Neural Process Lett (2008) 27:153–162
DOI 10.1007/s11063-007-9066-5

Novel Approaches to Probabilistic Neural Networks Through Bagging and Evolutionary Estimating of Prior Probabilities

Vasileios L. Georgiou · Philipos D. Alevizos · Michael N. Vrahatis

Published online: 23 December 2007

Abstract In this contribution, novel approaches are proposed for the improvement of the performance of Probabilistic Neural Networks as well as the recently proposed Evolutionary Probabilistic Neural Networks. The Evolutionary Probabilistic Neural Network's matrix of spread parameters is allowed to have different values in each class of neurons, resulting in a more flexible model that fits the data better and Particle Swarm Optimization is also employed for the estimation of the Probabilistic Neural Networks's prior probabilities of each class. Moreover, the bagging technique is used to create an ensemble of Evolutionary Probabilistic Neural Networks in order to further improve the model's performance. The above approaches have been applied to several well-known and widely used benchmark problems with promising results.

Keywords Probabilistic neural network · Particle swarm Optimization · Spread parameters · Prior probabilities · Bagging

Mathematics Subject Classification (2000) 62F40 · 62H30 · 62M45 · 68T10 · 82C32

V. L. Georgiou (✉) · P. D. Alevizos · M. N. Vrahatis
Department of Mathematics, Computational Intelligence Laboratory (CI Lab), University of Patras, Patras 26110, Greece
e-mail: vlg@math.upatras.gr

P. D. Alevizos
e-mail: philipos@math.upatras.gr

M. N. Vrahatis
e-mail: vrahatis@math.upatras.gr

V. L. Georgiou · P. D. Alevizos · M. N. Vrahatis
University of Patras Artificial Intelligence Research Center (UPAIRC), University of Patras, Patras 26110, Greece

1 Introduction

One of the models that have been applied to a wide variety of real-world problems is the probabilistic neural network (PNN) [29] that is used for classification tasks in several fields of science [8,13,15,16,32]. It is closely related to the well-known discriminant analysis using kernel functions and the final classification is obtained using the Bayes decision rule [14].

In our approach, several new ideas are proposed in order to improve the performance of the probabilistic neural network as well as the evolutionary probabilistic neural network (EPNN) [10] and other variations of PNNs. We have applied these novel approaches to EPNNs but they can also be applied to other PNN variants. At first, an expansion of the EPNN's spread parameters' matrix Σ is proposed which allows the kernels of each class of neurons to have a different matrix. Also the weighting of the kernels' outputs which was conducted by the prior probabilities of each class, is now achieved by particle swarm optimization (PSO) algorithm. Finally, the bagging technique is used for the further improvement of the performance and the robustness of EPNN [4].

For completeness purposes, let us briefly present some background material. The PNN was introduced by Specht [29] as a new neural network type, which is closely related to kernel discriminant analysis [14]. In fact, the model that Specht introduced is the neural network implementation of kernel discriminant analysis which incorporates the Bayes decision rule and the nonparametric density function estimation of a population according to Parzen.

The training procedure of the PNN is quite simple and requires only a single pass of the patterns of the training dataset $\mathcal{T}$, which results in a very short training time. In fact, the training procedure is just the construction of the PNN from the available data. The structure of the PNN always has four layers; the *input layer*, the *pattern layer*, the *summation layer*, and the *output layer* [10,29]. An input feature vector, $X \in \mathbb{R}^p$, is applied to the p input neurons and is passed to the pattern layer. The pattern layer is fully interconnected with the input layer and is organized into K groups, where K is the number of classes present in the data set. Each group of neurons in the pattern layer consists of N_k neurons, where N_k is the number of training vectors that belong to class k, $k = 1, 2, \ldots, K$. The ith neuron in the kth group of the pattern layer computes its output using a Gaussian kernel function of the form,

$$f_{ik}(X) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X - X_{ik})^T \Sigma_k^{-1} (X - X_{ik})\right), \tag{1}$$

where $X_{ik} \in \mathbb{R}^p$ is the center of the kernel and Σ_k is the matrix of spread (smoothing) parameters of the kernel function. The summation layer consists of K neurons and estimates the conditional probabilities of each class,

$$G_k(X) = \sum_{i=1}^{N_k} \pi_k f_{ik}(X), \quad k \in \{1, 2, \ldots, K\}, \tag{2}$$

where π_k is the prior probability of class k, $\sum_{k=1}^{K} \pi_k = 1$. So a vector X is classified to the class that has the maximum output of its summation neurons.

A faster version of the PNN can be obtained by using only a part of the training data set $\mathcal{T}$ instead of the whole training data set. Such a training set $\mathcal{L}$ can be obtained either by randomly sampling from the available data or by finding some "representatives" of the training data through a clustering technique. In our approach, we identified an adequate number of informative representatives (mean centers) from each class by using the K-medoids clustering algorithm [17] on the training data of each class. The classification accuracy of a

PNN is influenced by the spread parameters of its kernels. For the estimation of promising values of the spread parameters, PSO algorithm has been employed [10]. PSO is a stochastic, population-based optimization algorithm [18] and its concept is to exploit a population of individuals to synchronously probe promising regions of the search space. In this context, the population is called a *swarm* and the individuals (i.e., the search points) are called *particles*. Let us assume a d-dimensional search space $\mathcal{S} \subset \mathbb{R}^d$ where each one of the NP particles Z_i, $(i = 1, 2, \ldots, NP)$ moves with an adaptable velocity V_i within the search space and retains in a memory the best position BP_i it has ever encountered. At every iteration, their best positions are communicated between the particles of the swarm and the overall best particle BP_g is selected. The particles at iteration t are updated according to the following equations

$$V_i^{t+1} = \chi \left[V_i^t + c_1 r_1 \left(BP_i^t - Z_i^t \right) + c_2 r_2 \left(BP_g^t - Z_i^t \right) \right], \tag{3}$$

$$Z_i^{t+1} = Z_i^t + V_i^{t+1}, \tag{4}$$

where $i = 1, 2, \ldots, NP$; c_1 and c_2 are two positive constants called *cognitive* and *social* parameter, respectively; χ is a parameter called *constriction coefficient*; and r_1, r_2, are random vectors uniformly distributed within [0, 1] [5]. So, all vector operations in Eqs. 3 and 4 are performed componentwise and the best positions are then updated according to the equation

$$BP_i^{t+1} = \begin{cases} Z_i^{t+1}, & \text{if } f\left(Z_i^{t+1}\right) < f\left(BP_i^t\right), \\ BP_i^t, & \text{otherwise,} \end{cases}$$

where f is the objective function to be minimized by PSO. In our case f declares the misclassification proportion of the validation set. The constriction coefficient is derived analytically through the formula

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \tag{5}$$

for $\varphi > 4$, where $\varphi = c_1 + c_2$, and $\kappa = 1$, according to the stability analysis of Clerc and Kennedy [5,31]. Nevertheless, the particles are always bounded in the search space $\mathcal{S}$. For details we refer to [23,24].

In addition, to further improve the classification accuracy and the robustness of the classification model, we incorporated the bagging technique [4,19]. Let $\mathcal{L}$ be a training set which consists of $\{(X^{(i)}, Y^{(i)}), i = 1, 2, \ldots, N_{\text{train}}\}$, where $X^{(i)} \in \mathbb{R}^p$. In a classification task, where an unknown vector $X \in \mathbb{R}^p$ must be classified into one of K predefined classes, the $Y^{(i)}$'s take values in $\{1, 2, \ldots, K\}$. Regardless of the model being used, a classifier $\Phi(X, \mathcal{L})$ is constructed. Suppose we have a sequence of training sets $\mathcal{L}_m$ each consisting of N_{train} independent observations from the same underlying distribution as $\mathcal{L}$. The task is to construct a better classifier than $\Phi(X, \mathcal{L})$ using the set $\{\mathcal{L}_m\}$.

It is typical that only a single training set is available so in order to obtain the appropriate samples, bootstrap samples are drawn from $\mathcal{L}$. Each bootstrap sample consists of N_{train} cases and is drawn at random from $\mathcal{L}$ with replacement. Some cases may not be selected at all and others may be selected multiple times. For each bootstrap sample $\mathcal{L}_m$, a classifier $\Phi(X, \mathcal{L}_m)$ is constructed. In order to aggregate all the $\{\Phi(X, \mathcal{L}_m)\}$, a voting procedure is employed. Let N_k be the number of times that the classifiers have voted for class k, $k = 1, 2, \ldots, K$. The final classification is made by a max-rule of $\{N_k\}$. In other words, the final classifier is $\Phi_B(X, \mathcal{L}) = \text{argmax}_k(N_k)$ meaning that a majority voting scheme is employed. The above procedure is named "boostrap `aggregating`" and the acronym `bagging` is used for it.

2 The Proposed Approach

One drawback of PNN's performance is the need to estimate a promising value for the spread parameter of the network's kernel function. This is usually obtained by a trial-and-error procedure, although several alternative methods have been proposed in the literature [10–12]. The Evolutionary PNN, proposed in [10], employs the PSO algorithm for the selection of the appropriate spread parameters. The selection of spread parameters' matrix Σ is accomplished by the minimization of the misclassification proportion of the training or validation set with respect to Σ which is allowed to be a diagonal matrix whose entries may or may not be equal. The former case is referred to as *homoscedastic* PNN while the latter is referred as *heteroscedastic* PNN.

In this contribution, we expand the above concept in a way that every group of neurons in the pattern layer of the PNN, has its own matrix $\Sigma_k,\ k = 1, 2, \ldots, K$. Now, the matrix of the spread parameters of the neurons' kernels of class k can differ from the ones of class l, for $k \neq l$. This results in a better adaptation of the kernels to the data, since the density of the data of one class could have a different shape from the density of the data of the other classes. Therefore, the matrix of spread parameters of our proposed approach will have the following form $\Sigma_k = \text{diag}(\sigma_{1k}^2, \sigma_{2k}^2, \ldots, \sigma_{pk}^2),\ k = 1, 2, \ldots, K$ and the optimization dimension of PSO will be $d = k\,p$.

Another novelty is that besides the optimization of the PNN's spread parameters, PSO algorithm is also employed for the better estimation of the prior probabilities π_k. The prior probabilities are neither estimated from the training data set nor set arbitrarily but they are also included in the PSO optimization of the PNN together with the spread parameters. This allows the PNN to have an even better adaptation and fit the data properly. It allows the model to have an even higher degree of freedom of fitting the data. Of course, π_k's are constrained to $\sum \pi_k = 1$. In fact, π_k's function as a new way of weighting the output of each kernel function of the PNN.

As it was mentioned in Sect. 1, a way to improve the performance of a classification model is to create an ensemble of m classifiers and use a majority voting system for the final classification. The technique that was adapted in our contribution is the bagging technique. A sequence of bootstrap samples is created from the training set and from each one of the samples an EPNN is constructed. An unknown vector X is classified to the class that has collected the more votes of the EPNNs. This technique accomplishes to construct a more robust and stable classifier. In conclusion, our approach consists of the following steps:

Step 1: Apply the K-Medoids algorithm on each class of the training set $\mathcal{T}$ to obtain the clustered training set $\mathcal{L}$.
Step 2: Construct m PNNs using $\mathcal{L}$ (Σ_k and π_k randomly initialized).
Step 3: For $i = 1, 2, \ldots, m$; Estimate Σ_k and π_k of PNN_i by PSO, (fitness function: misclassification proportion on the whole training set $\mathcal{T}$).
Step 4: Compute final classification from m PNNs using a majority voting scheme.

3 Experimental Results

The aforementioned novel approaches of the EPNN were applied to six randomly chosen benchmark problems from UCI data repository [2] from several fields of science with encouraging results. In fact, the data sets were downloaded from Proben1 database [25] since the format that the data sets are offered in Proben1, can be easily used in machine learning models. We have the following benchmark data sets:

(1) The first data set that was applied to the proposed models is the *Wisconsin breast cancer database* (WBCD) and its aim is to predict whether a breast tumour is benign or malignant [20]. There are nine continuous attributes based on cell descriptions gathered by microscopic examination and 699 instances.
(2) On the second data set (*Card*), the aim is to predict the approval or non-approval of a credit card to a customer [26]. There are 51 attributes which are unexplained for confidential reasons and the number of observations is 690.
(3) The aim of the third data set, namely *The Pima Indians Diabetes Data Set*, is to predict the onset of diabetes, therefore, there are two classes [28]. The input features are the diastolic blood pressure; triceps skin fold thickness; plasma glucose concentration in a glucose tolerance test; and diabetes pedigree function. These eight inputs are all continuous without missing values and there are 768 instances.
(4) The fourth data set, *Glass*, consists of 214 instances and its aim is to classify a piece of glass into six different types, namely float processed or non float processed building windows, vehicle windows, containers, tableware and heat lamps [7]. The classification is based on nine inputs, which are the percentages of content on eight different elements plus the refractive index and this task is motivated by forensic needs in criminal investigation.
(5) In the fifth data set, namely *Heart Disease*, the aim is to predict whether at least one of the four major vessels of the heart is reduced in diameter by more than 50%, so there are two classes [6]. The 35 attributes of the 920 patients are age, sex, smoking habits, subjective patient pain descriptions and results of various medical examinations such as blood pressure and cardiogram.
(6) The last data set is the *Horse* data set and its task is to predict the fate of a horse that has a colic [25]. The prediction whether the horse would survive, would die or would be euthanized is based on 58 inputs of a veterinary examination of the horse and there are 364 instances.

So, the models that the aforementioned benchmark problems were applied to are the following ones:

(a) *PNN*: At first, we have the original PNN where for the selection of the spread parameter σ an exhaustive search in the search space is executed and in order to be fair with the rest of the models, the number of functional evaluations that were computed for the selection of σ is equal to the corresponding one of the EPNNs.
(b) *CL.PNN*: The same model (PNN) is used but the clustered training set is employed instead of the whole training set.
(c) *GGEE.PNN*: A variation of the PNN that is proposed by Gorunescu *et al.* [12] has also been used.
(d) *Hom.EPNN*: The homoscedastic EPNN [10] was applied to these data sets in order to compare the obtained results with the corresponding ones of the proposed approaches. The Hom.EPNN utilizes the whole training data set for the construction of the PNN's pattern layer, which means that the demand in memory allocation and computational power is large
(e) *Het.EPNN*: The heteroscedastic version of the EPNN [10] that utilizes the whole training set was also applied to the aforementioned problems.
(f) *CL.Hom.EPNN*: Another model used in the comparison was the Hom.EPNN where only the clustered training set was used for PNN's construction.
(g) *CL.Het.EPNN*: Also, the clustered version of Het.EPNN was applied to the benchmark problems.

(h) *B.EPNN*: In order to examine whether the weighting of pattern neurons by PSO offers a remarkable increase in the performance of the EPNN, the Bagging EPNN (B.EPNN) does not incorporate the PSO weighting but it incorporates bagging, clustered training set and generalized spread parameters' matrix.
(i) *B.P.EPNN*: The proposed approach (Bagging EPNN with Prior evolutionary estimation) where the EPNN incorporates all the aforementioned ways of improvement: Bagging, clustered training set, generalized spread parameters' matrix and prior estimating by PSO.

The B.EPNN and B.P.EPNN are the new approaches presented in this paper.

The aforementioned models were applied to these six benchmark data sets using 10 times 10-fold cross-validation. Each training data set was applied to all the models except B.EPNN and B.P.EPNN for 5 times and the median of the test sets' classification accuracy was recorded in order to eliminate the bias of the accuracy due to the random initialization of PSO. In PSO, the default parameter values, $c_1 = c_2 = 2.05$, $\chi = 0.729$, were used [5]. In the Hom.EPNN, the number of particles was set to $NP = 5$ and PSO was evolved for 50 generations in order to find the value of σ that minimized the classification error on the whole training set. Moreover, in the Het.EPNN and in both the B.EPNN and the B.P.EPNN, 10 particles were evolved for 100 generations. For the bagging EPNNs, 11 bootstrap samples were drawn from each clustered training data set and based on these bootstrap samples, an ensemble of 11 EPNNs was constructed and the final classification was obtained by a majority voting procedure.

The mean generalization ability (classification accuracy) of each problem's test sets along with its standard deviation (SD) is reported in Table 1 for the PNN, the CL.PNN, the Hom.EPNN and Het.EPNN utilizing the whole training set and the corresponding clustered training set (CL.Hom.EPNN and CL.Het.EPNN respectively), the GGEE.PNN and the proposed Bagging EPNNs with and without the PSO weighting of pattern layer's neurons (B.P.EPNN and B.EPNN respectively). The corresponding training CPU times are presented in Table 2 for all the models. For the bagging EPNN variants, the total CPU time for the training of all the 11 models is presented.

The normality assumption was met in all the runs according to the Kolmogorov–Smirnov normality test; thus to investigate the statistical significance of the results, a corrected resampled t-test [3,21] was conducted to compare the mean classification accuracies of the best proposed model and the best of the remaining models for each dataset. The highest mean classification accuracy is shown in boldface letters and the statistically superior mean performance is depicted in a box.

From Table 1, we can see that the proposed approach obtained a statistically superior mean performance in two out of six problems (Breast Cancer, Diabetes). Moreover, in four out of the six cases, the Het.EPNN achieved a superior mean performance compared with the B.EPNN and B.P.EPNN but in Card and Heart the difference is quite small. Nevertheless, in Horse and especially in Glass there was a considerable difference between the mean performances. We should recall that in Horse there are three classes of data and in Glass six classes. Consequently, we can say that as the number of classes in the data set increases, the original EPNN achieved higher performance. This is possibly due to the fact that in multiclass problems, the clustering technique did not extract "reliable" centers or PSO could not find promising values for all the spread parameters. It should also be noted that these two benchmark problems do not have a large number of instances, so it may be better to use the Het.EPNN. Moreover, in all cases where we have a binary classification problem, the weighting of pattern neurons by PSO obtained a slightly better performance compared to the B.EPNN where the prior probabilities π_k were estimated from the relative frequencies of the classes. Another remark is that the performance of the clustered versions of Hom.EPNN and

Table 1 Classification accuracy of the models

Dataset Model	Breast Cancer		Card		Diabetes	
	Mean	SD	Mean	SD	Mean	SD
PNN	95.79	0.25	82.10	0.76	65.08	**0.05**
CL.PNN	91.91	0.84	80.49	0.66	65.08	0.05
GGEE.PNN	96.39	0.20	84.31	0.63	69.43	0.68
Hom.EPNN	95.82	0.28	85.35	0.38	67.67	0.88
Het.EPNN	95.32	0.57	**87.67**	0.51	69.37	0.80
CL.Hom.EPNN	90.50	1.58	82.02	1.15	65.35	0.48
CL.Het.EPNN	87.89	1.74	85.20	0.97	69.30	1.59
B.EPNN	96.85	0.46	86.64	0.51	71.00	1.02
B.P.EPNN	**97.17**	**0.16**	86.83	**0.34**	**71.22**	1.00
Dataset	Glass		Heart		Horse	
Model	Mean	SD	Mean	SD	Mean	SD
PNN	33.25	3.40	79.23	0.48	64.63	0.72
CL.PNN	41.74	1.64	79.84	0.71	60.23	1.67
GGEE.PNN	50.07	**1.44**	80.68	0.52	61.97	1.23
Hom.EPNN	68.52	1.55	81.50	**0.27**	66.54	0.79
Het.EPNN	**75.36**	1.77	**82.60**	0.40	**68.48**	0.97
CL.Hom.EPNN	54.04	3.61	79.96	0.56	61.81	0.77
CL.Het.EPNN	47.25	2.75	77.62	1.16	58.89	1.51
B.EPNN	54.91	3.98	82.28	0.62	66.47	1.40
B.P.EPNN	52.74	4.13	82.35	1.05	66.16	1.56

Table 2 CPU time for the training of the models (seconds)

Dataset	Breast cancer		Card		Diabetes	
Model	Mean	*SD*	Mean	SD	Mean	SD
PNN	42.09	0.66	182.01	7.88	49.58	0.38
CL.PNN	**0.08**	**0.002**	**0.23**	**0.004**	**0.10**	0.004
GGEE.PNN	1.52	0.17	5.46	0.06	1.87	0.03
Hom.EPNN	89.12	1.07	266.10	74.56	101.17	0.48
Het.EPNN	171.78	1.07	521.60	142.74	195.27	0.92
CL.Hom.EPNN	0.16	0.02	0.49	0.06	0.18	**0.003**
CL.Het.EPNN	0.32	0.06	0.66	0.14	0.36	0.008
B.EPNN	82.78	8.86	309.85	1.88	106.42	0.92
B.P.EPNN	90.01	0.92	309.73	2.62	106.24	0.81
Dataset	Glass		Heart		Horse	
Model	Mean	SD	Mean	*SD*	Mean	SD
PNN	3.82	0.02	207.99	45.27	29.29	0.88
CL.PNN	**0.11**	**0.002**	**0.32**	**0.02**	**0.18**	**0.01**
GGEE.PNN	3.66	0.08	6.47	0.92	5.46	0.14
Hom.EPNN	9.16	0.65	223.28	4.28	76.10	7.97
Het.EPNN	17.21	0.76	438.10	6.82	169.92	23.39
CL.Hom.EPNN	0.21	0.01	0.67	0.08	0.37	0.02
CL.Het.EPNN	0.45	0.01	1.37	0.16	0.76	0.03
B.EPNN	29.03	0.14	394.49	5.93	126.76	2.02
B.P.EPNN	28.30	0.11	393.22	4.95	123.03	2.24

Het.EPNN is inferior in all cases compared to the original versions that utilize the whole training set. Although the clustered versions are much faster in CPU time, the performance is quite poor compared to the whole versions.

We should note that the bagging technique and generally ensemble techniques increase the training and response time since we have to train 11 models in our case and not only one. But, the number of neurons in the pattern layer of the Het.EPNN was about 20 times larger than the corresponding B.P.EPNN's, which has as a result a much faster model both in training and response time. On the other hand, if we use the clustered training set for the Het.EPNN., the performance is worse than the Het.EPNN and the B.P.EPNN, so we see that the bagging technique helps to improve the performance of the CL.Het.EPNN. So, the proposed approaches do not claim that they can achieve higher performance than EPNN, but they obtain almost similar results with shorter training time in five out of six cases since they utilize properly the clustered training set. In Table 2 the mean training CPU times are reported for all the aforementioned models. The proposed approach demands less CPU time than Het.EPNN in five out of six cases and in some cases such as the Breast Cancer problem the B.P.EPNN training time is the half of the corresponding Het.EPNN training time.

Moreover, in order to give an insight to the classification ability of the proposed approaches compared to any method and not just PNN variants, we have collected the best reported classification accuracies (generalization accuracy) of the aforementioned benchmark data sets that have been achieved by any model and sampling technique after an extensive search in the literature. The results are reported in Table 3.

For the Br.Cancer data set, the best result has been achieved by a Multilayer Perceptron trained by a hybrid algorithm that combines the Levenberg–Marquardt algorithm and Genetic algorithms [1]. Next, the best classification accuracy for the Card problem has been achieved by a immune network ensemble where each antigen represents a non-linear projection (NLP) [9]. The best result for Diabetes dataset has been achieved by an AdaBoost NN and an Immune Network Ensemble with random subspace method (RSM) antigen [1]. For the Glass problem, a sequential multi-category classifier (SMC) using radial basis function (RBF) networks gave the best result [30]. The best generalization accuracy ever for the Heart problem has been achieved by a MLP that its weights and connections (architecture) were commonly evolved by a Genetic Algorithm [22]. Finally, for the Horse dataset, the best result has been achieved by a NN trained by a Genetic Algorithm and a feature selection procedure [27]. As we can see, the performance of the proposed approach is more or less close to the best model's performance on each dataset, besides the Glass where there is a great superiority of the SMC-RBF network.

Table 3 Comparison of classification accuracies for the best ever model and the proposed approaches

Dataset	B.EPNN		B.P.EPNN		Overall best		
	Mean	SD	Mean	SD	Mean	SD	Model
Br.cancer	98.07	0.46	98.95	0.38	99.98	0.11	GALM [1]
Card	88.09	1.37	87.51	1.13	90.30	–	NLP [9]
Diabetes	73.70	2.38	73.23	3.14	80.00	–	AdaB. &RSM [9]
Glass	64.76	–	57.14	–	78.09	–	SMC-RBF [30]
Heart1	80.52	1.04	80.43	0.87	84.23	3.43	GA MLP [22]
Horse1	72.53	1.72	71.98	1.49	80.22	–	GenAlg NN [27]

4 Conclusion

Several approaches are proposed in this contribution in order to improve the performance of PNNs as well as EPNNs and other PNN variations. An expansion of the EPNN's spread matrix is proposed so that the new model will achieve a better fit to the data. In addition, a new way of weighting the neurons' outputs is achieved by PSO besides the typical way where the prior probabilities are estimated from the relative frequencies of their classes. Moreover, a further improvement is achieved by the bagging technique. From the experimental results, it is obvious that the proposed approach achieves higher generalization ability than the original PNN and Gorunescu et al. PNN and obtain almost similar results with EPNN especially in binary classification problems. However, the proposed B.P.EPNN has lower computing and memory requirements due to its smaller pattern layer, so it can be used instead of EPNN in large data sets and in binary classification problems since it can achieve similar performance using the clustered training set, something that the clustered Het.EPNN can not.

Acknowledgements We would like to thank the referees for their valuable comments, remarks and suggestions that greatly improved our paper. We also thank the European Social Fund (ESF), Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the Program IRAKLEITOS for funding the above work.

References

1. Alba E, Chicano JF (2004) Training neural networks with GA Hybrid Algorithms. In: Genetic and evolutionary computation—GECCO 2004, genetic an:d evolutionary computation conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part I, vol. 3102 of Lecture notes in computer science. pp 852–863
2. Asuncion A, Newman D (2007) UCI machine learning repository. http://www.ics.uci.edu/~mlearn/MLRepository.html
3. Bouckaert RR, Frank E (2004) Evaluating the replicability of significance tests for comparing learning algorithms. In: Dai H, Srikant R Zhang C (eds) Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining, LNAI 3056, Sydney, Australia, pp 3–12
4. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
5. Clerc M, Kennedy J (2002) The particle swarm—explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73
6. Detrano R, Janosi A, Steinbrunn W, Pfisterer M, Schmid J, Sandhu S, Guppy K, Lee S, Froelicher V (1989) International application of a new probability algorithm for the diagnosis of coronary artery disease. Am J Cardiol 64:304–310
7. Evett IW, Spiehler EJ (1987) Rule induction in forensic science. Technical report, Central Research Establishment, Home Office Forensic Science Service
8. Ganchev T, Tasoulis DK, Vrahatis MN, Fakotakis N (2007) Generalized locally recurrent probabilistic neural networks with application to text-independent speaker verification. Neurocomputing 70(7–9):1424–1438
9. Garcia-Pedrajas N, Fyfe C (2007) Immune network based ensembles. Neurocomputing 70:1155–1166
10. Georgiou VL, Pavlidis NG, Parsopoulos KE, Alevizos PD, Vrahatis MN (2006) New self–adaptive probabilistic neural networks in bioinformatic and medical tasks. Int J Artif Intel Tools 15(3):371–396
11. Gorunescu M, Gorunescu F, Ene M, El-Darzi E (2005) A heuristic approach in hepatic cancer diagnosis using a probabilistic neural network-based model. In: Proceedings of the international symposium on applied stochastic models and data analysis. Brest, France, pp 1016–1024
12. Gorunescu F, Gorunescu M, Revett K, Ene M (2007) A hybrid incremental Monte Carlo searching technique for the "Smoothing" parameter of probabilistic neural networks. In: Proceedings of the international conference on knowledge engineering, principles and techniques, KEPT 2007. Cluj-Napoca, Romania, pp 107–113
13. Guo J, Lin Y, Sun Z (2004) A novel method for protein subcellular Localization based on boosting and probabilistic neural network. In: Proceedings of the 2nd Asia-Pacific bioinformatics conference (APBC2004). Dunedin, New Zealand, pp 20–27

14. Hand JD (1982) Kernel discriminant analysis. Research Studies Press, Chichester
15. Holmes E, Nicholson JK, Tranter G (2001) Metabonomic characterization of genetic variations in toxicological and metabolic responses using probabilistic neural networks. Chem Res Toxicol 14(2):182–191
16. Huang CJ (2002) A performance analysis of cancer classification using feature extraction and probabilistic neural networks. In: Proceedings of the 7th Conference on Artificial Intelligence and Applications. pp 374–378
17. Kaufman L, Rousseeuw PJ (1990) Finding groups in data: an introduction to cluster analysis. John Wiley and Sons, New York
18. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings IEEE International Conference on Neural Networks, vol IV. Piscataway, NJ, pp 1942–1948
19. Kotsiantis S, Pintelas P (2004) Combining bagging and boosting. Int J Comput Intel 1(4):324–333
20. Mangasarian OL, Wolberg WH (1990) Cancer diagnosis via linear programming. SIAM News 23:1–18
21. Nadeau C, Bengio Y (2003) Inference for the generalization error. Mach Learn 52(3):239–281
22. Neruda R, Slusny S (2007) Variants of memetic and hybrid learning of perceptron networks. In: DEXA Workshops. pp 158–162
23. Parsopoulos KE, Vrahatis MN (2002) Recent approaches to global optimization problems through particle swarm optimization. Nat Compu 1(2–3):235–306
24. Parsopoulos KE, Vrahatis MN (2004) On the computation of all global minimizers through particle swarm optimization. IEEE Trans Evolut Compu 8(3):211–224
25. Prechelt L (1994) Proben1: a set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe
26. Quinlan J (1987) Simplifying decision trees. Int J Man-Mach Studies 27(3):221–234
27. Sexton RS, Sikander NA (2001) Data mining using a genetic algorithm-trained neural network. Int J Intell Systems Account, Finance Manage 10:201–210
28. Smith J, Everhart J, Dickson GW, Knowles CB, Johannes JR (1988) Using the ADAP learning algorithm to forecast the onset of diabetes melitus. Johns Hopkins APL Technical Digest 10:262–266
29. Specht DF (1990) Probabilistic neural networks. Neural Netw 1(3):109–118
30. Suresh S, Sundararajan N, Saratchandran P (2007) A sequential multi-category classifier using radial basis function networks. Neurocomputing doi: 10.1016/j.neucom.2007.06.003
31. Trelea IC (2003) The particle swarm optimization algorithm: convergence analysis and parameter selection. Info Process Lett 85:317–325
32. Wang Y, Adali T, Kung S, Szabo Z (1998) Quantification and segmentation of brain tissues from MR Images: a probabilistic neural network approach. IEEE Trans Image Process 7(8):1165–1181