Musical Composer Identification through Probabilistic and Feedforward Neural Networks

Maximos A. Kaliakatsos-Papakostas, Michael G. Epitropakis, and Michael N. Vrahatis

Computational Intelligence Laboratory (CI Lab), Department of Mathematics,
University of Patras Artificial Intelligence Research Center (UPAIRC), University of Patras,
GR-26110 Patras, Greece
{maxk,mikeagn,vrahatis}@math.upatras.gr

Abstract. During the last decade many efforts for music information retrieval have been made utilizing Computational Intelligence methods. Here, we examine the information capacity of the Dodecaphonic Trace Vector for composer classification and identification. To this end, we utilize Probabilistic Neural Networks for the construction of a "similarity matrix" of different composers and analyze the Dodecaphonic Trace Vector's ability to identify a composer through trained Feedforward Neural Networks. The training procedure is based on classical gradient-based methods as well as on the Differential Evolution algorithm. An experimental analysis on the pieces of seven classical composers is presented to gain insight about the most important strengths and weaknesses of the aforementioned approach.

1 Introduction

The common approach to tackle the composer identification and classification problem is through music theory and involves symbolic or score element analysis, which is done by musicians or musicologists, such as musical motif recognition, note durations and musical interval analysis. The main domain in computer music data extraction is based on statistical refinement through the wave forms of musical pieces [1]. In the literature, significant results have been demonstrated for musical genre recognition [2,3], key signature [4,5], chord identification [5,6] and composer identification [7] among others. A vital question would be: which symbolic features of a musical score embody the required information for computer based composer identification?

Data extraction through score analysis has been proven to be useful for chordal analysis [8], epoch classification of musical pieces [9] as well as composer identification [1,10,11] among others.

In the paper at hand, we study the information capacity of a simple and compact score-based data extraction technique, the *Dodecaphonic Trace Vector* (DTV), for the Musical Composer Identification task (MCI). The DTV roughly represents densities of degrees in a diatonic major scale in a musical piece and is analogous to the *Pitch Chroma Profile* proposed in [6]. To this end, based on the DTV we examine the composer identification task, by firstly discovering similarities between composers through supervised classifiers, namely the Probabilistic Neural Networks, and secondly investigating the DTV representation identification capabilities through Feedforward Neural Networks.

C. Di Chio et al. (Eds.): EvoApplications 2010, Part II, LNCS 6025, pp. 411–420, 2010.

The rest of the paper is organized as follows. In Section 2, the dataset acquisition and the tools that have been applied to extract certain symbolic features of a score are described. In Section 3, we briefly demonstrate the probabilistic and feedforward neural networks, while Section 4 is devoted to the presentation of the methodology and the produced experimental results. The paper ends with concluding remarks and some pointers for future work.

2 Data Set and Data Extraction

This section is devoted to describe the methods and techniques utilized for data acquisition and refinement to capture the desired information from musical pieces.

To this end, we have collected a dataset consisting of 350 musical pieces, composed by seven classical music composers. Specifically, musical pieces of the composers Bach, Beethoven, Brahms, Chopin, Handel, Haydn and Mozart, have been collected in MIDI format from [12]. Fifty musical works have been collected from each composer. To comply with constrains regarding composition styles for different musical instruments, an effort has been made so that most of the works collected by each composer were already transcribed for piano and correspond to an almost uniform collection of musical forms. Furthermore, in order to formulate a scale-tolerant collection, an almost equal number of major scale and minor scale pieces have been included.

To process each work, a conversion to a more understandable file format had to be made. Hence, we have incorporated the MSQ tool to transform the MIDI file format to a simple text file format [13]. MSQ converts each note to a text symbol preserving information about duration, time onset, pitch and velocity. Through the above encoding each Pitch Height can be described using an integer, but the information on the identity of unison notes is not maintained, which has been named *enharmonic equivalence* [5,14], e.g. whether a number corresponds to $C\sharp$ or $D\flat$.

Here, we investigate whether a compact information index such as the *Dodecaphonic Trace Vector*, which is briefly described in the following paragraph, may incorporate sufficient information from a musical piece, to tackle the MCI task.

Dodecaphonic Trace Vector. A musical piece is like a journey, it begins and ends following a certain path. This path could be based on notes of either a certain scale (tonal music), or more than one scales, depending on the form of each piece and its composer's personal style. A collection of traces of this path can be created with the *Dodecaphonic Trace Vector* described below.

We consider a 12-dimensional vector, the *Chroma Profile* [15] vector of a musical piece denoted by $CP = (CP(1), CP(2), \ldots, CP(12))$, the elements of which can be defined by the following equation:

$$CP\big(n \bmod (12) + 1\big) = \sum_{n \bmod (12) \in M} 1,$$

where n denotes a note in the musical piece M. This equation simply states that the first element, $CP(1)$, is the summation of all the notes n that satisfy the equation $n \bmod (12) \equiv 0$, which has been characterized as *octave equivalence* [14]. In our example

$CP(1)$ is the summation of all Cs, while $CP(2)$ is the summation of all $C\sharp$ or $D\flat$ in any octave.

The norm of the CP vector of a musical piece depends on its length. A very short piece is expected to have less notes than a very long one, under similar circumstances, i.e. both being in the same tempo and composed by the same composer. To extract useful data concerning the composer out of the CP vector, regardless the length of the piece, a normalization has to be done. The normalized vector $N = (N(1), N(2), \ldots, N(12))$ can be defined by the following equation:

$$N(i) = \frac{CP(i)}{\max\limits_{1 \leqslant j \leqslant 12} CP(j)}, \qquad \text{for} \quad i \in \{1, 2, \ldots, 12\}.$$

Finally, to follow the traces of the melodies and the harmony (chord changes) of a composer, the utilized methods should be *key tolerant*, in a sense that the key of the composed piece is not essential. What is considered to be important information, is the way that a composer manipulates the *degrees* of the major or minor scale in his/hers compositions and the deriving expansions occurring on scale changes through out the piece. A procedure to capture that information of a collection of musical pieces is, first, to transpose all these pieces in the key of C major and then to calculate their normalized N vector. Additionally, the minor scale pieces were transposed in the key of A minor. Thus, the *Dodecaphonic Trace Vector* is defined by the normalized vectors of musical pieces transposed to the key of C major or A minor.

3 Classification Methods Tested

Our methodology is based on two simple steps and provides insights on the uniqueness of the DTV in the compositions of each composer. Firstly, a classification is initiated for the construction of a similarity matrix between the composers using a Probabilistic Neural Network and secondly, a Feedforward Neural Network is utilized to identity each composer from another. It has to be noted that the experimental results should be discussed with musical experts to further verify the uniqueness of the DTV for each composer.

For completeness purposes let us briefly describe the classification methods used for the composer identification task.

Probabilistic Neural Networks. Probabilistic Neural Networks (PNNs) introduced by Sprecht in 1990 [16] as a new neural network structure. PNNs are utilized to classify objects in a predetermined number of classes. PNNs are based on kernel discriminant analysis and incorporate the Bayes decision rule and a non-parametric density function [17]. A PNN's structure consists of four layers, the *input*, the *pattern*, the *summation* and the *output* layer [16,18]. To this end, a pattern vector, $x \in \mathbb{R}^p$ is applied to the p input neurons and propagates to the pattern layer. The pattern layer is fully interconnected with the input layer, organized in 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 belongs to class k where

$k = 1, 2, \ldots, K$. Hence, the i-th neuron in the k-th group of the pattern layer calculates its output utilizing a Gaussian kernel function defined by the following equation:

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

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

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

where π_k is the prior probability of each class k ($\sum_{k=1}^{K} \pi_k = 1$). Thus, a new pattern vector X, not included in the training set, is classified to the class which produces the maximum output of its summation neurons. It has to be noted here that PNN's classification ability is strongly affected by the value of the *smoothing parameter*, σ, that, roughly speaking, determines the range of each class and consequently its performance [16,19]. Probabilistic Neural Networks have been recently utilized in musical applications [3].

Feedforward Neural Networks. Although, many different models of Artificial Neural Network have been proposed, the Feedforward Neural Networks (FNNs) are the most common and widely used. FNNs have been successfully utilized to tackle difficult real-world problems [20,21,22,23]. For completeness purposes let us briefly describe their structure and their supervised training methodology.

A Feedforward Neural Network consists of simple processing units called neurons, which are arranged in layers, the *input*, the *hidden* layers and the *output* layer. The neurons between successive layers are fully interconnected, and each interconnection corresponds to a *weight*. The training process is an incremental adaptation of the connection weights which acquires the knowledge of the problem at hand and stores it to the network's weights. More specifically, consider a FNN, net, whose l-th layer contains N_l neurons, where $l = 1, 2, \ldots, M$. When a pattern appears in its *input* layer the signal deriving by the multiplication of the input and the weights of the first layer neurons is summed and passed through a nonlinear activation function such as the well known logistic function $f(x) = (1 + e^{-x})^{-1}$. Those signals are then propagated to the next layer of neurons, l, multiplied by the weights of the next neuron layer, and the procedure continues until the signal reaches the output layer. Hence, the j-th layer, can be described by $net_j^l = \sum_{i=1}^{N_{l-1}} w_{ij}^{l-1,l} y_i^{l-1}$, where $w_{ij}^{l-1,l}$ is the weight from the i-th neuron at the $(l-1)$ layer to the j-th neuron of the l-th layer, while $y_j^l = f(net_j^l)$ is the activation function of the j-th neuron in the l-th layer. The training procedure can be accomplished by *minimizing* the *error function* $E(w)$ which can be defined by the *sum of the squared differences* between the actual output of the FNN, denoted by $y_{j,p}^M$, and the desired output, denoted by $d_{j,p}$, relative to the appeared input,

$$E(w) = \frac{1}{2} \sum_{p=1}^{P} \sum_{j=1}^{N_M} \left(y_{j,p}^M - d_{j,p}\right)^2 = \sum_{p=1}^{P} E_p(w),$$

where $w \in \mathbb{R}^n$ is the vector of the network weights and P represents the number of patterns used in the training data set.

Traditional methods for minimizing the above error function require the estimation of the partial derivatives of the error function with respect to each weight. These are called *gradient-based descent methods* and information concerning the gradient are estimated by back propagating the error from the output to the first layer neurons using the *delta rule*, a procedure thoroughly described in [20,24]. Furthermore, novel methods proposing stochastic evolution of the weight vector do not require the computation of the gradient of the error function [21,22,25]. These methods are beneficial against gradient-based methods not only in terms of *computational cost* but also their global optimization characteristics enhance the training procedure which is less likely to be trapped in local minima [21,22,23].

Here we incorporate, both classical and stochastic training methodologies to enhance the training procedure. Hence, we have used three well–known and widely used classical methods, namely the Levenberg-Marquardt (LM) [26,27], the Resilient Backpropagation (Rprop) [28], and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) [29], as well as, we have used an evolutionary approach, namely the Differential Evolution method [25]. Due to space limitations, we are not in a position to briefly describe here the above methods. The interested reader is referred to [25,26,27,28,29].

4 Methodology and Experimental Results

This section demonstrates the experimental results and the experimental procedure utilized to tackle the Musical Composer Identification task. Hence, based on the DTV, the first step of our methodology is to calculate the similarity of the composers through the utilization of Probabilistic Neural Networks. To this end, we construct a similarity matrix for all musical composers. The construction of the similarity matrix is based on a simple method. To the best of our knowledge, this method is utilized for the first time and it is described below.

The set of the seven composers that we have collected, allows the construction of a square matrix with seven rows and seven columns. Each row and column represents one composer. Each entry of the matrix would be wished to produce an indication about the similarity between the composers corresponding to the respective row and column in dependence to the similarity along the rest of the composers. It has to be noted here that the diagonal elements of the aforementioned matrix, represents the similarity between a composer with himself, and would have no entry to be computed. A final and notable comment is that a consistent similarity matrix of this form should be symmetric.

We will demonstrate the PNNs role in the similarity matrix through a simple test case example. Let us consider that we want to construct the entries of the first column, which are related to Bach[1]. We utilize a PNN with six classes, one for each composer except Bach, and we train it by incorporating all pieces from the six composers, i.e. fifty pieces at the current data set. Afterwards, the fifty pieces of Bach are given to the PNN for classification. Thus, the utilized network will classify them to the classes of

[1] Since the first column is related to Bach, then the first row is related also to Bach.

the other six composers. In this way it can be assumed that the more pieces classified to a composer's class, the more similar his composition style is related to Bach's.

The construction of the first column of the similarity matrix is completed if we assign no value to the first row (representing the similarity between Bach and himself) and normalize the classified cases to probabilities per column by dividing every element of the first column by the total of the pieces classified (50). We observe that the sum of the column elements is one. To this end, Table 1 exhibits the similarity matrix obtained by the described procedure. The PNNs have been implemented in MATLAB© platform with $\sigma = 0.1$.

Table 1. Musical Composers Similarity Matrix through Probabilistic Neural Networks

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	0.08	0.14	0.08	0.36	0.08	0.06
Beethoven	0.12	—	0.26	0.22	0.02	0.20	0.34
Brahms	0.16	0.10	—	0.30	0.06	0.08	0.08
Chopin	0.00	0.18	0.36	—	0.04	0.12	0.08
Handel	0.30	0.00	0.04	0.08	—	0.18	0.26
Haydn	0.26	0.22	0.08	0.26	0.22	—	0.18
Mozart	0.16	0.42	0.12	0.06	0.30	0.34	—

One can observe that the sum of the entries of each row is not one as well as the matrix is not symmetric. For example, it can been seen that 26% of the pieces composed by Bach were more similar to the composing traces of Haydn, while only 8% of Haydn's pieces fitted best Bach's composing traces. The asymmetric property of the Table 1 is discussed in the final section.

Other notable results have also been given through another classification task. For each composer we have constructed two sets, (a) the set of training pieces and (b) the set of test pieces. The former set consists of 35 and the latter of 15 pieces out of the 50 of each composer. We have used the 35 training pieces of all seven composers to create a PNN with seven classes. The remaining 15 pieces of a composer have been presented to the PNN and the percentage of the pieces classified to each of the seven composer, has been recorded in the composer's related column. The results shown in Table 2 are the average results of 100 classification tasks as described, each task has been accomplished with different, randomly selected, training and test sets. For example, in the first column we can see the percentage of the pieces of Bach classified to each composer, including Bach.

It should be commented here that the fact that the diagonal elements are greater than any other element in the respective row or column (with an exception made to the sixth diagonal element), provides evidence for the compositional information capacity of the Dodecaphonic Trace Vector. Furthermore, it has to be noted that the uniformly selected musical forms of the pieces of each composer is reflected by the high values of some non-diagonal elements compared to their corresponding diagonal element values.

FNN Identification Success Table. Each element of the Identification Success Table (IST) (Tables 3–6) shows the mean value of the successful composer identification

Table 2. PNN Verification Table

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	64.6%	6.1%	4.5%	6.6%	14.3%	6.4%	5.5%
Beethoven	3.4%	33.3%	13.9%	11.1%	0.5%	13.6%	10.1%
Brahms	1.4%	4.7%	39.1%	15.0%	0.7%	5.0%	3.5%
Chopin	0.2%	12.4%	20.2%	53.0%	2.0%	5.6%	2.0%
Handel	13.4%	0.5%	4.7%	1.7%	66.1%	18.8%	10.6%
Haydn	6.1%	12.4%	6.5%	8.8%	6.6%	24.3%	11.6%
Mozart	10.9%	30.6%	11.1%	3.8%	9.8%	26.3%	56.7%

efforts of a FNN to distinguish whether a piece belongs to the composer of the respective row or column.

These are the mean values computed over 50 different training-testing identification tasks for each pair of composers. During each of the 50 identification tasks between two composers, of row A and column B, the network has been trained to respond 1 to the 35 training pattern of pieces composed by A and 0 to the 35 training pattern of pieces composed by B.

To test the network's performance we have used the 15 remaining pieces of A, for which we know the network should respond a value near 1 (desired output), and the 15 remaining pieces of B, for which the network should respond a value near 0 (desired output). When an unknown piece (a piece that does not belong to the training set) has been presented to the network, the network's response, x, has been considered as 1, if $x > 0.5$ and 0 in any other case.

The success rate of an identification task has been estimated as the percentage of the desired network outputs over all 30 test pieces, which is the sum of the right network responses divided by 30. For each one of the 50 identification tasks a different set of 35 training pieces for each composer has been used, which it also holds for the 15 pieces of each composer that have been used for testing. Moreover, a 5-fold cross-validation methodology has also been conducted with similar results.

The presented values on the IST correspond to the mean success value of the 50 identification tasks for each composer pair. A final comment about the training procedure would be that the sequence of the presented training patterns was randomized so that their targets would not include more than four continuously presented patterns targeted with 1.

Experimental results for the trained FNNs. In Tables 3–6, we exhibit the IST for FNNs that have been trained using the LM, Rprop and BFGS methods, respectively, as well as, with the Differential Evolution algorithm [25]. These results have been produced by the FNNs of the Neural Networks Toolbox of MATLAB [31] using the default toolbox parameters. Additionally, for the Differential Evolution algorithm we have utilized a population of twenty potential solutions and have evolved them for 500 generations with the DE/best/1/bin strategy by incorporating the default parameters for its control parameters i.e. $F = 0.5$, and $CR = 0.9$ [25].

Table 3. Identification Success Table for FNN trained with the LM method

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	81.20%	76.40%	87.00%	67.13%	69.33%	82.20%
Beethoven	83.13%	—	66.46%	63.86%	84.06%	65.00%	69.80%
Brahms	75.80%	66.40%	—	54.06%	84.13%	76.20%	78.00%
Chopin	85.46%	64.86%	55.33%	—	85.73%	77.86%	77.00%
Handel	69.80%	84.53%	84.93%	85.33%	—	70.86%	78.06%
Haydn	71.40%	62.33%	76.73%	74.80%	69.13%	—	55.80%
Mozart	82.33%	71.06%	80.00%	79.06%	80.06%	53.60%	—

Table 4. Identification Success Table for FNN trained with the Rprop method

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	82.86%	78.20%	85.00%	68.46%	72.06%	83.20%
Beethoven	83.13%	—	70.00%	65.20%	85.40%	65.60%	71.06%
Brahms	79.86%	66.60%	—	52.73%	85.46%	79.26%	80.60%
Chopin	87.06%	65.00%	54.93%	—	87.40%	76.80%	78.66%
Handel	69.13%	86.60%	86.06%	88.73%	—	71.60%	78.40%
Haydn	72.20%	65.67%	77.67%	76.33%	72.20%	—	55.06%
Mozart	83.53%	73.00%	80.06%	77.86%	80.60%	53.93%	—

Table 5. Identification Success Table for FNN trained with the BFGS method

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	83.46%	79.26%	87.33%	67.60%	72.46%	82.73%
Beethoven	84.00%	—	66.20%	66.06%	86.86%	64.26%	72.40%
Brahms	77.46%	68.33%	—	53.66%	85.13%	78.86%	77.33%
Chopin	86.40%	66.53%	55.13%	—	88.53%	74.93%	79.26%
Handel	65.33%	85.60%	85.86%	88.86%	—	70.53%	79.53%
Haydn	70.13%	63.66%	78.80%	75.33%	67.13%	—	53.86%
Mozart	81.53%	72.46%	79.06%	78.53%	78.33%	54.60%	—

Table 6. Identification Success Table for FNN trained with DE/best/1/bin

	Bach	Beethoven	Brahms	Chopin	Handel	Haydn	Mozart
Bach	—	82.20%	75.13%	83.53%	69.00%	71.80%	81.80%
Beethoven	80.80%	—	62.60%	66.20%	80.86%	66.40%	69.80%
Brahms	75.86%	68.33%	—	50.06%	82.53%	79.40%	77.00%
Chopin	83.80%	64.66%	47.46%	—	84.53%	76.86%	78.73%
Handel	69.66%	86.46%	79.60%	86.40%	—	70.53%	80.93%
Haydn	72.80%	66.66%	77.93%	75.73%	72.00%	—	51.40%
Mozart	82.60%	71.80%	78.00%	75.26%	79.53%	52.26%	—

5 Discussion and Concluding Remarks

Through this work, evidence has been provided that the Dodecaphonic Trace Vector and, consequently, the Chroma Profile vector contain considerable capacity of information for the individuality of a composer. Moreover, experts with sufficient musical background need to amplify the findings of the work at hand and aid the musical analysis by educing related information.

The PNN similarity matrix in Table 1, as well as the IST table of the FNNs imply that as long as the similarity between two composers increases, the identification effectiveness between those two decreases. The latter comment is a sensible assumption, though we can see some exceptions, for example in Table 1 we can see that the similarity between Mozart and Handel is greater than the similarity between Mozart and Haydn, though in Table 3 the identification task is more precise for the first couple.

Future work would incorporate further analysis of the lack of symmetry of Table 1 and inquiry on the perspective for information extraction out of it. These pieces of information, combined with historical facts, could lead to an *influence* diagram between the composers that provides or validates the evidence on the existence of composers of *fundamental influence*.

More accurate results could be reached by refining further the musical structure through the symbolic analysis of musical scores using more sophisticated representation of musical items, or by incorporating pitch transitions and pitch durations. It is also intended to compare the DTV efficiency against other approaches. Moreover, instead of the PNN used here, any other supervised clustering tool, such as Support Vector Machines [32], could be used for the construction of the similarity matrix. The same holds for the identification success table. Finally, it is evident that a widely acceptable musical database should be created, in order to analyze and compare musical data extraction approaches.

References

1. Lebar, J., Chang, G., Yu, D.: Classifying musical score by composer: A machine learning approach, `http://www.stanford.edu/class/cs229/projects2008.html`
2. Tzanetakis, G., Cook, P.: Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing 10(5), 293–302 (2002)
3. Mostafa, M.M., Billor, N.: Recognition of western style musical genres using machine learning techniques. Expert Systems with Applications 36(8), 11378–11389 (2009)
4. Peeters, G.: Musical key estimation of audio signal based on hidden markov modeling of chroma vectors. In: Proc. of the Int. Conf. on Digital Audio Effects DAFx 2006, pp. 127–131 (2006)
5. Gomez, E.: Tonal description of polyphonic audio for music content processing. INFORMS Journal on Computing 18(3), 294–304 (2006)
6. Fujishima, T.: Realtime chord recognition of musical sound: A system using common lisp music. In: Proc. ICMC, pp. 464–467 (1999)
7. Meador, S., Uhlig, K.: Content-based features in the composer identification problem, `http://www.stanford.edu/class/cs229/projects2008.html`
8. Pardo, B., Birmingham, W.P.: Algorithms for chordal analysis. Computer Music Journal 26(2), 27–49 (2002)

9. Beran, J.: Statistics in Musicology, 1st edn., July 2003. Chapman & Hall/CRC (2003)
10. Pollastri, G.S.E.: Classification of melodies by composer with hidden markov models (November 2001), http://www.computer.org/portal/web/csdl/doi/10.1109/WDM.2001.990162
11. Liu, Y.W.: Modeling music as markov chains: Composer identification, https://www-ccrma.stanford.edu/~jacobliu/254report.pdf
12. MIDI: The classical MIDI connection site map, http://www.classicalmidiconnection.com
13. Koepf, S., Haerpfer, B.: The MSQ project, http://www.aconnect.de/friends/msq2/msq.htm
14. Schell, D.: Optimality in musical melodies and harmonic progressions: The travelling musician. European Journal of Operational Research 140(2), 354–372 (2002)
15. Homei, M., Kazushi, N.: Extraction and analysis of Chroma-Profile from MIDI data. Joho Shori Gakkai Kenkyu Hokoku 2003(82(MUS-51)), 97–101 (2003)
16. Specht, D.F.: Probabilistic neural networks. Neural Networks 3(1), 109–118 (1990)
17. Hand, D.: Kernel Discriminant Analysis. John Wiley & Sons Ltd., Chichester (1982)
18. Georgiou, V.L., Pavlidis, N.G., Parsopoulos, K.E., Alevizos, P.D., Vrahatis, M.N.: New self-adaptive probabilistic neural networks in bioinformatic and medical tasks. International Journal on Artificial Intelligence Tools 15(3), 371–396 (2006)
19. Georgiou, V.L., Alevizos, P.D., Vrahatis, M.N.: Novel approaches to probabilistic neural networks through bagging and evolutionary estimating of prior probabilities. Neural Processing Letters 27(2), 153–162 (2008)
20. Haykin, S.S.: Neural networks and learning machines. Prentice Hall, Englewood Cliffs (2008)
21. Plagianakos, V.P., Vrahatis, M.N.: Parallel evolutionary training algorithms for 'hardware–friendly' neural networks. Natural Computing 1, 307–322 (2002)
22. Magoulas, G.D., Plagianakos, V.P., Vrahatis, M.N.: Neural network-based colonoscopic diagnosis using on-line learning and differential evolution. Applied Soft Computing 4, 369–379 (2004)
23. Plagianakos, V.P., Magoulas, G.D., Vrahatis, M.N.: Evolutionary training of hardware realizable multilayer perceptrons. Neural Computing and Application 15, 33–40 (2005)
24. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: Parallel distributed processing: explorations in the microstructure of cognition, foundations, vol. 1, pp. 318–362. MIT Press, Cambridge (1986)
25. Price, K., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Natural Computing Series. Springer, New York (2005)
26. Marquardt, D.W.: An algorithm for Least-Squares estimation of nonlinear parameters. SIAM Journal on Applied Mathematics 11(2), 431–441 (1963)
27. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 5(6), 989–993 (1994)
28. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The Rprop algorithm. In: Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, CA, pp. 586–591 (1993)
29. Nocedal, J., Wright, S.: Numerical Optimization. Springer, Heidelberg (1999)
30. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization. Academic Press, London (1982)
31. Demuth, H., Beale, M.: Neural Network Toolbox 6: For use with MATLAB: User's Guide. In: The Mathworks, Cochituate Place, 24 Prime Park Way, Natick, MA, USA (1992-2009)
32. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2(2), 121–167 (1998)