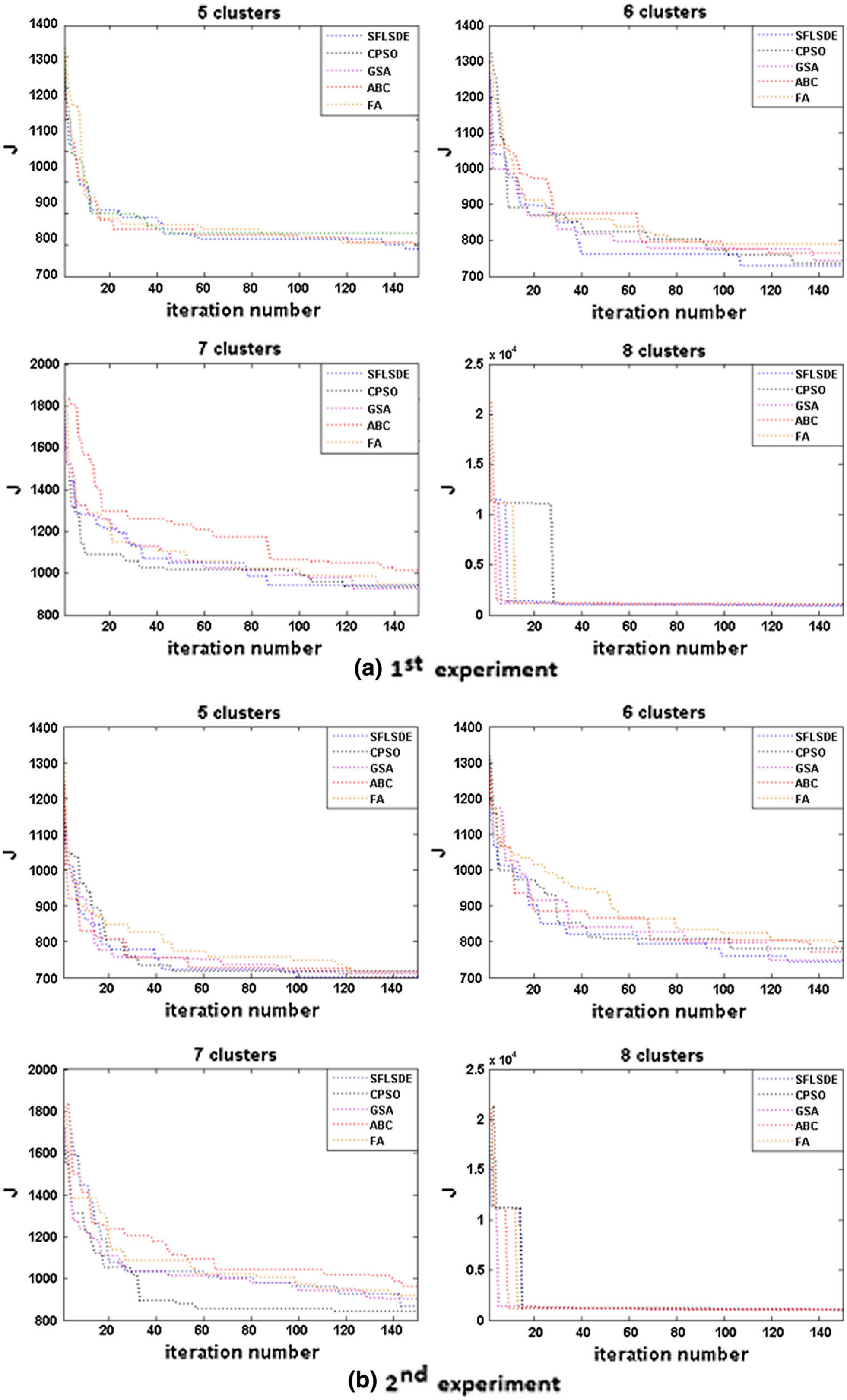
Finally, it is necessary to mention that all of the simulations are carried out on the Matlab software with Microsoft Windows 7 operation system on a PC with a Pentium IV, Intel core i7 CPU, and 4 GBs RAM.

6 Results and discussion

In this section, the results from several experiments are reported to evaluate the performance of the proposed intelligent approach. In the first stage of the experiments, the potential of the proposed method is tested for efficient partitioning of the dataset. Thereafter, the proposed estimation tool together with the other rival estimators is used to develop a map between the input features and the sea-ice thickness. Some mathematical formulations are also presented to calculate the complexity of the resulting modular framework and also to evaluate the estimation error of the modular architecture based on the errors of each of the independent modules.

Figures 5 and 6 indicate the real-time evolution of the rival clustering methods for partitioning the low- and high-frequency information, using different numbers of clusters. The clustering scenarios differ in that one is conducted using the ice surface temperature as the third features, while the other uses the ice thickness.

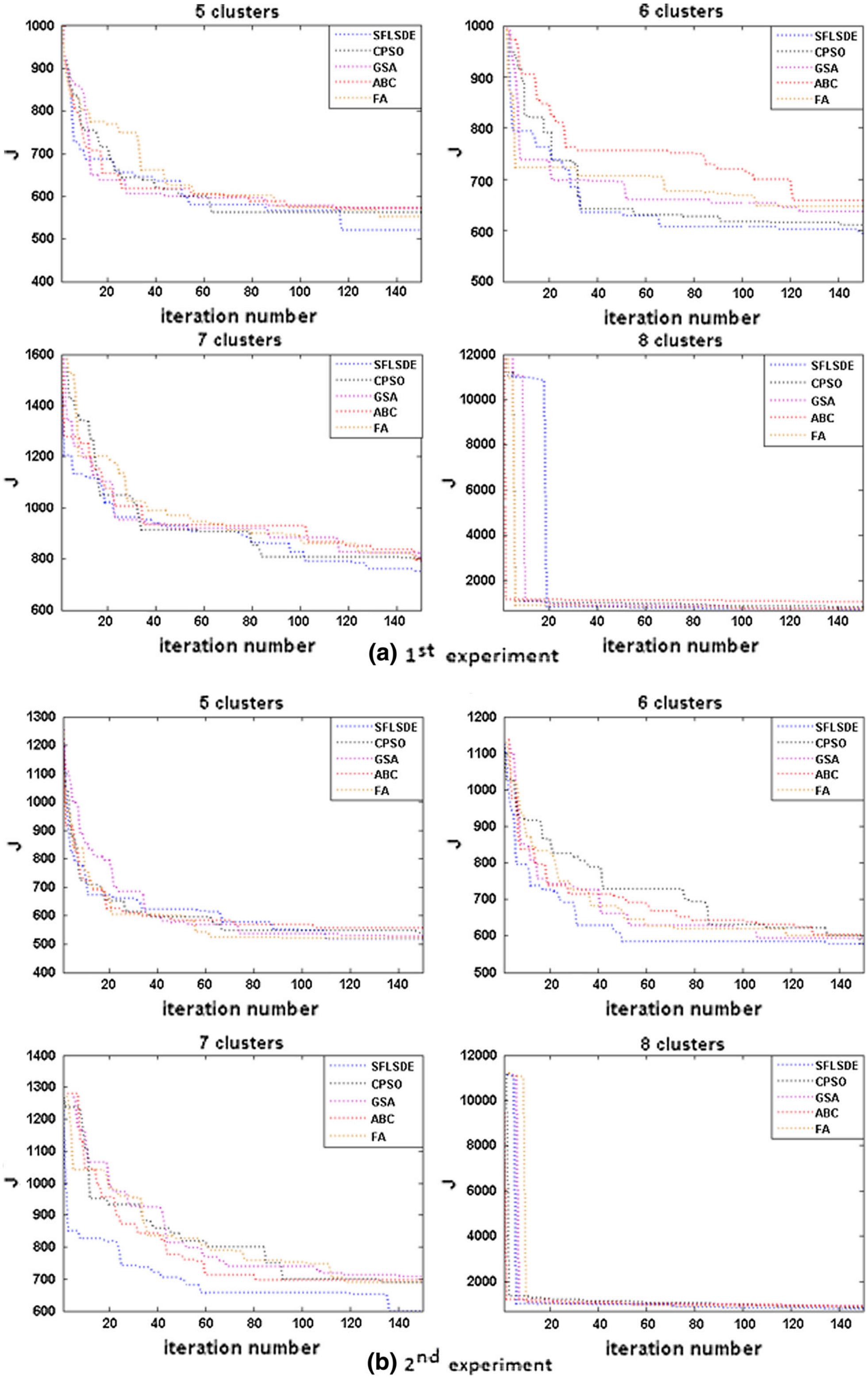
Figure 5 indicates that for both of the clustering experiments, increasing the number of clusters decreases the convergence speed of the metaheuristic algorithms. This is logical, as increasing the number of clusters increases the dimensionality of the clustering problem and makes the optimization landscape more intricate. Looking more closely at the results, it can be seen that for both of the experiments of the low-frequency scenario, SLFSDE-clust and CPSO have the fastest convergence speed. Also, the simulation results indicate that for most of the clustering scenarios, ABC cannot



(a) 1^{st} experiment

(b) 2^{nd} experiment

Fig. 5 Mean real-time evolution of the considered clustering techniques for low frequency database



(a) 1st experiment

(b) 2nd experiment

Fig. 6 Mean real-time evolution of the considered clustering techniques for high frequency database

Table 4 Comparison of the fitness values for the rival clustering techniques for low-frequency database

	SFLSDE	CPSO	ABC	FA	GSA	K-means
1st experiment						
Mean						
5 clusters	686.393	701.983	705.289	737.317	702.405	813.072
6 clusters	730.888	739.334	765.138	790.889	743.352	823.643
7 clusters	924.837	941.147	1016.831	949.601	942.647	1058.192
8 clusters	875.665	1021.737	1022.921	974.824	1031.121	1048.114
Best						
5 clusters	685.129	701.203	704.399	736.795	701.692	812.374
6 clusters	728.635	736.593	764.025	788.238	742.897	821.283
7 clusters	923.882	940.571	1014.354	946.778	941.392	1051.719
8 clusters	874.014	1019.422	1019.389	972.588	1028.861	1044.273
Worst						
5 clusters	688.923	703.023	706.625	738.102	703.475	814.121
6 clusters	733.142	741.391	766.622	792.663	744.262	825.217
7 clusters	925.475	942.012	1018.384	951.483	943.485	1054.312
8 clusters	878.142	1023.153	1025.269	976.316	1032.616	1049.417
Std.						
5 clusters	0.959	0.972	1.219	1.715	0.976	1.541
6 clusters	1.468	1.564	1.388	1.728	0.704	1.154
7 clusters	0.872	0.789	2.207	2.577	1.146	2.802
8 clusters	2.261	2.043	3.220	2.041	2.056	2.572
2nd experiment						
Mean						
5 clusters	702.363	718.489	716.492	704.372	712.950	817.741
6 clusters	743.245	772.333	761.568	780.800	747.848	826.799
7 clusters	868.277	845.697	967.730	919.404	907.398	1058.021
8 clusters	964.811	1005.625	1051.512	1010.815	1019.425	1052.736
Best						
5 clusters	701.623	717.681	714.325	702.009	710.814	816.283
6 clusters	742.008	771.026	760.115	779.744	746.885	825.374
7 clusters	864.813	840.182	962.264	916.599	899.839	1045.495
8 clusters	962.558	1004.49	1050.296	1009.094	1017.692	1051.495
Worst						
5 clusters	703.475	719.029	717.938	705.948	714.375	819.928
6 clusters	745.102	774.294	763.748	782.384	749.293	828.938
7 clusters	873.475	849.374	971.374	921.274	912.438	1066.384
8 clusters	968.192	1006.283	1053.203	1013.246	1021.873	1054.385
Std.						
5 clusters	1.014	0.738	1.978	2.157	1.950	1.996
6 clusters	1.694	1.790	1.989	1.446	1.318	1.952
7 clusters	4.744	5.034	4.989	2.560	6.901	11.441
8 clusters	3.085	0.982	1.592	2.274	2.290	1.582

compete with the other rival techniques in terms of the accuracy of the final solution as well as the convergence speed. By inspecting the real-time evolution results of high-frequency scenario, presented in Fig. 6, it can be seen that similar performance is observed. The only interesting observation refers to the obvious superiority of SFLSDE-clust for $K = 5$ and 7 (first experiment), and $K = 7$ (second experiment). Taking into account that the number of datapairs of high-frequency database is larger than that of low-frequency database, it can be inferred that by increasing the complexity of the informa-

Table 5 Comparison of the fitness values for the rival clustering techniques for high-frequency database

	SFLSDE	CPSO	ABC	FA	GSA	K-means
1st experiment						
Mean						
5 clusters	521.307	573.047	564.530	553.165	576.274	723.110
6 clusters	596.196	611.849	663.606	647.349	638.230	805.126
7 clusters	751.953	801.822	806.098	800.826	793.113	823.057
8 clusters	710.680	758.257	1068.621	765.606	815.858	870.477
Best						
5 clusters	519.859	571.605	562.736	551.493	572.447	721.485
6 clusters	593.202	610.894	658.623	645.797	636.195	802.948
7 clusters	749.655	799.469	804.154	798.627	791.193	821.293
8 clusters	708.305	756.145	1062.731	764.396	814.934	863.273
Worst						
5 clusters	523.483	575.241	567.236	554.285	578.826	724.194
6 clusters	598.192	613.283	666.928	648.384	641.283	808.395
7 clusters	753.485	803.392	807.395	802.293	794.394	824.234
8 clusters	714.243	761.426	1077.316	767.423	817.245	881.283
Std.						
5 clusters	1.983	1.974	2.464	1.526	3.493	1.483
6 clusters	2.733	1.308	4.548	1.417	2.786	2.983
7 clusters	2.097	2.148	1.775	2.008	1.753	1.610
8 clusters	3.252	2.892	7.988	1.658	1.265	9.864
2nd experiment						
Mean						
5 clusters	519.763	527.548	557.784	541.608	522.848	530.914
6 clusters	577.025	602.906	604.176	595.642	580.677	595.988
7 clusters	600.462	697.795	691.430	690.477	708.564	773.216
8 clusters	798.613	864.862	916.250	848.113	893.427	934.927
Best						
5 clusters	518.757	526.398	556.718	539.193	520.695	527.374
6 clusters	575.091	600.826	602.365	592.878	578.419	594.384
7 clusters	599.142	696.937	689.791	689.447	706.707	767.878
8 clusters	794.624	862.581	913.051	845.758	892.194	931.273
Worst						
5 clusters	521.273	529.273	559.384	543.218	524.284	533.274
6 clusters	579.928	604.294	605.384	597.485	582.183	598.394
7 clusters	602.443	699.082	693.889	692.023	709.803	771.225
8 clusters	801.273	866.384	918.384	849.684	895.278	937.364
Std.						
5 clusters	1.378	1.574	1.460	2.204	1.965	3.231
6 clusters	2.649	1.899	1.653	2.523	2.061	2.196
7 clusters	1.808	1.174	2.244	1.410	1.695	2.310
8 clusters	3.641	2.083	2.921	2.150	1.689	3.336

tion to be partitioned, the superiority of SFLSDE, at least in terms of convergence speed, becomes much more obvious.

Tables 4 and 5 list the details of the statistical results for both experiments of low-frequency and high-frequency databases. The presented results indicate that by considering x_7 (ice temperature) as the third clustering feature, the performance of the clustering methods for partitioning both the low-frequency and high-frequency database is slightly improved. However, as it was mentioned, the best clustering should satisfy some other conditions. The std. values indicate

Table 6 Computational time for the rival clustering approaches for low-frequency database (s)

	SFLSDE	CPSO	ABC	FA	GSA
1st experiment					
5 clusters	7.663	7.362	8.102	7.711	7.585
6 clusters	8.221	8.018	8.625	8.322	8.212
7 clusters	9.256	9.008	9.412	9.302	9.121
8 clusters	10.434	10.123	10.728	10.503	10.219
2nd experiment					
5 clusters	6.304	6.115	7.894	7.703	7.602
6 clusters	8.211	8.003	8.749	9.373	8.414
7 clusters	9.201	9.013	10.793	10.512	9.803
8 clusters	12.666	12.293	13.237	12.831	12.121

Table 7 Computational time for the rival clustering approaches for high-frequency database (s)

	SFLSDE	CPSO	ABC	FA	GSA
1st experiment					
5 clusters	8.231	8.113	8.332	8.214	8.523
6 clusters	9.902	9.341	9.901	9.763	9.182
7 clusters	11.473	10.212	12.663	12.411	9.839
8 clusters	14.735	13.426	15.902	15.464	13.273
2nd experiment					
5 clusters	9.452	9.293	9.873	9.637	9.557
6 clusters	11.666	10.912	12.384	11.374	11.102
7 clusters	15.089	14.452	16.212	15.503	14.857
8 clusters	17.557	17.283	17.839	17.583	17.364

that the computational robustness of the rival methods is relatively similar. However, in most of the clustering scenarios, the robustness of ABC is less than the other methods, as its std. value is larger than the others. By checking the results of K-means clustering approach, it can be easily inferred that for all of the clustering scenarios, the heuristic clustering methods show superior results. Also, the results of the conducted simulations indicate that the performance of GSA and CPSO are relatively the same for most of the clustering scenarios. However, none of them can beat SFLSDE-clust algorithm for the considered clustering scenarios. All in all, the convergence and performance evaluation results indicate that SFLSDE-clust can be considered as a very powerful information partitioning scheme and can be used as the distributor of M-RRNN.

In addition to all of the above experiments, the authors would like to test the computational complexity of the considered rival clustering techniques. This will uncover whether the better performance of SFLSDE-clust is obtained at the cost of a higher computational burden. Tables 6 and 7 list the computational time of the rival clustering methods for both of the experiments on low-frequency and high-frequency datasets, respectively. The obtained results clearly demonstrate that the algorithmic complexity of SFLSDE is close to CPSO and GSA. Indeed, the obtained results indicate that the complexity of ABC is higher than the other rival techniques which implies that it has a much more complex algorithmic structure. The presented results of the computational complexity indicates that SFLSDE-clust not only has a very good performance, but also requires an acceptable rate of computational complexity to carry out the clustering task.

It is also important to find out the effect of using x_7 and x_{10} features on the partitioning of the considered region. To this aim, the clusters formed for low-frequency database using each of the mentioned clusters are indicated in Fig. 7. The presented results clearly indicate that the final partitions are quite different when considering either x_7 or x_{10} as the third feature. This graphical illustration clearly proves the importance of using the two experiments for extracting the best clustering scenario.

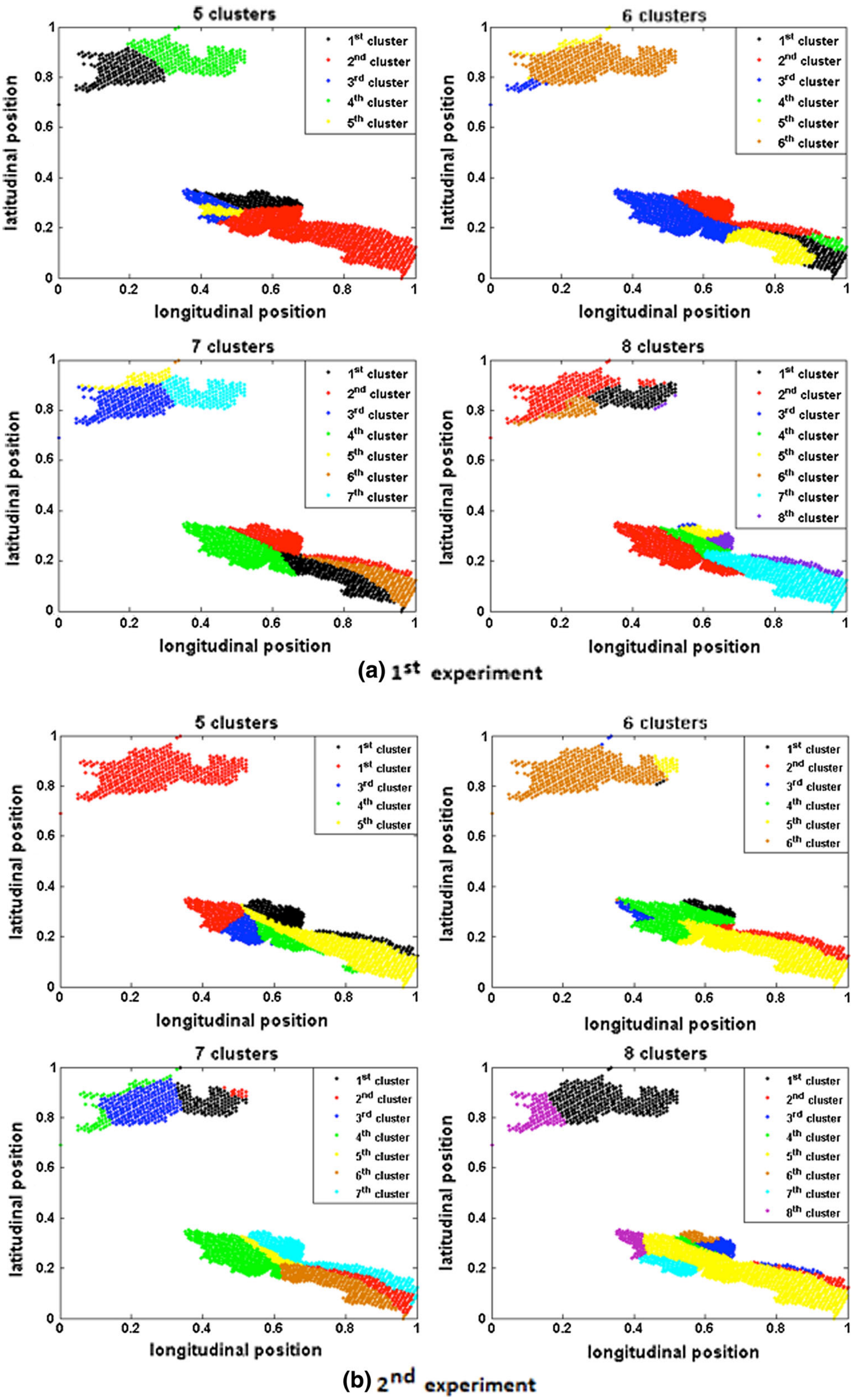
As mentioned earlieer, some additional performance evaluation metrics are required to realize which clustering scenario can better partition the data into a set of clusters. Table 8 lists the results of clustering for the two experiments on low-frequency dataset. By checking the reported values, it can be seen that $K = 7$ for the first experiment leads to the best results in terms of the considered performance evaluation indices. Table 9 lists the results of information clustering for high-frequency dataset. It can be seen that, for this case, the best performance is achieved for $K = 7$, considering x_{10} as the third clustering feature. Interestingly, for both of the datasets, the best clustering results are achieved when the number of clusters are equal to 7. The details of the formed centroids for the considered clusters of both of the experiments are given in Tables 10 and 11 for low-frequency and high-frequency datasets, respectively. The details of the centroid positions for $K = 7$ are shown in bold font.

Tables 12 and 13 list the number of data handled by each of the clusters for both of the experiments on low-frequency and high-frequency databases, respectively. The results disclose the sharing of information among clusters, and also indicates the number of data pairs used by each module.

By having the information regarding the distribution of the datapairs for each module, it is possible to calculate the distribution of computational complexity of the resulting modular structure as follows:

$$\text{Complexity} = \frac{n_j}{n} N_j \quad \text{for} \quad j = 1, \ldots, K, \tag{4}$$

where N_j indicates the number of hidden nodes of jth module, n is the number of datapairs, and n_j represents the number of data handled by jth module. As the number of



(a) 1st experiment

(b) 2nd experiment

Fig. 7 Comparison of clustering results for the two experiments for low-frequency database

Table 8 Comparison of the performance measures for each partitioning schemes for low-frequency data

Metrics	1st experiment				2nd experiment			
	5	6	7	8	5	6	7	8
Compactness	0.6915	2.7848	1.5225	1.9055	0.3595	2.4335	1.6293	2.1672
Separation	0.1219	0.1829	0.3113	0.2905	0.1894	0.2065	0.2944	0.3103
Ray–Turi’s index	15.9889	12.0367	10.2752	11.2583	7.1098	11.2607	10.8090	12.6560
Dunn’s index	0.06125	0.0584	0.0788	0.0546	0.0615	0.0683	0.0927	0.08221
DB index	13.7172	14.3314	16.4956	15.3746	14.1723	15.1724	16.2734	16.3543

Table 9 Comparison of the performance measures for each partitioning schemes for high-frequency data

Metrics	1st experiment				2nd experiment			
	5	6	7	8	5	6	7	8
Compactness	2.0665	3.3250	2.8849	3.2080	1.2794	1.0877	1.0115	1.2220
Separation	0.1034	0.1721	0.1992	0.2317	0.1132	0.1756	0.2050	0.3665
Ray–Turi’s index	9.8837	12.2712	10.2635	11.7522	9.9554	8.5567	7.7767	9.4602
Dunn’s index	0.0758	0.08118	0.0874	0.08323	0.0878	0.0711	0.0912	0.0665
DB index	12.2734	13.2817	14.2773	13.3235	14.2213	15.3441	16.2212	16.0032

hidden nodes at each module is fixed and equal to 10, the computational complexity of the modular frames corresponding to the selected distribution of data can be calculated as follows:

$$\text{1st scenario}: \frac{2940}{16670}, \frac{3290}{16670}, \frac{2030}{16670}, \frac{4390}{16670}, \frac{540}{16670}, \frac{1950}{16670}, \frac{1530}{16670}$$

$$\text{2nd scenario}: \frac{1830}{16670}, \frac{990}{16670}, \frac{2470}{16670}, \frac{4170}{16670}, \frac{3630}{16670}, \frac{1070}{16670}, \frac{54}{16670}.$$

The above information indicate that most of the designed modules have a relatively equal computational complexity. It seems that 1st, 2nd, and 4th modules of the first modular architecture, and 4th and 5th modules of the second architecture have a little bit more complexity as compared to the other modules. All in all, the results indicate that a neat balance is held between the modular architectures, which indicates that the proposed distributor does an acceptable job. Tables 14 and 15 list the MSE error of each of the modules for both first and second experiments on low-frequency and high-frequency databases. To calculate the total estimation error of each modular frame based on the MSE error of its modules, the following formulation is taken into account:

$$\mathrm{MSE}_{\mathrm{total}} = \frac{1}{n} \sum_{j=1}^{K} n_j \mathrm{MSE}_j. \qquad (5)$$

The total MSE of each of the modular frames is also included in Tables 14 and 15. Based on the reported values, it can be easily seen that the estimation error of modular frames with 7 modules is less than the other counterparts. This is in concurrence with the results of clustering which suggests the use of 7 modules to form the modular framework.

Figures 8 and 9 depict the correlations between the estimation of M-RRNN with 7 modules for both of the experiments on the low-frequency data, and those observed by MODIS sensor. It can be easily seen that, for both training and testing phases, the ice thickness estimates and the observed values are in a very good correlation. In particular, it can be seen that the correlation is higher when the low-frequency database is used, in agreement with earlier studies (Scott et al. 2014; Kaleschke et al. 2012) that indicate the potential of low-frequency microwave data for ice thickness estimation.

After selecting the most promising architecture for M-RRNN, that would be necessary to continue the experiments to find out whether the use of RRNN at the heart of the modular frames can afford the best results, and also to realize the computational advantage of the modular frames as compared to the standard estimators with the same number of hidden nodes (the same structural complexity). Tables 16 and 17 compare the accuracy and robustness of modular frames with different types of estimators at their heart. The results of the estimation indicate that the performance of modular frames with RRNN and OPRNN is better than the other rival techniques. However, the RRNN does a slightly better job. Moreover, it should be mentioned that the training procedure of OPRNN is a little bit more complex than RRNN as

Table 10 Details of the cluster centers obtained by SFLSDE for low-frequency data (un-normalized)

Clusters	1st	2nd	3rd	4th	5th	6th	7th	8th
1st experiment								
5 cluster								
x_1	301.63	302.36	301.76	301.58	301.22	–	–	–
x_2	56.29	56.53	56.40	55.37	54.96	–	–	–
x_7	250.34	251.77	252.36	253.67	252.72	–	–	–
6 cluster								
x_1	300.34	300.81	301.46	299.81	301.23	299.87	–	–
x_2	58.91	58.83	57.73	57.83	58.13	57.30	–	–
x_7	250.73	249.07	250.59	249.96	253.36	252.96	–	–
7 cluster								
x_1	299.31	300.70	301.79	299.62	300.61	298.62	299.25	–
x_2	59.44	60.49	59.27	59.78	59.10	58.45	57.64	–
x_7	256.48	250.13	251.23	256.41	255.47	255.35	251.08	–
8 cluster								
x_1	303.15	301.24	300.50	301.61	303.51	299.38	301.95	303.72
x_2	60.43	61.27	56.32	59.58	59.30	57.15	55.80	58.41
x_7	251.48	251.10	248.08	256.04	247.60	256.30	253.39	251.10
1st experiment								
5 cluster								
x_1	299.95	300.95	300.50	300.04	299.75	–	–	–
x_2	57.53	57.08	57.64	57.70	57.16	–	–	–
x_{10}	0.288	0.264	0.369	0.248	0.332	–	–	
6 cluster								
x_1	301.41	299.77	300.79	301.58	299.42	301.12	–	–
x_2	57.38	57.29	58.13	58.13	56.77	56.87	–	–
x_{10}	0.380	0.421	0.410	0.297	0.255	0.259	–	–
7 cluster								
x_1	300.85	299.33	302.48	302.57	302.12	300.96	300.11	–
x_2	56.43	56.42	57.59	58.75	59.61	59.24	57.49	–
x_{10}	0.320	0.238	0.150	0.371	0.422	0.368	0.096	–
8 cluster								
x_1	298.33	298.75	299.55	301.72	298.09	301.20	300.18	301.50
x_2	55.21	58.31	56.67	61.16	56.70	57.39	60.88	57.60
x_{10}	0.744	0.520	0.397	0.512	0.112	0.495	0.512	0.737

it includes both Lasso and ridge regression techniques. So, it can be inferred that the use of RRNN at the heart of M-RRNN can afford the best results. Moreover, the robustness results indicate that all of the considered frames have an acceptable robustness. This in turn implies that using modular architecture can improve the robustness of the estimation.

The final experiment intends to evaluate the efficacy of the proposed modular architecture in comparison with a number of rival architectures with the same structural complexity. Tables 18 and 19 list the results of the estimation. By checking the obtained results, it can be observed that the accuracy of the modular frame is higher than the other standard rival estimation approaches. It can be seen that the modular estimator has a significantly better performance. Moreover, by comparing the estimation error of the standard estimators with their modular architectures presented in the previous experiment, it can be seen that considering a modular frame for each of the considered estimators can drastically improve their performance. However, it can be also seen that the robustness of the standard estimators is relatively the same as their modular counterparts, and the robustness improvement of the modular estimators is not significant. All in all, the results of the simulations indicate that the modular frames can significantly improve the accuracy of the estimation; however, it was observed that the robustness of the both standard and modular structures is relatively equal to each other.

Table 11 Details of the cluster centers obtained by SFLSDE for high-frequency data (un-normalized)

Clusters	1st	2nd	3rd	4th	5th	6th	7th	8th
1st experiment								
5 cluster								
x_1	301.37	300.80	300.33	301.13	301.12	–	–	–
x_2	57.48	56.47	56.81	56.92	57.64	–	–	–
x_3	250.14	249.58	250.73	249.73	250.74	–	–	–
6 cluster								
x_1	301.33	300.46	300.44	301.77	301.21	300.35	–	–
x_2	57.91	57.98	57.08	58.34	57.46	58.25	–	–
x_3	248.65	251.35	250.44	249.84	250.56	249.27	–	–
7 cluster								
x_1	301.11	300.13	300.89	302.91	300.97	300.18	298.59	–
x_2	53.74	59.90	59.99	57.21	53.38	59.73	54.54	–
x_3	252.03	249.33	249.99	248.79	250.67	249.09	252.76	–
8 cluster								
x_1	300.05	301.04	299.91	300.28	300.53	302.15	300.72	300.91
x_2	53.38	58.10	53.38	54.47	58.58	59.72	57.71	55.65
x_3	252.19	253.56	251.56	254.59	252.33	249.44	248.63	246.63
2nd experiment								
5 cluster								
x_1	301.23	301.17	300.74	300.497	300.04	–	–	–
x_2	55.91	55.34	56.40	55.25	56.80	–	–	–
x_3	0.440	0.393	0.219	0.280	0.329	–	–	–
6 cluster								
x_1	300.45	299.78	300.39	300.90	300.12	300.17	–	–
x_2	56.34	55.38	56.10	57.77	57.70	56.85	–	–
x_3	0.161	0.387	0.215	0.130	0.497	0.036	–	–
7 cluster								
x_1	300.29	301.97	300.34	300.30	300.11	300.77	301.50	–
x_2	57.94	57.24	53.42	53.38	57.38	58.14	55.32	–
x_3	0.567	0.292	0.423	0.602	0.676	0.546	0.732	–
8 cluster								
x_1	301.22	300.96	302.02	303.82	297.89	300.06	302.74	298.76
x_2	57.89	59.66	60.42	56.35	54.90	59.02	57.86	53.38
x_3	0.917	0.124	0.46	0.51	0.469	0.722	0.007	0.413

Table 12 Number of data handled by each module for the two experiments (low-frequency database)

	1st experiment				2nd experiment			
	5 modules	6 modules	7 modules	8 modules	5 modules	6 modules	7 modules	8 modules
Module 1	426	150	294	284	248	96	136	313
Module 2	559	283	329	835	609	146	105	48
Module 3	210	375	203	12	83	97	372	149
Module 4	351	70	439	22	122	342	395	21
Module 5	121	311	54	50	505	562	72	816
Module 6	0	478	195	40	0	424	284	42
Module 7	0	0	153	381	0	0	303	116
Module 8	0	0	0	43	0	0	0	162

Table 13 Number of data handled by each module for the two experiments (high-frequency database)

	1st experiment				2nd experiment			
	5 modules	6 modules	7 modules	8 modules	5 modules	6 modules	7 modules	8 modules
Module 1	507	379	165	293	159	58	183	126
Module 2	232	414	471	129	544	194	99	59
Module 3	406	80	155	138	84	66	247	168
Module 4	212	46	80	198	90	238	417	390
Module 5	113	274	253	217	593	565	363	374
Module 6	0	277	70	203	0	349	107	116
Module 7	0	0	276	107	0	0	54	122
Module 8	0	0	0	185	0	0	0	115

Table 14 MSE error of each of the modules for low-frequency data

	1st experiment				2nd experiment			
	5 modules	6 modules	7 modules	8 modules	5 modules	6 modules	7 modules	8 modules
Module 1	0.0279	0.0308	0.0302	0.0308	0.0302	0.0305	0.0305	0.0305
Module 2	0.0288	0.0309	0.0300	0.0303	0.0314	0.0302	0.0312	0.0302
Module 3	0.0305	0.0294	0.0291	0.0294	0.0305	0.0289	0.0289	0.0285
Module 4	0.0313	0.0293	0.0270	0.0275	0.0293	0.0293	0.0294	0.0294
Module 5	0.0289	0.0284	0.0274	0.0284	0.0284	0.0295	0.0304	0.0298
Module 6	0	0.0288	0.0265	0.0311	0	0.0306	0.0295	0.0305
Module 7	0	0	0.0301	0.0306	0	0	0.0308	0.0305
Module 8	0	0	0	0.0308	0	0	0	0.0308
Total	0.0293	0.0294	0.0292	0.0304	0.0300	0.0298	0.0298	0.0300

Table 15 MSE error of each of the modules for high-frequency data

	1st experiment				2nd experiment			
	5 modules	6 modules	7 modules	8 modules	5 modules	6 modules	7 modules	8 modules
Module 1	0.0287	0.0252	0.0294	0.0285	0.0294	0.0294	0.0282	0.0275
Module 2	0.0294	0.0294	0.0278	0.0294	0.0288	0.0284	0.0276	0.0284
Module 3	0.0297	0.0312	0.0265	0.0297	0.0286	0.0266	0.0254	0.0255
Module 4	0.0304	0.0341	0.0231	0.0254	0.0278	0.0274	0.0229	0.0264
Module 5	0.0286	0.0319	0.0265	0.0258	0.0315	0.0286	0.0283	0.0293
Module 6	0	0.0214	0.0238	0.0241	0	0.0241	0.0226	0.0243
Module 7	0	0	0.0243	0.0255	0	0	0.0237	0.0254
Module 8	0	0	0	0.0266	0	0	0	0.0279
Total	0.0293	0.0275	0.0265	0.0268	0.0299	0.0273	0.0256	0.0271

7 Conclusions and future work

In this investigation, the authors proposed a modular ridge randomized neural network (M-RRNN) with a nature-inspired distributor to sea ice thickness along the Labrador coast. The proposed modular estimator used unsupervised learning to partition the spatio-temporal information consisting of latitude and longitude values and either sea ice thickness or sea ice temperature. Thereafter, each of the K partitions is fed to K independent RRNNs to create a nonlinear map between the input features, consisting of the AMSR-E brightness temperatures and data from the forecasting system, and the sea-ice thickness from MODIS. Based on a comprehensive comparative analysis, the following conclusions were derived:

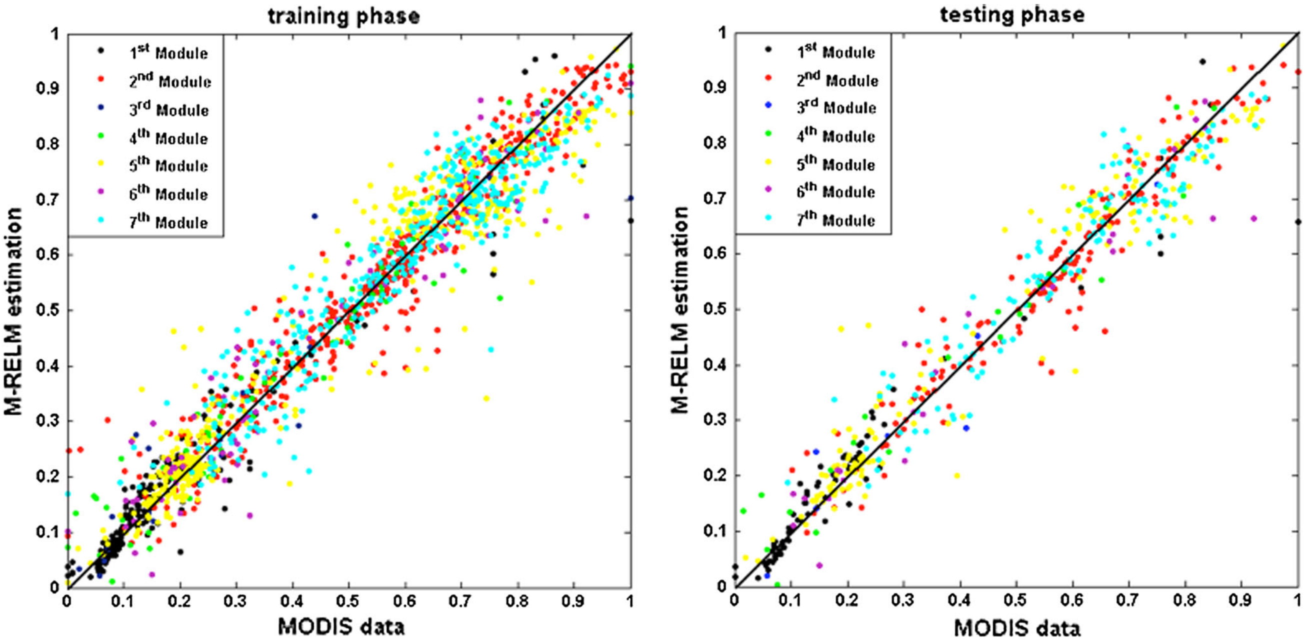


Fig. 8 Correlation of the estimation ice thickness using the M-RRNN vs. ice thickness from MODIS for training and testing phases for low-frequency database

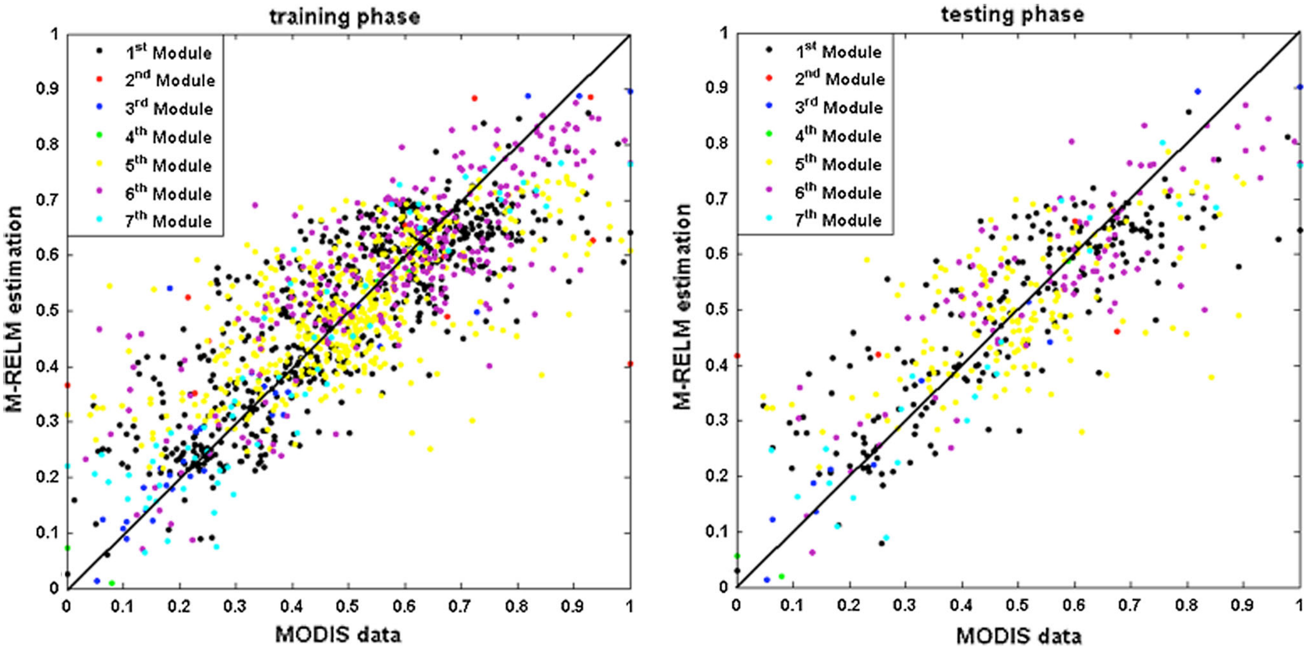


Fig. 9 Correlation of the estimation ice thickness using the M-RRNN vs. ice thickness from MODIS for training and testing phases for high-frequency database

(a) It was observed that the modified scale-factor local search differential evolutionary algorithm (SFLSDE) can be used as an efficient distributor to partition the captured data into a set of subgroups. Furthermore, the comparative studies clearly demonstrated that SFLSDE can surpass PSO, FA, GA, and ABC for the same problem.

(b) The results of the analysis revealed that the use of RRNN as module at the heart of the proposed modular architecture can outperform the modular identifiers using BPNN, OPRNN, RNN, and MLFF-IOP modules for estimation. Furthermore, it was indicated that RRNN has an analytical training strategy which always gets around singularity because of using the ridge norm.

Table 16 MSE of the modular frameworks with different estimation modules for low-frequency database

	M-RRNN	M-RNN	M-OPRNN	M-BPNN	M-MLFF
Accuracy					
Training error	0.0292	0.0297	0.0292	0.0301	0.0298
Testing error	0.0308	0.0311	0.0310	0.0312	0.0309
Robustness					
Training error	0.0003	0.0004	0.0004	0.0004	0.0003
Testing error	0.0004	0.0005	0.0006	0.0008	0.0007

Table 17 MSE of the modular frameworks with different estimation modules for high-frequency database

	M-RRNN	M-RNN	M-OPRNN	M-BPNN	M-MLFF
Accuracy					
Training error	0.0256	0.0276	0.0260	0.0285	0.0291
Testing error	0.0336	0.0355	0.0341	0.0321	0.0334
Robustness					
Training error	0.0003	0.0005	0.0004	0.0004	0.0005
Testing error	0.0003	0.0005	0.0004	0.0005	0.0007

Table 18 Comparing the accuracy and robustness of the proposed modular estimator with standard counterparts with the same structural complexity for low-frequency database

	M-RRNN	RNN	OPRNN	BPNN	MLFF
Accuracy					
Training error	0.0292	0.0299	0.0293	0.0304	0.0302
Testing error	0.0308	0.0315	0.0313	0.0318	0.0312
Robustness					
Training error	0.0003	0.0007	0.0006	0.0005	0.0006
Testing error	0.0004	0.0005	0.0006	0.0007	0.0008

Table 19 Comparing the accuracy and robustness of the proposed modular estimator with standard counterparts with the same structural complexity for the high-frequency database

	M-RRNN	RNN	OPRNN	BPNN	MLFF
Accuracy					
Training error	0.0256	0.0302	0.0293	0.0308	0.0306
Testing error	0.0336	0.0378	0.0368	0.0377	0.0359
Robustness					
Training error	0.0003	0.0007	0.0003	0.0007	0.0008
Testing error	0.0003	0.0005	0.0006	0.0008	0.0008

(c) By comparing the resulting modular randomized neural network (M-RNN) with standard identifiers of the same computational complexity, it was realized that the modular estimator has a higher robustness and accuracy.

(d) The successful performance of M-RRNN inspires further studies of intelligent approaches in a complementary manner with the current physical and semi-empirical estimation tools. The findings of the current research contributes to the use of advanced methods from intelligent computing to geophysical problems.

In future, the authors would like to further expand this study by trying to add a degree of uncertainty to the model using the fuzzy numbers theory to find out its impact on suppressing the undesired effects of random white noise associated with the sensory data.

Acknowledgements This work was a part of a project funded by the Natural Sciences and Engineering Research Council (NSERC) of Canada. Their support is gratefully acknowledged.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix 1

Algorithm 1. Golden sectioning local search

1. Verify the upper and lower bounds of searching domain ($a = -1$ and $b = 1$).
2. Generate two intermediate values, i.e. $F^1(i)$ and $F^2(i)$, as below:
$$F^1(i) = b - \frac{b-a}{\phi},$$
$$F^2(i) = a + \frac{b-a}{\phi},$$
where $\phi = 1.6181$.
3. Proceed with the mutation operator to find two different mutant solutions:
$$\mathbf{v}^j(i) = \mathbf{s}_1(i) + F^j(i)\left(\mathbf{s}_3(i) - \mathbf{s}_2(i)\right), \quad j = 1, 2.$$
4. Proceed with the crossover mechanism to determine the final solutions:
$$u_k^j(i) = \begin{cases} s_k^j(i) & \text{if rand}(0,1) < \text{CR}(i) \\ v_k^j(i) & \text{otherwise} \end{cases}$$
for $j = 1, 2$ and $k = 1, \ldots, d$, where d represents the dimensionality of the optimization problem.
5. Evaluate the fitness of each of the two obtained offsprings.
6. If fitness($\mathbf{u}^1(i)$)>fitness($\mathbf{u}^2(i)$), then $\mathbf{u}(i) = \mathbf{u}^1(i)$, $F(i) = F^2(i)$ and $b = F^2(i)$; else, $\mathbf{u}(i) = \mathbf{u}^2(i)$, $F(i) = F^2(i)$ and $a = F^1(i)$.
7. If the termination criteria are satisfied stop the process, otherwise return to Step 2.

Appendix 2

Algorithm 2. Hill climbing local search

1. Define an initial value for climbing h.
2. Produce three intermediate values, i.e. $F^1(i)$, $F^2(i)$ and $F^3(i)$, as below:
$$F^1(i) = F(i) - h$$
$$F^2(i) = F(i)$$
$$F^3(i) = F(i) + h.$$
3. Perform the mutation operator using the obtained scale factors to find the mutant solutions:
$$\mathbf{v}^j(i) = \mathbf{s}_1(i) + F^j(i)\left(\mathbf{s}_3(i) - \mathbf{s}_2(i)\right), \quad j = 1, 2, 3.$$
4. Proceed with the crossover mechanism to calculate the final solutions:
$$u_k^j(i) = \begin{cases} s_k^j(i) & \text{if rand}(0,1) < \text{CR}(i) \\ v_k^j(i) & \text{otherwise} \end{cases}$$
for $j = 1, 2, 3$ and $k = 1, \ldots, d$.
5. Calculate the fitness of each of the three obtained offsprings and extract the scale factor that yields the best fitness $F^*(i)$.
6. If $F^*(i) = F(i)$ then $h = h/2$; else, $\mathbf{u}(i) = \mathbf{u}^*(i)$ and $F(i) = F^*(i)$.
7. If the termination criteria are satisfied stop the process; otherwise, return to Step 2.

Appendix 3

Algorithm 3. Standard operators

1. Let F_ℓ and F_u be the minimum and maximum scale factor values in Table 2, respectively. Update the scale factor using the following formula:
$$F(i) = \begin{cases} F_\ell + F_u u_1 & \text{if } u_2 < \tau_1 \\ F(i) & \text{otherwise.} \end{cases}$$
2. Update the crossover rate factor using the following formula:
$$\text{CR}(i) = \begin{cases} u_3 & \text{if } u_4 < \tau_2 \\ \text{CR}(i) & \text{otherwise.} \end{cases}$$
3. Proceed with the mutation operator to find the mutant solutions:
$$\mathbf{v}(i) = \mathbf{s}_1(i) + F(i)\left(\mathbf{s}_3(i) - \mathbf{s}_2(i)\right).$$
4. Proceed with the crossover mechanism to calculate the final solutions:
$$u_k(i) = \begin{cases} s_k(i) & \text{if rand}(0,1) < \text{CR}(i) \\ v_k(i) & \text{otherwise.} \end{cases}$$
for $k = 1, \ldots, d$.

References

Belchansky GI, Douglas DC, Platonov NG (2008) Fluctuating Arctic sea ice thickness changes estimated by an in situ learned and empirically forced neural network model. J Clim 21(4):716–729

Broomhead DS, Lowe D (1988) Multivariable functional interpolation and adaptive networks. Complex Syst 2:321–355

Burger M, Neubauer A (2003) Analysis of Tikhonov regularization for function approximation by neural networks. Neural Netw 16(1):79–90

Chuang L-Y, Hsiao C-J, Yang C-H (2011) Chaotic particle swarm optimization for data clustering. Expert Syst Appl 38(12):14555–14563

Ding Y, Feng Q, Wang T, Fu X (2014) A modular nuerla network architecture with concept. Neurocomputing 125(11):3–6

Eicken H (2013) Arctic sea ice needs better forecasts. Nature 497:431–433

Farooq A (2000) Biologically inspired modular neural networks. Ph.D. thesis, Virginia Tech

Fodor JA (1983) Modularity of mind: an essay on faculty psychology. MIT Press, Cambridge

Hall DK, Key JR, Casey KA, Riggs GA, Cavalieri DJ (2004) Sea ice surface temperature product from MODIS. IEEE Trans Geosci Remote Sens 42(5):1076–1087

Hall DK, Riggs GA, Salomonson VV (2007) MODIS/terra sea ice extent 5-min L2 swath 1 km V005. National Snow and Ice Data Center, Boulder

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning. Number 1. Springer, Berlin

Haverkamp D, Soh L-K, Tsatsoulis C (1995) A comprehensive, automated approach to determining sea ice thickness from SAR data. IEEE Trans Geosci Remote Sens 33(1):46–57

Hornik K (1991) Approximation capabilities of multilayer feedforward networks. Neural Netw 4(2):251–257

Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1):489–501

Huang P-S, Deng L, Hasegawa-Johnson M, He X (2013) Random features for kernel deep convex network. In: 2013 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 3143–3147

Iwamoto K, Ohshima KI, Tamura T, Nihashi S (2013) Estimation of thin ice thickness from AMSR-E data in the Chukchi sea. Int J Remote Sens 34(2):468–489

Javadi M, Abbaszadeh ASAA, Sajedin A, Ebrahimpour R (2013) Classification of ECG arrhythmia by a modular neural network based on mixture of experts and negative correlated learning. Biomed Signal Process Control 8:289–296

Kaleschke L, Tian-Kunze X, Maab N, Makynen M, Matthias D (2012) Sea ice thickness retrieval from SMOS brightness temperatures during the Arctic freeze-up period. J Geophys Res 39. doi:10.1029/2012GL050916

Karaboga D, Ozturk C (2011) A novel clustering approach: artificial bee colony (ABC) algorithm. Appl Soft Comput 11(1):652–657

Le Q, Sarlós T, Smola A (2013) Fastfood-approximating kernel expansions in loglinear time. In: Proceedings of the international conference on machine learning

Lin H, Yang L (2012) A hybrid neural network model for sea ice thickness forecasting. In: 2012 eighth international conference on natural computation (ICNC). IEEE, pp 358–361

Lowe D (1989) Adaptive radial basis function nonlinearities, and the problem of generalisation. In: First IEE international conference on (Conf. Publ. No. 313) Artificial neural networks. IET, pp 171–175

LukošEvičIus M, Jaeger H (2009) Reservoir computing approaches to recurrent neural network training. Comput Sci Rev 3(3):127–149

Meireles MRG, Almeida PEM, Simoes MG (2003) A comprehensive review for industrial applicability of artificial neural networks. IEEE Trans Ind Electron 50(3):585–601

Melin P (2012) Modular neural networks and type-2 fuzzy systems for pattern recognition. Studies in computational intelligence. Springer, Berlin, Heidelberg

Mozaffari A, Fathi A (2012) Identifying the behaviour of laser solid freeform fabrication system using aggregated neural network and the great salmon run optimization algorithm. Int J Bio-Inspir Comput 4(5):330–343

Mozaffari A, Behzadipour S, Kohani M (2014) Identifying the tool-tissue force in robotic laparoscopic surgery using neuro-evolutionary fuzzy systems and a synchronous self-learning hyper level supervisor. Appl Soft Comput 14:12–30

Neri F, Tirronen V (2009) Scale factor local search in differential evolution. Memet Comput 1:153–171

Nihashi S, Ohshima KI, Tamura T, Fukamachi Y, Saitoh S (2009) Thickness and production of sea ice in the okhotsk sea coastal polynyas from AMSR-E. J Geophys Res: Oceans 114(C10)

Pao YH, Park GH, Sobajic DJ (1994) Learning and generalization characteristics of the random vector functional-link net. Neurocomputing 6(2):163–180

Park J, Sandberg IW (1991) Universal approximation using radial-basis-function networks. Neural Comput 3(2):246–257

Pratama M, Anavatti SG, Angelov PP, Lughofer E (2014a) PANFIS: a novel incremental learning machine. Neural Netw Learn Syst IEEE Trans 25(1):55–68

Pratama M, Anavatti SG, Lughofer E (2014b) GENEFIS: toward an effective localist network. Fuzzy Syst IEEE Trans 22(3):547–562

Pratama M, Anavatti S, Lu J (2015a) Recurrent classifier based on an incremental meta-cognitive-based scaffolding algorithm. Fuzzy Syst IEEE Trans 23(6):2048–2066

Pratama M, Anavatti SG, Joo M, Lughofer ED (2015b) pClass: an effective classifier for streaming examples. Fuzzy Syst IEEE Trans 23(2):369–386

Pratama M, Lu J, Zhang G (2015c) Evolving type-2 fuzzy classifier. Fuzzy Syst IEEE Trans. doi:10.1109/TFUZZ.2015.2463732

Rahimi A, Recht B (2007) Random features for large-scale kernel machines. In: Advances in neural information processing systems, pp 1177–1184

Raja MAZ, Samar R (2014) Numerical treatment for nonlinear MHD Jeffery–Hamel problem using neural networks optimized with interior point algorithm. Neurocomputing 124:178–194

Rodan Ali, Tiňo Peter (2011) Minimum complexity echo state network. Neural Netw IEEE Trans 22(1):131–144

Rojas R (1996) Neural networks: a systematic introduction. Springer, Berlin

Schmidt WF, Kraaijveld MA, Duin RPW (1992) Feedforward neural networks with random weights. In: Pattern recognition, 1992. Conference B: pattern recognition methodology and systems, Proceedings, 11th IAPR international conference, vol II. IEEE, pp 1–4

Scott KA, Buehner M, Caya A, Carrieres T (2012) Direct assimilation of AMSR-E brightness temperatures for estimating sea ice concentration. Mon Weather Rev 140(3):997–1013

Scott KA, Buehner M, Carrieres T (2014) An assessment of sea-ice thickness along the Labrador coast from AMSR-E and MODIS data for operational data assimilation. IEEE Trans Geosci Remote Sens 52(5):2726–2737

Senthilnath J, Omkar SN, Mani V (2011) Clustering using firefly algorithm: performance study. Swarm Evolut Comput 1(3):164–171

Sheikh RH, Raghuwanshi MM, Jaiswal AN (2008) Genetic algorithm based clustering: a survey. In: International conference on emerging trends in engineering and technology, Nagpur, pp 314–319

Soh L-K, Tsatsoulis C, Gineris D, Bertoia C (2004) ARKTOS: an intelligent system for SAR sea ice image classification. IEEE Trans Geosci Remote Sens 42(1):229–248

Wang X, Key JR, Liu Y (2010) A thermodynamic model for estimating sea and lake ice thickness with optical satellite data. J Geophys Res: Oceans (1978–2012) 115(C12)

Weeks W (2010) On sea ice. University of Alaska Press, Fairbanks

Widrow Bernard, Greenblatt Aaron, Kim Youngsik, Park Dookun (2013) The no-prop algorithm: a new learning algorithm for multilayer neural networks. Neural Netw 37:182–188

Wu L, Moody J (1996) A smoothing regularizer for feedforward and recurrent neural networks. Neural Comput 8(3):461–489

Yamazaki T, Tanaka S (2007) The cerebellum as a liquid state machine. Neural Netw 20(3):290–297

Yu Y, Lindsay RW (2003) Comparison of thin ice thickness distributions derived from RADARSAT Geophysical Processor System and advanced very high resolution radiometer data sets. J Geophys Res: Oceans 108(C12)

Zhang L, Suganthan PN (2015) A comprehensive evaluation of random vector functional link networks. Inf Sci. doi:10.1016/j.ins.2015.09.025