Journal of Ambient Intelligence and Humanized Computing (2020) 11:3405–3416
https://doi.org/10.1007/s12652-019-01543-4

ORIGINAL RESEARCH

β-Hill climbing algorithm with probabilistic neural network for classification problems

Mohammed Alweshah[1] · Aram Al-Daradkeh[1] · Mohammed Azmi Al-Betar[2] · Ammar Almomani[2] · Saleh Oqeili[1]

Received: 25 May 2019 / Accepted: 5 October 2019 / Published online: 19 October 2019

Abstract
Classification is a crucial step in the data mining field. The probabilistic neural network (PNN) is an efficient method developed for classification problems. The success factor of using PNN for classification problems implies in finding the proper weight during classification process. The main goal of this paper is to improve the performance of PNN by finding the best weight for the PNN using the recent local search approach called β-hill-climbing (β-HC) optimizer. This algorithm is an extension version of the traditional hill climbing algorithm in that it uses a stochastic operator to avoid local optima. The proposed approach is evaluated against 11 benchmark datasets ,and the experimental results showed that the proposed β -HC with PNN approach performed better in terms of classification accuracy than the original PNN, HC-PNN and other six well-established approaches using the same experimented benchmarks.

Keywords Probabilistic neural network (PNN) · β-Hill-climbing · Optimization · Classification

1 Introduction

In machine learning, classification is normally known as a supervised learning mechanism where the initial input data is analyzed and used in to learn the system how to classify the future data or observations (Brownlee 2016). Therefore, The classification process is considered as an important task in data mining. In simple terms, classification identifies a set of data objects that resemble each other in the same dataset and are different from the objects in other datasets. In order to manipulate the classification problem, a training set (input dataset) is required that contains example records with a number of attributes. The classification algorithm uses the training dataset to construct a model and that model can be used to allocate unclassified records to one of the defined classes (Alweshah 2018; Alweshah and Abdullah 2015). Examples of classification problems are speech recognition Juang et al. (1997), handwriting recognition (LeCun et al. 1990), biometric identification (Yampolskiy and Govindaraju 2008), document classification (Alweshah et al. 2017b; Yan et al. 2018), image and video classifications (Yan et al. 2019a, b), and many others as reported in the latest comprehensive survey (Alweshah et al. 2017a).

Many techniques have been used to solve classification problems.Examples include the support vector machine (SVM) (Adankon and Cheriet 2009), neural network (NN) (Specht 1991), radial basis function (RBF) network (Craddock and Warwick 1996), naïve Bayes (NB) (Friedman et al. 1997) and many others (Alshareef et al. 2015b; Faris et al. 2016; Aljarah et al. 2018; Khan et al. 2019; Anagaw and Chang 2019; Boveiri et al. 2019). The NN is a common technique for solving classification problems and there are different types of NN, including the feed-forward neural network and multilayer perceptron (MLP), as well as a radial basis function network (RFB), modular networks and the probabilistic neural network (PNN) (Specht 1990).

✉ Mohammed Alweshah
weshah@bau.edu.jo

Aram Al-Daradkeh
aram.daradka@gmail.com

Mohammed Azmi Al-Betar
mohbetar@bau.edu.jo

Ammar Almomani
ammarnav6@bau.edu.jo

Saleh Oqeili
saleh@bau.edu.jo

[1] Prince Abdullah Bin Ghazi Faculty of Information and Communication Technology, Al-Balqa Applied University, Al-Salt, Jordan

[2] Department of Information Technology, Al-Huson University College, Al-Balqa Applied University, Irbid, Jordan

The NN is exceptionally effective and has been utilized to solve many classification problems. It was created in the late 1950s by Rosenblatt in Alweshah (2014). The NN consists of two fundamental components: the neuron and the link. The neuron alludes to a sort of processing component, while the link interfaces two neurons together. Each link has an individual weight. Each neuron gets a simulating signal from other neighboring neurons, and from these signals it forms data and creates an output result (Alshareef et al. 2015a).

The PNN is a strong data mining tool and it is an algorithm that can be used for a huge number and complex (input/output) relationships. The PNN is a special form of NN model with a statistical Bayesian decision rule. The general structure of a PNN consists of four layers: (1) an input layer, where the dimensions of the input vector transcribes the dimension of the input layer; (2) a pattern layer (second layer), where the dimension of the pattern layer is equal to the dimension of the number of examples in the training set; (3) a summation layer (third layer), which contains the number of classes in the set of examples, and (4) a decision layer (fourth layer), which classifies the test example into predefined classes.

The success of the PNN is depended on the proper selection of the weight during the classification process (Iliou and Anagnostopoulos 2010). The way of selecting the optimal value of weight vector for PNN is recently modelled as an optimization problem (Iliou and Anagnostopoulos 2010). Several optimization problems such as metaheuristics have been recently utilized to tackle such a problem. metaheuristics-based optimization approaches are categorized into either single-solution-based or population-based searches (Blum and Roli 2003).

Population-based approaches focus on maintaining and improving multiple candidate solutions.They are normally manipulating several solutions at a time using recombination, mutation and selection operators. Although they are very efficient in handling several search space regions at the same time, they cannot find the local optimal solution at each search space regions to which they handled. Two main population-based metaheuristics are popularly known: evolutionary algorithms and swarm intelligence. Evolutionary algorithms include genetic algorithms (GAs) (Hu et al. 2016), harmony search algorithm (Geem et al. 2001). Another category of metaheuristics is swarm intelligence. Examples of this category include ant colony optimization (ACO) (Dorigo and Stützle 2010), PSO, social cognitive optimization, the penguin search optimization algorithm (PeSOA), the artificial bee colony (ABC) (Zhu and Kwong 2010) and the Grey Wolf Optimiser (GWO) algorithm (Al Nsour et al. 2019).

The single-solution approaches are initiated with a single provisional solution. Iteratively, that solution can be improved by moving to its neighbouring solution until the local optimal solution is reached. Examples of single-solution-based used for NN include simulated annealing (SA) (Ren and Qu 2014), iterated local search (ILS) (El-Bouri 2012), variable neighborhood search (VNS) (Xie et al. 2012), guided local search (GLS) (Tairan and Zhang 2010) and β-hill climbing (Al-Betar 2017).

β-Hill clibming is a recent local-search based algorithm (Al-Betar 2017) It is able to escape the local optimal trap using β-operator. It has several successful features such as it has few parameters to be set in the initial search. It is a very simple optimization frameworks. It can be flexibly adapt for any kind of optimization problems such as economic load dispatch (Al-Betar et al. 2018), denoising electrocardiogram (ECG) signals (Alyasseri et al. 2018, 2017b, a), Multiple-reservoir scheduling (Alsukni et al. 2017), sudoku game (Al-Betar et al. 2017), Substitution-Boxes (Alzaidi et al. 2018), gene selection (Alomari et al. 2018) and feature selection (Abualigah et al. 2017). Furthermore, theoretical concepts of β-hill climbing related to the adaptive parameter settings are also improved (Al-Betar et al. 2019). Also, β-hill climbing is hybridized with other population-based algorithms to improve exploitation concepts (Abed-alguni and Alkhateeb 2018).

In this paper, the β-hill-climbing (β-HC) algorithm is adapted to seek for the optimal weight for the PNN so as to improve the performance (accuracy). The weight of PNN is modelled as an optimization problem to be solved by β-HC. In order to evaluate the proposed method, 11 test benchmarks of classification problems are used. Comparative evaluation between PNN without β-HC, PNN with hill climbing, PNN with β-HC shows that the PNN with β-HC is able to overcome the PNN significantly without β-HC, PNN with hill climbing. It can find the optimal weight for the PNN and strike the right balance between the exploration and exploitation during the search.

The rest of this paper is organized as follows: In Sect. 2, the background of PNN algorithm and β-HC is described. Then, in Sect. 3 the proposed β-hill-climbing-based PNN is discussed in detail. In Sect. 4, the experimental results are presented and analyzed. Finally, Sect. 5 concludes the paper and provide some possible directions for future improvements.

2 Research background

2.1 Probabilistic neural network (PNN)

The PNN introduced by Specht (1990) has been shown to have effective classification performance, even when used to a wide variety of different problems (Berrar et al. 2002; Wasserman 1993; Specht 1990; Sweeney et al. 1994; Mao et al. 2000; Kwigizile et al. 2004; Manimala and Selvi 2007;

Araghi et al. 2009). The training of a PNN does not involve the implementation of heuristic searches to get the best smoothing factor, for the training, which is an optimization problem. A PNN uses a statistical algorithm, known as kernel discriminant analysis (Mika et al. 1999), in which the methods are organized into a four-layered feed-forward network consisting of an input layer, hidden layer, pattern/summation layer and decision/output layer, as shown in Fig. 1.

Figure 1 illustrates a typical PNN model. Each input neural symbolizes a distinct feature in the training/test data sets. The number of the inputs is equivalent to the number of features in the data set. A PNN has a relatively quicker training process than BP. It basically has an analogous structure, which ensures convergence with an ideal classifier as the dimension of the associated training set increases, and has training samples that can be possibly included or eliminated without the need for substantial retraining (Wasserman 1993). The four layers of the PNN network are described below:

Input layer: This contains one neuron for each indicator variable. For the categorical factors, the N − 1 neurons are connected, where N shows the number of categories. The input neuron supposedly standardizes the value range by subtracting the middle value. after that, it divides it between quartile range value. From this point onward, the input neurons are believed to sustain these qualities in each neuron exhibited inside the hidden layer.

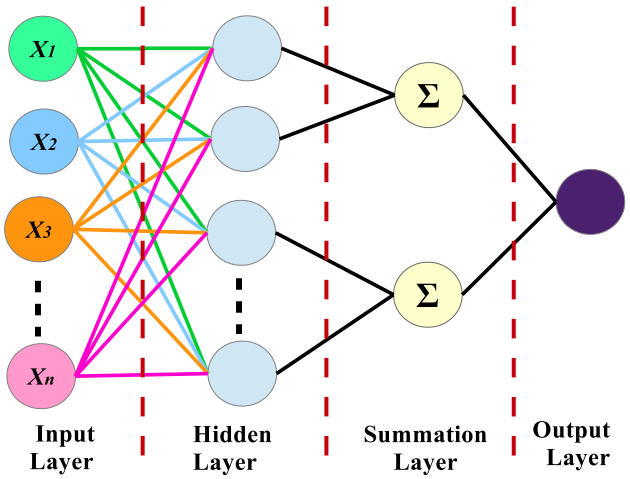

Fig. 1 Architecture of a typical probabilistic neural network

Hidden layer: This contains a single neuron for each case in the training dataset. The neurons store the indicator variable values and the objective values for the cases. At the point when the x vector of the input values of the input layer is displayed, the Euclidean separation from the center point of the neuron for the experiment is ascertained by the hidden neurons and, after that, with the assistance of the sigma value the RBF kernel function is applied. The resultant value is then passed to the neurons in the pattern layer.

Pattern/summation layer: For each class of objective factors, a single pattern neuron is available. The objective class for each training group is put away alongside each hidden neuron, where the weight value emerging from the hidden neurons is passed to the pattern neurons that correspond to the type of hidden neurons. The pattern neurons then include the values for their representation classes (hence, these values refer to the weighted votes for the particular category). In practice, this layer operate a dot product process (Z) on both input vector $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)$ and weight vector $\boldsymbol{W} = (w_1, w_2, \ldots, w_n)$ such that $Z = X.W$. Thereafter, the non-linear activation function is triggered using Z as shown in Eq. (1)

$$f(X) = \exp \frac{(w_i - x)^t - (w_i - x)}{2\sigma^2} \tag{1}$$

where σ is the smoothing parameter that should be set in the initial training process.

Decision/output layer: This final layer compares the weight votes in favour of each target category that is

gathered in the pattern layer and, after that, uses the highest vote to predict the target group.

For mathematical formulations, the interpretation of PNN concepts is modelled as a multilayer framework of an FFNN as shown in Eq. (2).

$$h_i \times c_i \times f_i > h_j \times c_j \times f_j \quad \forall i \neq j \tag{2}$$

where h_i is the preceding prospect, in which the new object belongs to class i, and c_i is the cost of misclassifying an object that belongs to class i, and f_i is the probability density function (PDF) of class i. On the other hand, it is significant that $\boldsymbol{X}$ is the input vector that has to be classified. In general, the prior probabilities and cost of misclassification are called as priority when dealing with PNNs and kept equivalent; therefore the Bayes optimal decision rule can be reduced using Eq. (3).

$$g_i(\boldsymbol{X}) > g_j(\boldsymbol{X}) \quad \forall i \neq j. \tag{3}$$

This rule discloses that the outstanding classification choice can be developed just by accomplishing straightforward evaluations upon identifying the PDFs of the various classes. In particular, the PDF for a single class can be estimated based on Eq. (4):

$$g_i(\boldsymbol{X}) = \frac{1}{n_i \times \sigma} \sum_{k=1}^{n_i} \frac{X - x_{ik}}{\sigma} \tag{4}$$

where σ is a smoothing parameter, W is the weighting function, X is the unidentified input sample that has to be classified, X_{ik} is the kth training input from the ith class, n is the number of training inputs for class i, and $g_i(\boldsymbol{X})$ is the PDF approximation for class i.

Generally, the Euclidean distance is applied to determine the closeness or range between the training sample and the unidentified input. The Gaussian function is a typically used weighting function because it demonstrates the above-stated attributes and can be quickly calculated. Supervised learning provides an input pattern and changes the network factors (weights) to reduce the distance between the calculated output and the expected output. By setting the weighting function to the Gaussian function it is computed as shown in Eq. (5):

$$g_i(\boldsymbol{X}) = \frac{1}{n_i \times \sigma} \sum_{k=1}^{n_i} \exp^{\frac{(x-x_{ik})^2}{\sigma^2}}. \tag{5}$$

2.2 β-Hill climbing optimizer

The β-hill climbing is a recent local search-based algorithm introduced in Al-Betar et al. (2017). β-Hill climbing is initialized with a random solution $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$. Iteratively, that solution undergoes changes using three main operators: (1) $\mathcal{N}$-operator, (2) β-operator, and (3) $\mathcal{S}$-operator. β-Hill climbing is normally stopped when the maximum number of iterations is met. The feasibility of the solution is normally preserved during the search. Note that the hill climbing (HC) algorithm is the same of the β-hill climbing but without β-operator. Algorithm 1 pseudo-coded the basic version of β-hill climbing. The three operators are described as follows (Fig. 2):

$\mathcal{N}$-operator This operator is used to move the current solution $\boldsymbol{x}$ to its neighboring solution $\boldsymbol{x'}$ using a movement strategy with a probability of $\mathcal{N}$ in which the feasibility is maintained. The line 5 in Algorithm 1 is the pseudo-code for $\mathcal{N}$-operator. For more clarification, let x_i be assigned by a value of $v_i(k)$ of k^{th} location, the value of x'_i will be assigned by value as follows:

$$x'_i = v_{i,k} \pm m \tag{6}$$

where $v_{i,k} \pm m$ is the neighboring value of $v_i(k)$. It is worth mentioning that the $\mathcal{N}$-operator is invoked once at each iteration using a random walk mechanism. This means that the effect of moving to the neighbouring solution is not checked until the $\mathcal{S} - operator$ is invoked.

β-operator The β-operator is responsible for the exploration process by applying a uniform mutation to the current solution. $\forall i \in (1, 2, \ldots, n)$, the decision variable x_i is randomly selected to be changed using the β parameter as in Eq. (7). The lines from 6 to 10 in Algorithm 1 are the pseudo-code for β-operator.

$$x'_i \leftarrow \begin{cases} v_r & U[0,1] \leq \beta \\ i'_i & otherwise. \end{cases}$$

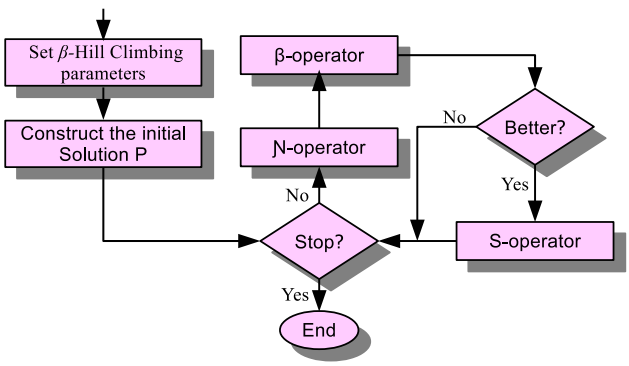

Fig. 2 β-Hill climbing optimizer

It is worth mentioning that β determines how often the uniform mutation is applied.

$\mathcal{S}$-operator The improved solution $\boldsymbol{x}'$ is evaluated using $f(\boldsymbol{x}')$. The current solution $\boldsymbol{x}$ will be replaced by the improved solution $\boldsymbol{x}'$, if better (i.e., $f(\boldsymbol{x}') \leq f(\boldsymbol{x})$). The lines from 11 to 13 in Algorithm 1 are the pseudo-code for $\mathcal{S}$ -operator.

The proposed method was assessed by figuring four counts. True positives (*TP*) refer to the class with the number of correctly assigned records; true negatives (*TN*) refer to the number of correct instances that are not part of the class; false positives (*FP*) refer to the number of instances that are incorrectly assigned; false negatives (*FN*) refer to the positive tuples that are labelled incorrectly.

Algorithm 1 Original β–hill climbing pseudo-code

```
1: x = (x_1, x_2, ..., x_n) { Generate the initial solution x}
2: Calculate f(x)
3: itr = 0
4: while (itr ≤ Max_Itr) do
5:    x' = x'_i = ν_{i,k±m} {N-operator}
6:    for i = 1, ··· , n do
7:       if (U[0,1] ≤ β) then
8:          x'_i = ν_i(r) {β-operator}
9:       end if
10:   end for
11:   if f(x') ≤ f(x) then
12:      x = x' {S-operator}
13:   end if
14:   itr = itr + 1
15: end while
```

3 β-Hill climbing optimizer for PNN classifier

In this paper, the β-HC algorithm is utilized to find the optimal weights that can be efficiently used in the PNN algorithm hoping to increase the accuracy of the classification process. Initially, random weights vector x are generated through the PNN algorithm, and the input values are multiplied by the corresponding weights by a model determined by the PNN. Figure3 shows the solution representation example

As shown in Fig. 4, the proposed method included two parts, the first one represents the PNN algorithm and the other represents the β-HC algorithm. In the first part, the training data is used, and the tested data is subject to the classification process. As for the process of adjusting the weights of the PNN is the responsibility of the β-HC algorithm, then the accuracy of the disaggregated data is calculated and the process is repeated until the completion criterion is met.

Classification accuracy computed in Eq. (7) is a term that refers to a statistical measure that displays how well the classifier is able to correctly identify the objects towards the labelled classes. The error rate computed in Eq.(8) is a measure of the objects that are incorrectly recognized. Furthermore, specificity and sensitivity were also calculated in Eqs. (9) and (10), respectively.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$ErrorRate = 1 - \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

$$Specificity = \frac{TN}{TN + FP} \tag{9}$$

$$Sensitivity = \frac{TP}{TP + TN}. \tag{10}$$

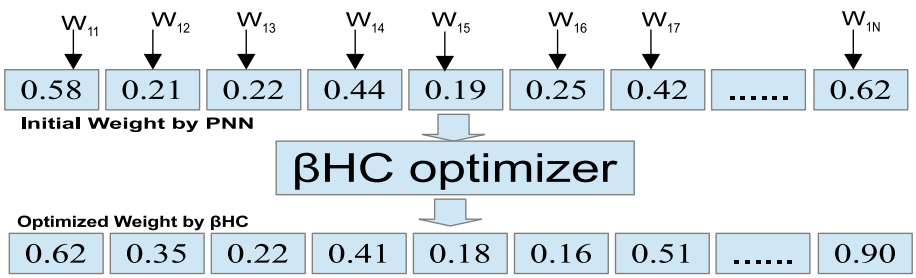

Fig. 3 Representation of initial weights

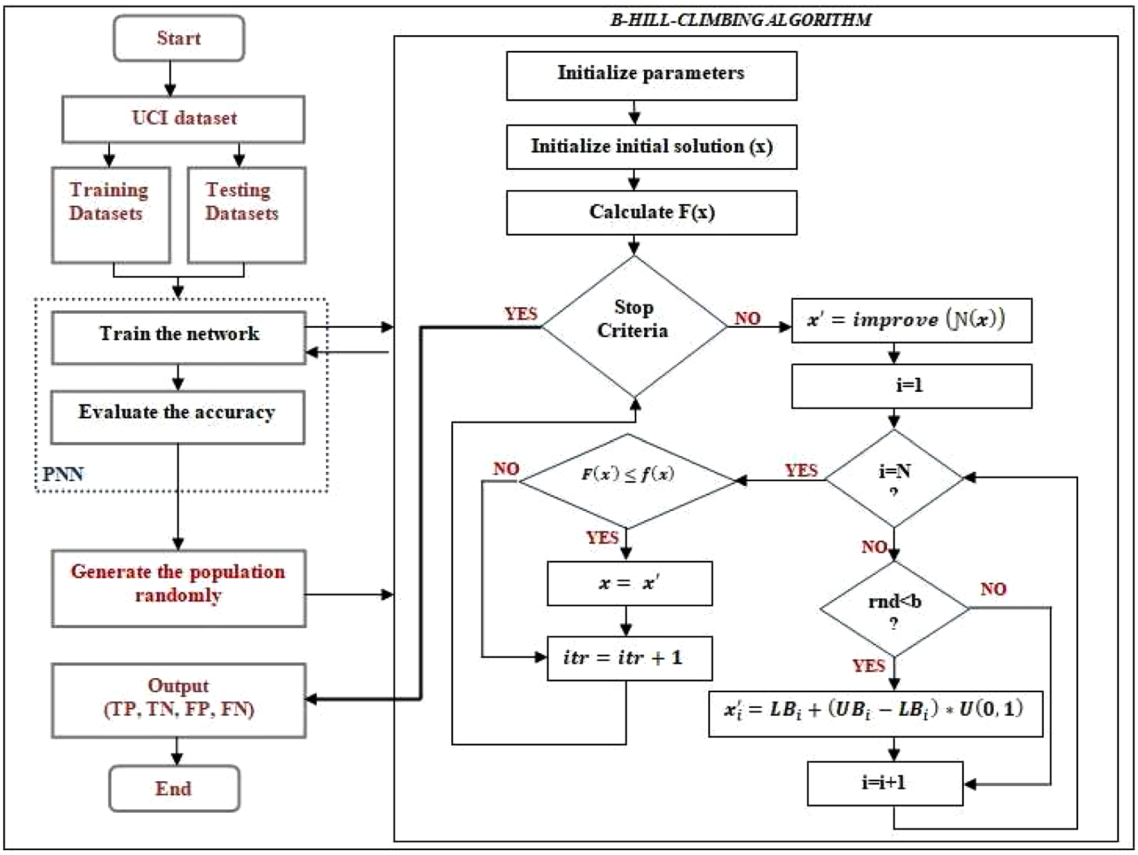

Fig. 4 Flowchart of proposed β-HC-PNN

Table 1 Parameter settings

Parameter	Value
Population size	50
Number of iterations	200
β	0.5

4 Experiments and results

In this section, the results produced by the proposed method is evaluated using a set of experiments. The proposed algorithm are programmed using Matlab R2010 on an Intel® Xeon® CPU E5-1630 v3@3.70 GHz computer are presented. Table 1 shows the parameter settings used for the proposed algorithm, which were determined after intensive preliminary experiments.

The aim of conducting these experiments is to check whether the classification accuracy is maximized using the PNN utilized β-HC.

The quality of the solutions is given in terms of the best accuracy value (in percent), after 30 independent runs for each of the 11 datasets. In this research, four performance measures are used: accuracy, error rate, sensitivity and specificity as formulated in the previous section. Table 2 presents the results for HC-based PNN, β-HC-based PNN and the original PNN classification techniques when applied to the 11 benchmark datasets. The results in bold highlights the best outcomes. The results summarized in table apparently proved that the proposed β-HC based PNN method was able to outperform the original PNN algorithm on all datasets and also mostly obtained better results than HC based PNN.

Figures 5 and 6 presents the simulation results for the convergence characteristics of β-HC based PNN and HC based PNN in comparison with original PNN using the 11 datasets. Each algorithm was run for 200 iterations. These results show that β-HC based PNN can converge faster than HC based PNN except when applied to GCD and ACA. Moreover, β-HC obtains similar convergence trend for the Heart, and API datasets. Interestingly, when applied to the Four class dataset, β-HC achieved 100% accuracy in all iterations.

We also investigate whether the performance of β-HC-PNN has a significant statistical different indicator from HC-PNN and the firefly algorithm (FA) in terms of classification accuracy, sensitivity, and specificity. Accordingly, a

Table 2 Results obtained by PNN, HC and β-HC on the test datasets

Dataset	Approach	TP	FP	TN	FN	Accuracy (%)	Sensitivity	Specificity	Error rate
PID	PNN	35	28	90	39	65.1	0.47	0.76	0.35
	HC-PNN	141	38	320	19	78.65	0.88	0.89	0.11
	β-HC-PNN	142	37	319	20	**81.25**	0.88	0.90	0.11
HSS	PNN	44	12	6	15	64.94	0.75	0.33	0.35
	HC-PNN	139	12	36	19	85.72	0.88	0.75	0.15
	β-HC-PNN	139	12	34	21	**85.72**	0.87	0.74	0.16
AP	PNN	23	1	1	2	88.88	0.92	0.5	0.11
	HC-PNN	54	1	15	1	96.3	0.98	0.94	0.03
	β-HC-PNN	53	2	15	1	**96.3**	0.98	0.88	0.04
BC	PNN	14	9	36	13	69.44	0.52	0.8	0.31
	HC-PNN	46	10	123	14	83.33	0.77	0.92	0.12
	β-HC-PNN	49	7	125	12	**84.72**	0.80	0.95	0.10
LD	PNN	18	15	34	19	60.47	0.49	0.69	0.40
	HC-PNN	73	26	113	21	87.21	0.78	0.81	0.20
	β-HC-PNN	77	22	112	22	**93.02**	0.78	0.84	0.19
Heart	PNN	27	5	23	13	73.53	0.68	0.82	0.26
	HC-PNN	79	3	85	15	86.76	0.84	0.97	0.1
	β-HC-PNN	79	3	90	10	**86.76**	0.89	0.97	0.07
GCD	PNN	133	46	39	32	68.8	0.81	0.46	0.31
	HC-PNN	435	31	164	45	83.6	0.91	0.84	0.11
	β-HC-PNN	439	27	182	27	**80.8**	0.94	0.87	0.08
Parkinson's	PNN	38	1	6	4	89.79	0.90	0.86	0.10
	HC-PNN	38	1	6	4	89.79	0.90	0.86	0.10
	β-HC-PNN	95	0	35	1	**91.84**	0.99	1.00	0.01
SPECTF	PNN	49	4	5	9	80.59	0.84	0.56	0.19
	HC-PNN	133	12	30	5	92.54	0.96	0.71	0.09
	β-HC-PNN	138	7	30	5	93.04	0.99	0.94	0.02
ACA	PNN	60	14	84	15	83.24	0.8	0.86	0.17
	HC-PNN	193	15	237	20	94.22	0.91	0.94	0.08
	β-HC-PNN	197	11	245	12	**93.06**	0.94	0.96	0.05
Fourclass	PNN	78	0	138	0	100	1	1	0
	HC-PNN	78	0	138	0	100	1	1	0
	β-HC-PNN	78	0	138	0	**100**	1	1	0

sample T test with a significance interval of 95% ($\alpha = 0.05$) is used. The results show that there is a significant difference between the comparative algorithms in favour to β-HC-PNN. To elaborate, with regards to accuracy, β-HC-PNN is statistically different from HC-PNN and FA (p value < 0.05). As for sensitivity and specificity, β-HC is also statistically different from HC-PNN and FA. Figure 7 presents the box plots that demonstrate the distribution resolution quality obtained by β-HC, HC-PNN and FA for the six tested datasets used Gorunescu (2011).

We also assessed all the computational results (with regards to classification accuracy) of the proposed β-HC against those of six popular approaches in the literature. The abbreviations of the comparative approaches are listed in Table 3.

Notably, none of the comparative approaches mentioned in Table 4 were applied to whole datasets used in our study, except for FA. Therefore, different combinations of these separate approaches were tested for each dataset, as described in Table 4. The best results are highlighted in bold font.

From the table it can be seen that β-HC ranked first in all datasets except ACA and GCD, thereby achieving the best performance with regards to classification accuracy. Also, HC ranked second. Moreover, β-HC and HC were able to classify the Four class dataset with no miss classifications (100% accuracy). In a nutshell, β-HC outperformed the other comparative approaches for almost all datasets used.

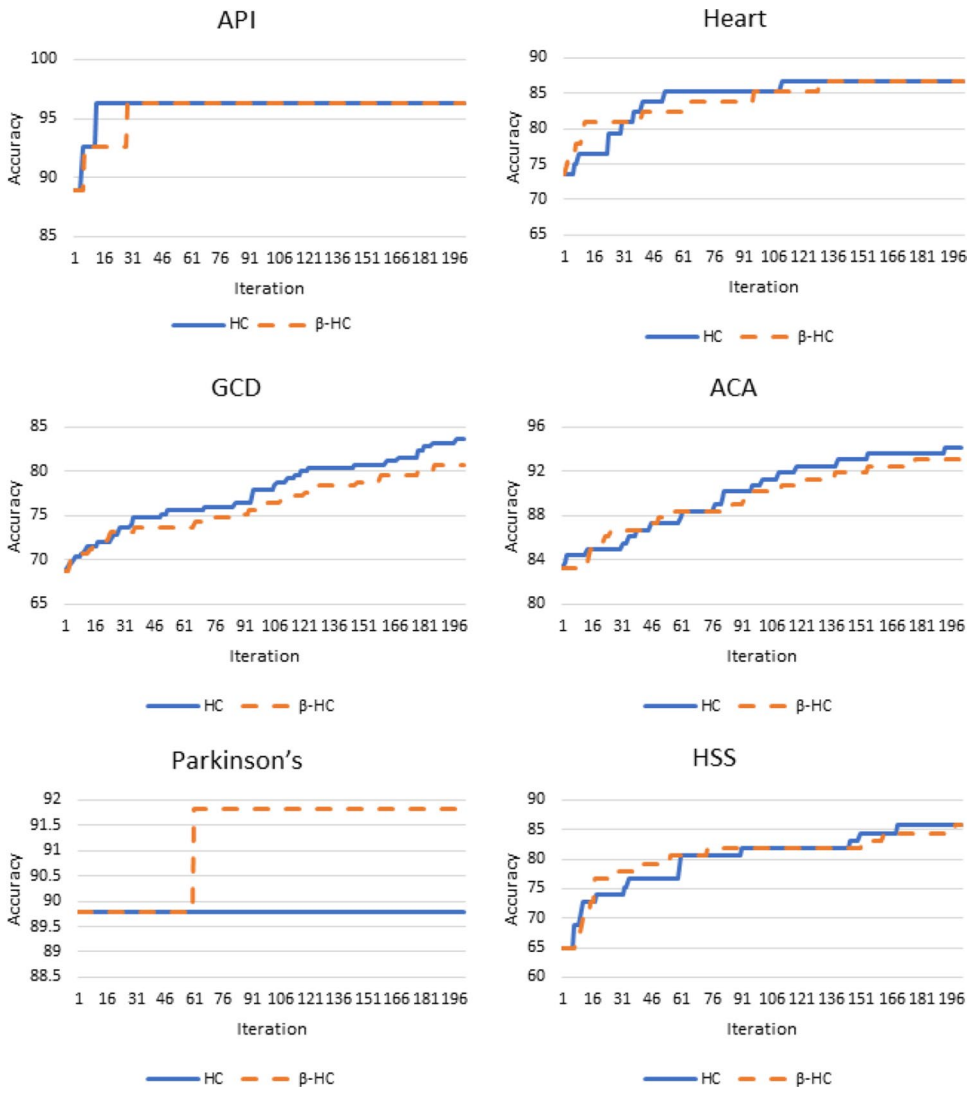

Fig. 5 Convergence characteristics of β-HC-PNN and HC-PNN

5 Conclusion and future work

The key goal of this research is to improve the classification accuracy of the PNN by (1) modifying the PNN's inner weights to discover high-quality solutions, thus improve classification accuracy and (2) attaining a high convergence speed. The proposed approach presented in this paper is the utilization of the β-HC metaheuristic algorithm with the PNN algorithm, where β-HC was utilized to optimize the weights of the PNN. The rationale for utilizing β-HC, which is a nature-inspired population-based algorithm, was based on its capacity to undertake an extensive exploration of the search space, its straightforwardness, and its potential for solving various complicated classification problems.

In order to evaluate the proposed method, 11 benchmark datasets circulated widely to evaluate the new classification methods are used in the experiments. Initially, the three experimented approaches which are PNN, HC-PNN, and β -HC-PNN are compared in terms of Accuracy, Sensitivity, Specificity, and Error rate. Interestingly, for all measures, the proposed β-HC-PNN achieved the best results.

Furthermore, the proposed β-HC-PNN is also compared with three well-established approaches using the same datasets. The results produced by β-HC-PNN are better than that produced by others for nine out of eleven datasets.

evaluated against 11 benchmark datasets and the experimental results showed that the proposed β-HC with PNN approach performed better in terms of classification accuracy than the original PNN, HC-PNN and other six well-established approaches using the same experimented benchmarks. In a nutshell, hybridizing β-HC with PNN improves the classification accuracy of PNN.

In general, the results obtained affirmed the powerful performance of the proposed algorithm. In the future, the following improvement can be investigated:

- The β-HCO can be hybridized with another classification method instead of PNN.

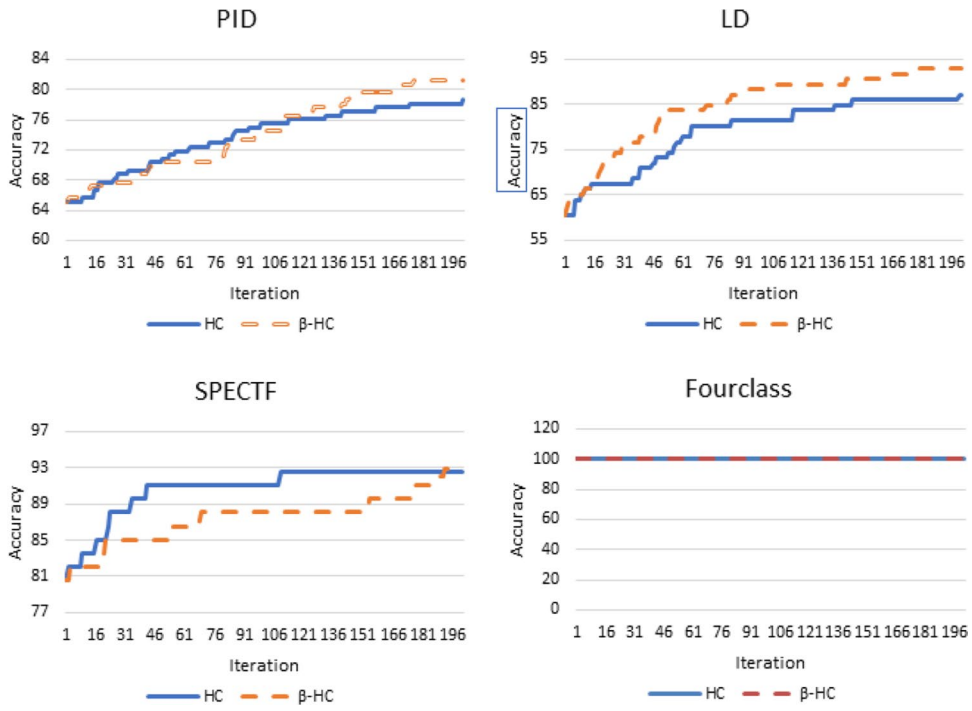

Fig. 6 Continued,Convergence characteristics of β-HC-PNN and HC–PNN

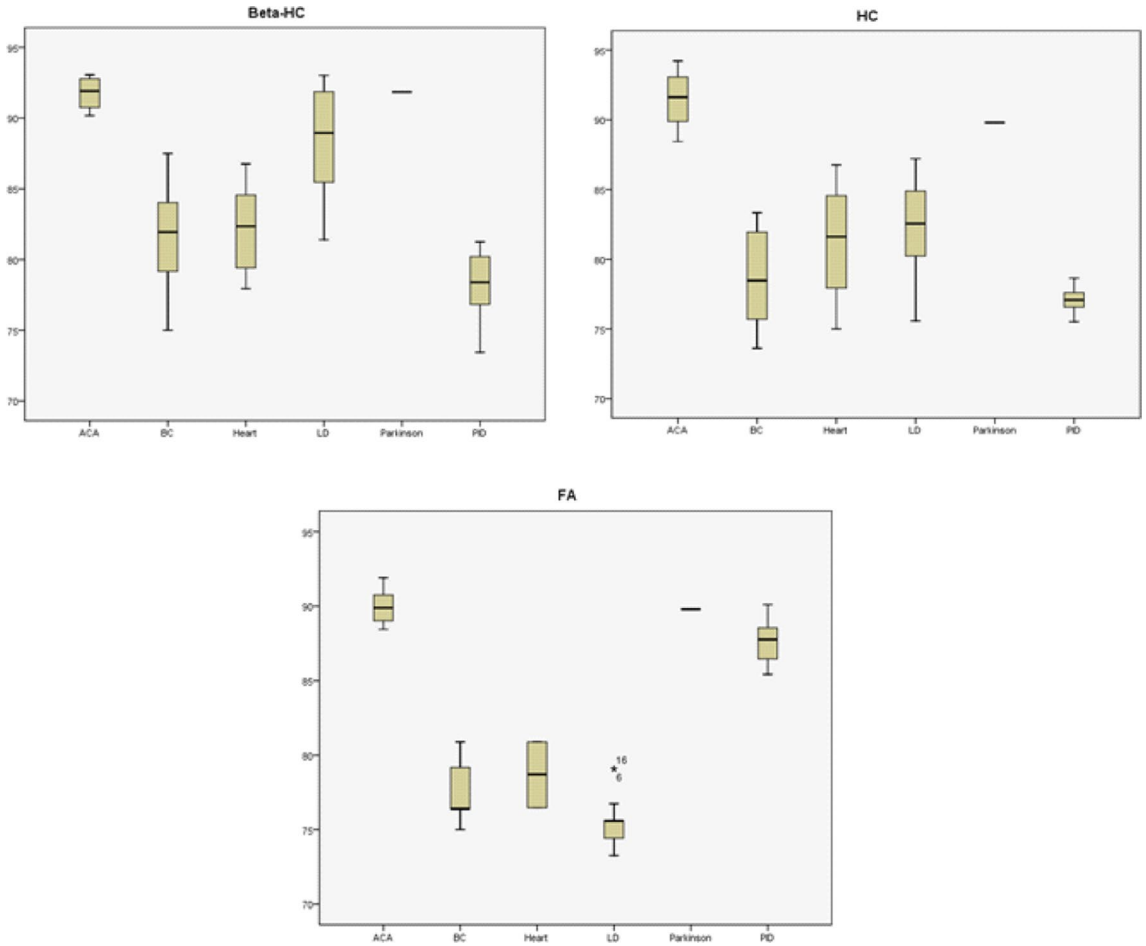

Fig. 7 Box plots for β-HC, HC and FA

Table 3 Acronyms of the compared methods

#	Acronym	Name of approach	References
1	M1	Neural fuzzy inference system	Cpałka et al. (2013)
2	M2	Fuzzy NN	Davis and Warren (2006)
3	M3	BP	Zarndt (1995)

Table 4 Comparison of the results of β-HC and the state-of-the-art

Dataset	Approach	Classification accuracy (%)
PID	β-HC	**81.25** (1st rank)
	HC	78.65
	M5	76.7
	M6	76.04
HSS	β-HC	**85.71** (1st rank)
	HC	85.71
	M5	72.7
	M6	83.12
AP	β-HC	**96.30** (1st rank)
	HC	96.3
	M5	91.4
	M6	92.59
BC	β-HC	**84.72** (1st rank)
	HC	83.33
	M8	73.5
	M5	84.3
	M6	80.88
LD	β-HC	**93.02** (1st rank)
	HC	87.2
	M12	70.97
	M6	79.07
Heart	β-HC	**86.76** (1st rank)
	HC	86.76
	M5	77.9
	M6	80.88
GCD	β-HC	83.60 (2nd rank)
	HC	83.60 (2nd rank)
	M5	**86.4**
	M6	78.4
Parkinson's	β-HC	91.84 (1st rank)
	HC	89.8
	M9	81.3
	M5	85.7
	M6	89.79
SPECTF	β-HC	93.04 (1st rank)
	HC	92.54
	M5	83.6
	M6	92.54
ACA	β-HC	93.06 (2nd rank)
	HC	**94.22**
	M5	69.9
	M6	91.91
Fourclass	β-HC	**100.00** (1st rank)
	HC	100
	M3	99.8
	M5	94.6
	M6	100

- Other classification datasets with combinatorial features can be used for further validation.
- Other local search methods can be investigated to show the strength of β-HCO.

References

Abed-alguni BH, Alkhateeb F (2018) Intelligent hybrid cuckoo search and β-hill climbing algorithm. J King Saud Univ Comput Inf Sci. https://doi.org/10.1016/j.jksuci.2018.05.003

Abualigah LM, Khader AT, Al-Betar MA, Alyasseri ZAA, Alomari OA, Hanandeh ES (2017) Feature selection with β-hill climbing search for text clustering application. In: Information and communication technology (PICICT), 2017 Palestinian international conference on, IEEE, pp 22–27

Adankon MM, Cheriet M (2009) Support vector machine. In: Li SZ, Jain AK (eds) Encyclopedia of biometrics, pp 1303–1308

Al-Betar MA (2017) β-Hill climbing: an exploratory local search. Neural Comput Appl 28(1):153–168. https://doi.org/10.1007/s00521-016-2328-2

Al-Betar MA, Awadallah MA, Bolaji AL, Alijla BO (2017) β-hill climbing algorithm for sudoku game. In: Information and communication technology (PICICT), 2017 Palestinian international conference on, IEEE, pp 84–88

Al-Betar MA, Awadallah MA, Doush IA, Alsukhni E, ALkhraisat H (2018) A non-convex economic dispatch problem with valve loading effect using a new modified β-hill climbing local search algorithm. Arab J Sci Eng 43(12):7439–7456

Al-Betar MA, Aljarah I, Awadallah MA, Faris H, Mirjalili S (2019) Adaptive β-hill climbing for optimization. Soft Comput. https://doi.org/10.1007/s00500-019-03887-7

Al Nsour H, Alweshah M, Hammouri AI, Al Ofeishat H, Mirjalili S (2019) A hybrid grey wolf optimiser algorithm for solving time series classification problems. J Intell Syst. https://doi.org/10.1515/jisys-2018-0129

Aljarah I, Faris H, Mirjalili S, Al-Madi N (2018) Training radial basis function networks using biogeography-based optimizer. Neural Comput Appl 29(7):529–553

Alomari OA, Khader AT, Al-Betar MA, Awadallah MA (2018) A novel gene selection method using modified mrmr and hybrid bat-inspired algorithm with β-hill climbing. Appl Intell 48(11):4429–4447

Alshareef A, Ahmida S, Bakar AA, Hamdan AR, Alweshah M (2015a) Mining survey data on university students to determine trends in the selection of majors. In: 2015 Science and information conference (SAI), IEEE, pp 586–590

Alshareef AM, Bakar AA, Hamdan AR, Abdullah SMS, Alweshah M (2015b) A case-based reasoning approach for pattern detection in malaysia rainfall data. Int J Big Data Intell 2(4):285–302

Alsukni E, Arabeyyat OS, Awadallah MA, Alsamarraie L, Abu-Doush I, Al-Betar MA (2017) Multiple-reservoir scheduling using β-hill climbing algorithm. J Intell Syst 28(4):559–570. https://doi.org/10.1515/jisys-2017-0159

Alweshah M (2014) Firefly algorithm with artificial neural network for time series problems. Res J Appl Sci Eng Technol 7(19):3978–3982

Alweshah M (2018) Construction biogeography-based optimization algorithm for solving classification problems. Neural Comput Appl 29(4):1–10

Alweshah M, Abdullah S (2015) Hybridizing firefly algorithms with a probabilistic neural network for solving classification problems. Appl Soft Comput 35:513–524

Alweshah M, Hammouri AI, Tedmori S (2017a) Biogeography-based optimisation for data classification problems. Int J Data Min Model Manag 9(2):142–162

Alweshah M, Rashaideh H, Hammouri AI, Tayyeb H, Ababneh M (2017b) Solving time series classification problems using support vector machine and neural network. Int J Data Anal Tech Strat 9(3):237–247

Alyasseri ZAA, Khader AT, Al-Betar MA (2017a) Optimal electroencephalogram signals denoising using hybrid β-hill climbing algorithm and wavelet transform. In: Proceedings of the international conference on imaging, signal processing and communication, ACM, pp 106–112

Alyasseri ZAA, Khader AT, Al-Betar MA, Abualigah LM (2017b) ECG signal denoising using β-hill climbing algorithm and wavelet transform. In: 2017 8th International conference on information technology (ICIT), IEEE, pp 96–101

Alyasseri ZAA, Khader AT, Al-Betar MA, Awadallah MA (2018) Hybridizing β-hill climbing with wavelet transform for denoising ECG signals. Inf Sci 429:229–246

Alzaidi AA, Ahmad M, Doja M, Al Solami E, Beg MS (2018) A new 1D Chaotic map and β-hill climbing for generating substitution-boxes. IEEE Access 6:55405–55418

Anagaw A, Chang YL (2019) A new complement naïve Bayesian approach for biomedical data classification. J Ambient Intell Human Comput 10(10):3889–3897

Araghi LF, Khaloozade H, Arvan MR (2009) Ship identification using probabilistic neural networks (PNN). Proc Int Multiconf Eng Comput Sci 2:18–20

Berrar DP, Downes CS, Dubitzky W (2002) Multiclass cancer classification using gene expression profiling and probabilistic neural networks. In: Biocomputing 2003, World Scientific, pp 5–16

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 35(3):268–308

Boveiri HR, Khayami R, Elhoseny M, Gunasekaran M (2019) An efficient swarm-intelligence approach for task scheduling in cloud-based internet of things applications. J Ambient Intell Human Comput 10(9):3469–3479

Brownlee J (2016) Supervised and unsupervised machine learning algorithms. Mach Learn Mastery 16(03)

Cpałka K, Rebrova O, Nowicki R, Rutkowski L (2013) On design of flexible neuro-fuzzy systems for nonlinear modelling. Int J Gen Syst 42(6):706–720

Craddock R, Warwick K (1996) Multi-layer radial basis function networks. an extension to the radial basis function. In: Proceedings of international conference on neural networks (ICNN'96), vol 2. IEEE, pp 700–705

Davis WL, Warren L (2006) Enhancing pattern classification with relational fuzzy neural networks and square BK-products. Florida State University, Tallahassee

Dorigo M, Stützle T (2010) Ant colony optimization: overview and recent advances. Springer, Boston, pp 227–263

El-Bouri A (2012) An investigation of initial solutions on the performance of an iterated local search algorithm for the permutation flowshop. In: 2012 IEEE Congress on evolutionary computation, IEEE, pp 1–5

Faris H, Aljarah I, Mirjalili S (2016) Training feedforward neural networks using multi-verse optimizer for binary classification problems. Appl Intell 45(2):322–332

Friedman N, Geiger D, Goldszmidt M (1997) Bayesian network classifiers. Mach Learn 29(2–3):131–163

Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68

Gorunescu F (2011) Data mining: concepts, models and techniques, vol 12. Springer Science & Business Media, Berlin

Hu L, Qin L, Mao K, Chen W, Fu X (2016) Optimization of neural network by genetic algorithm for flowrate determination in multipath ultrasonic gas flowmeter. IEEE Sens J 16(5):1158–1167

Iliou T, Anagnostopoulos CN (2010) SVM-MLP-PNN classifiers on speech emotion recognition field—a comparative study. In: 2010 Fifth international conference on digital telecommunications, IEEE, pp 1–6

Juang BH, Hou W, Lee CH (1997) Minimum classification error rate methods for speech recognition. IEEE Trans Speech Audio process 5(3):257–265

Khan A, Shah R, Imran M, Khan A, Bangash JI, Shah K (2019) An alternative approach to neural network training based on hybrid bio meta-heuristic algorithm. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-019-01373-4

Kwigizile V, Selekwa MF, Mussa RN (2004) Highway vehicle classification by probabilistic neural networks. In: FLAIRS conference, pp 664–669

LeCun Y, Boser BE, Denker JS, Henderson D, Howard RE, Hubbard WE, Jackel LD (1990) Handwritten digit recognition with a back-propagation network. In: Advances in neural information processing systems, pp 396–404

Manimala K, Selvi K (2007) Power quality disturbances classification using probabilistic neural network. In: International conference on computational intelligence and multimedia applications (ICCIMA 2007), vol 1. IEEE, pp 207–211

Mao KZ, Tan KC, Ser W (2000) Probabilistic neural-network structure determination for pattern classification. IEEE Trans Neural Netw 11(4):1009–1016

Mika S, Ratsch G, Weston J, Scholkopf B, Mullers KR (1999) Fisher discriminant analysis with kernels. In: Neural networks for signal processing IX: proceedings of the 1999 IEEE signal processing society workshop (Cat. No. 98TH8468), *Ieee*, pp 41–48

Ren K, Qu J (2014) Identification of shaft centerline orbit for wind power units based on hopfield neural network improved by simulated annealing. Mathematical problems in engineering 2014

Specht DF (1990) Probabilistic neural networks. Neural Netw 3(1):109–118

Specht DF (1991) A general regression neural network. IEEE Trans Neural Netw 2(6):568–576

Sweeney WP Jr, Musavi MT, Guidi JN (1994) Classification of chromosomes using a probabilistic neural network. Cytom J Int Soc Anal Cytol 16(1):17–24

Tairan N, Zhang Q (2010) Population-based guided local search: some preliminary experimental results. In: IEEE congress on evolutionary computation, IEEE, pp 1–5

Wasserman P (1993) Advanced methods in neural networks

Xie X, Li Y, Zhou H, Zheng Y (2012) Variable neighborhood search based multi-objective dynamic crane scheduling. In: Proceedings of 2012 international conference on measurement, information and control, IEEE, vol 1, pp 457–460

Yampolskiy RV, Govindaraju V (2008) Behavioural biometrics: a survey and classification. Int J Biometr 1(1):81–113

Yan C, Xie H, Chen J, Zha Z, Hao X, Zhang Y, Dai Q (2018) A fast Uyghur text detector for complex background images. IEEE Trans Multimedia 20(12):3389–3398

Yan C, Li L, Zhang C, Liu B, Zhang Y, Dai Q (2019a) Cross-modality bridging and knowledge transferring for image understanding. IEEE Trans Multimedia

Yan C, Tu Y, Wang X, Zhang Y, Hao X, Zhang Y, Dai Q (2019b) STAT: spatial-temporal attention mechanism for video captioning. IEEE Trans Multimedia

Zarndt F (1995) A comprehensive case study: an examination of machine learning and connectionist algorithms. PhD thesis, Brigham Young University. Department of Computer Science

Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173