TECHNICAL REPORT NO. TR-DIE-GPDS-08/23/2005

Title:	Analysis of Enhanced Decision-Based Neural Networks — Applications in UCI databases, channel equalization, texture classification and pharmacokinetics.
Authors:	Gustavo Camps-Valls — `gustavo.camps@uv.es` Luis Gómez-Chova Joan Vila-Francés José D. Martín-Guerrero Antonio J. Serrano López Emilio Soria-Olivas
Institution 1:	Escola Tècnica Superior d'Enginyeria (ETSE). Dept. Enginyeria Electrònica (DIE). Universitat de València. València, Spain.
Address 1:	Dr. Moliner 50. 46100 Burjassot, València (Spain).
Date:	08/23/2005
TR ID:	TR-DIE-GPDS-08/23/2005
Availability:	`http://www.uv.es/~gcamps/neural.htm` `http://www.uv.es/~gcamps`

ETSE - Dept. Enginyeria Electrònica
Universitat de València – Estudi General
València (Spain)

Analysis of Enhanced Decision-Based Neural Networks — Applications in UCI databases, channel equalization, texture classification and pharmacokinetics.

Gustavo Camps-Valls, Luis Gómez-Chova, Joan Vila-Francés,
José D. Martín-Guerrero, Antonio J. Serrano López and Emilio Soria-Olivas *

Abstract

This paper proposes intuitive modifications to the Decision-Based Neural Network (DBNN) learning algorithm. The DBNN adopts a hierarchical network structure and an effective credit-assignment scheme. The structure is formed by a set of subnets, constituted by (nonlinear) basis functions (BFs), which are dedicated to model each class. The competitive learning algorithm runs among dedicated BFs between classes following a reinforcement/antireinforcement weight update strategy. A limitation of the standard algorithm is that it does not run between BFs in the same class. The goal of our proposal is to stimulate competition at this scale as well. We introduce intuitive modifications to the updating rules of the standard DBNN, and benchmark its modified version in both synthetic and real problems: ten problems from the UCI Machine Learning benchmark database, channel equalization, texture classification and detection of Digoxin intoxication. Results show that the introduced modifications improve performance at the expense of reduced computational increase.

Keywords: Decision-based neural network; hierarchical network structure; competitive credit-assignment scheme; local competition, robust, stability, system identification, channel equalization, texture classification.

---

[*] All authors are with Grup de Processament Digital de Senyals, Universitat de València, Spain. E-mail: gcamps@uv.es, http://www.uv.es/∼gcamps. Correspondence: Gustavo Camps-Valls. C/ Dr. Moliner, 50. 46100 Burjassot (València, Spain)

Contents

1 Introduction 4

2 Decision-Based Neural Networks 4

3 Enhanced SCH-DBNN 6

4 Results 7

4.1 The UCI database 8
4.2 Channel equalization 10
4.3 Texture classification 11
4.4 Digoxin intoxication detection 12

5 Conclusions 15

References 16

1 Introduction

The fundamental principle of supervised learning is the presence of a teacher that tells the correct output label for the classification of a given input example. This learning principle has been extensively treated in the machine learning community, giving rise to many important contributions, such as linear perceptrons (Rumelhart, McClelland, & PDP Research Group, 1986), multilayer feedforward neural networks (Haykin, 1999), decision trees (Breiman, Friedman, Olshen, & Stone, 1984), Support Vector Machines (SVM) (Vapnik, 1998), and hierarchical structures (Kung, 1995). See (Friedman, Hastie, & Tibshirani, 2001) for a comprehensive introduction to statistical learning. They all are trained using a specific decision-based learning algorithm usually based upon credit-assignment criteria. For instance, weights in a neural network are usually updated based on the back-propagation of the committed error using an iterative procedure.

The original learning algorithm of the DBNN is decision-boundary driven, which usually provides very fast and satisfactory learning performance. However, different strategies are needed when dealing with highly overlapping distributions and/or issues on false acceptance/rejection. Some modifications to deal with this problem in standard DBNN have been proposed; non-linear discriminant functions (Kung & Mao, 1991), HiPer nets (Kung & Taur, 1992), Probabilistic DBNN via expectation-maximization (Lin & Kung, 1995), fuzzy-decision neural networks (Taur & Kung, 1993), and modular networks (Kung, 1995). In (Kung & Taur, 1995), the capabilities of decision-based neural networks (DBNN) were successfully explored in the context of signal and image classification. The DBNN adopts a hierarchical network structure with basis functions (BFs), and an effective credit-assignment scheme. On the one hand, the hierarchical structure offers a great flexibility and scalability to accommodate complex decision boundaries. On the other hand, a well-designed reinforcement learning algorithm for DBNNs ensures competition in a more natural way. The joint consideration allows us to stablish a good trade-off between complexity of the network and results.

However, DBNNs learning algorithms are based on the competition among dedicated BFs between classes, but not between BFs in the same class. This paper proposes some modifications to the original DBNN learning algorithm in order to tackle this problem. The goal of our proposal is to stimulate competition among BFs in the same class network. For this purpose, we introduce intuitive modifications to the updating rules of the standard DBNN. Results in many problems of pattern recognition, signal and image processing confirm the usefulness of our method.

The remainder of the paper is organized as follows. Section 2 reviews the DBNN standard algorithm. Section 3 shows the proposed formulation. Section 4 shows results of the models in both synthetic and real pattern recognition problems. Section 5 ends the paper with some conclusions.

2 Decision-Based Neural Networks

When dealing with multi-classification problems, it may result of great convenience to make a task division, and thus to implement dedicated models to each category or class to be classified. Neural networks built as One-Class-One-Network (OCON) structures are aimed to achieve this objective since they are made up of a set of local subnets, each of them dedicated to a given output class (Kung, 1995).

Learning algorithms used with this kind of structures are based on mitigating the influence of the errors produced in the output units when they are back-propagated. This is accomplished by updating only the weights of the units that make a mistake in the classification. A reinforced learning is applied to the subnet corresponding to the winner class, and an anti-reinforced learning to the loser one. Consequently, we are encouraging competition between subnets by giving a "prize" (in the way of a negative gradient on the error surface) to the subnet corresponding to the winner class, and a "penalization" (in the way of a positive gradient) to the one corresponding to the loser class, as follows:

$$\Delta \mathbf{w} = \pm \eta \nabla \phi(\mathbf{x}, \mathbf{w}) = \pm \eta \left[\frac{\partial \phi}{\partial \mathbf{w}_1}, ..., \frac{\partial \phi}{\partial \mathbf{w}_P} \right]^T, \tag{1}$$

where P is the total number of parameters, η is a positive learning rate, and ϕ represents the discriminant decision function, and is formed by a set of Basis Functions (BFs), which can be linear, radial, polynomial functions, etc. This constitutes the standard form of the DBNN (Decision-Based Neural Network) learning algorithm.

The formal updating rule of the DBNN is as follows. Given a labeled training data set $\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(M)}\}$, each one corresponding to an output class $\{\mathcal{Y}_i, i = 1, ..., L\}$, each class is modeled by a subnet with discriminant functions $\phi(\mathbf{x}, \mathbf{w}_i)$, $i = 1, ..., L$ (Figure 1). If the mth training pattern $\mathbf{x}^{(m)}$ belongs to class $\mathcal{Y}_i$, and

$$\phi(\mathbf{x}^{(m)}, \mathbf{w}_j^{(m)}) > \phi(\mathbf{x}^{(m)}, \mathbf{w}_l^{(m)}), \qquad \forall l \neq j, \tag{2}$$

the winning class for the pattern m is the jth class (or subnet). Therefore, two possibilities arise, which are summarized in the following scheme:

1. When $j = i$, then the pattern $\mathbf{x}^{(m)}$ is already correctly classified and thus, no update is necessary.

2. When $j \neq i$, then the pattern $\mathbf{x}^{(m)}$ is still misclassified and thus, the following update is performed:

   (a) *Reinforced learning:*

$$\mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta \nabla \phi(\mathbf{x}, \mathbf{w}_i) \tag{3}$$

   (b) *Antireinforced learning:*

$$\mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta \nabla \phi(\mathbf{x}, \mathbf{w}_j) \tag{4}$$

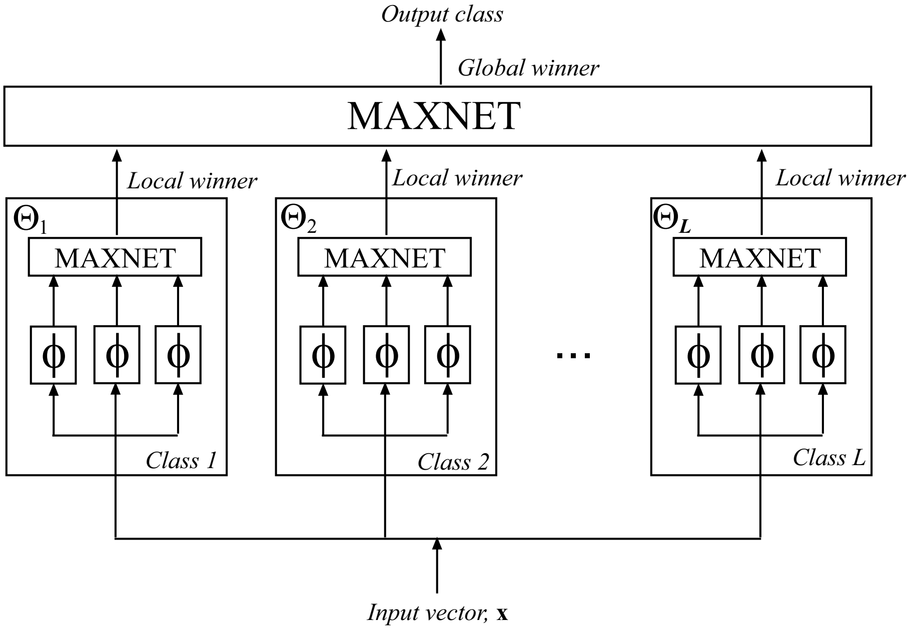

Figure 1: *The subcluster Hierarchical DBNN is based on a set of units or subclusters for each subnet. Each subnet is specially directed to model a class or category, and it is formed by subnodes or subclusters.*

Note that, for all $k \neq i$ and $k \neq j$, $\mathbf{w}_k^{(m+1)} = \mathbf{w}_k^{(m)}$, i.e. only the weights of the winner subnet and the correct class subnet are updated. The rest remain unchanged. This characteristic increases the speed of the learning process.

In order to further localize the training credit-assignament, the so-called *Subcluster Hierarchical DBNN* (SCH-DBNN) was proposed in (Kung & Taur, 1995). The basic structure is shown in Fig. 1. For the subcluster hierarchical structure, the concepts of local winner and global winner were introduced. The local winner s_l is the winner among the subnodes within the same lth subnet, $s_l = Arg\{\max_{k_l} \Theta_l(\mathbf{x}, \mathbf{w}_{k_l})\}$. The global winner is the winner among all the subnets. The jth subnet will be labeled as the global winner if its local winner wins over all the other local winners, $\Theta_j(\mathbf{x}, \mathbf{w}_{s_j}) > \Theta_l(\mathbf{x}, \mathbf{w}_{s_l})\ \forall j \neq l$. This modification obviously involves substituting the discriminant function of the subnets by the local winners in equations (3) and (4), as follows: $\phi(\mathbf{x}, \mathbf{w}_i) \leftrightarrow \Theta_i(\mathbf{x}, \mathbf{w}_{s_i})$, and $\phi(\mathbf{x}, \mathbf{w}_j) \leftrightarrow \Theta_j(\mathbf{x}, \mathbf{w}_{s_j})$. Therefore, when $\mathbf{x}$ is misclassified, antireinforced learning is applied to the local winner in the global winner subnet, and the reinforced learning is applied to the local winner in the correct class subnet.

3 Enhanced SCH-DBNN

Despite that the previous approaches deal very efficiently with global and local credit-assignment issues, they are prone to some limitations because confidence/penalization relies only on inter-class competition, i.e. no in-class BFs responses are taken into account but the one from the local winner subnode. This limitation could lead to biased trainings if parameter η is not tuned correctly and, for many iterations, a node in a subnet becomes the local winner. This could limit the potential of the neural network and, in turn, increase the training time and efforts.

This problem can be alleviated by promoting local competition in the subclusters themselves. We introduce some modifications to the SCH-DBNN learning algorithm that take into account the local information provided by the BFs in a same class (subnet). The new algorithm is called Enhanced Subcluster Hierarchical DBNN (ESCH-DBNN). For short, we introduce a local competition parameter that trades-off the inter-class against the in-class competition of BFs and thus, assigns different penalization of BFs in the same subnet.

Following the previous formulation: (a) we will update the weights in the network only when an error is committed, and (b) the winner subnode s_l of the subnet l will be governed by the usual reinforced/antireinforced updates using the η learning rate. Additionally, the updating rules will also penalize errors of all loser subnodes by using a different learning rate, γ.

Let us now assume that the pattern $\mathbf{x}^{(m)}$ should belong to class i, but the jth subnet is selected as the global winner. In this general situation, the governing algorithm is as follows:

1. When $j = i$, then the pattern $\mathbf{x}^{(m)}$ is already correctly classified and thus, no update is necessary.
2. When $j \neq i$, then the pattern $\mathbf{x}^{(m)}$ is still misclassified and thus, the following update is performed:
   (a) *Enhanced reinforced learning:*

$$\mathbf{w}_{s_i}^{(m+1)} = \mathbf{w}_{s_i}^{(m)} + \eta \nabla \Theta_i(\mathbf{x}, \mathbf{w}_{s_i}) \tag{5}$$

$$\mathbf{w}_{i \neq s_i}^{(m+1)} = \mathbf{w}_{i \neq s_i}^{(m)} + \gamma \nabla \Theta_i(\mathbf{x}, \mathbf{w}_{i \neq s_i}) \tag{6}$$

   (b) *Enhanced antireinforced learning:*

$$\mathbf{w}_{s_j}^{(m+1)} = \mathbf{w}_{s_j}^{(m)} - \eta \nabla \Theta_j(\mathbf{x}, \mathbf{w}_{s_j}) \tag{7}$$

$$\mathbf{w}_{j \neq s_j}^{(m+1)} = \mathbf{w}_{j \neq s_j}^{(m)} - \gamma \nabla \Theta_j(\mathbf{x}, \mathbf{w}_{j \neq s_j}) \tag{8}$$

Note that intuitively, the algorithm motivates nodes to become local winners both in winner (Eq. (5)) and loser subclusters (Eq. (6)). Additionally, the antireinforced learning is not only provided to the local winner in the correct class subnet (Eq. (7)), but also to their competing local nodes (Eq. (8)). In practice, good results are obtained using $\gamma < \eta$, which remarks the idea of controlling the "extra" credit assigned to loser nodes throughout the network. Finally, note that our algorithm reduces to the classical SCH-DBNN for $\gamma = 0$.

4 Results

In this section, we test our proposal in many synthetic and real pattern recognition problems: ten problems from the UCI database, channel equalization, and texture classification of Brodatz images. In all cases, we used linear basis functions (LBF), which has demonstrated a good trade-off between generalization capabilities and simplicity compared to radial or elliptic basis functions, and to the classical multilayer perceptron (Kung & Taur, 1995). We varied the number of nodes in a subcluster (< 20 to avoid overfitting), the weight initialization range and the learning rate (η between 0.01 and 3) in order to determine the best topology. In the case of the Enhanced SCH-DBNN we additionally varied the γ parameter between 0.001 and 0.1. The selection of the best subset of free parameters is usually done by cross-validation methods but this can lead to poor generalization capabilities and lack of representation. We alleviated this problem by using the 8-fold cross-validation method[1] with the training data set.

Data were first normalized to give zero mean and unit variance. Many discriminative methods are often more accurate and efficient when dealing with only two classes. For large numbers of classes, higher-level multiclass methods utilize these two-class classification methods as the basic building blocks, namely "one-against-the-rest" procedures. However, such approaches lead to suboptimal solutions when dealing with multiclass problems and the well-known problem of the "false positives". This is not a difficulty for the DBNN, given that each subnet is dedicated to model a given class and thus, they inherently work in a multiclassification scheme.

All models were developed in MATLAB® environment (Mathworks, Inc). Since the computational burden was very high, *m-files* were translated to MEX-files and the programs were run on fast workstations (Pentium 4, 2GHz processor, 512Mbytes RAM).

4.1 The UCI database

The UCI database contains data sets and domain theories that can be used to evaluate learning algorithms (Blake & Merz, 1998). We selected ten databases from the UCI Machine Learning repository to test the capabilities of our proposal. The selected databases have different number of features, input dimension and classes in order to analyze all possible situations of a given algorithm.

Table 1 shows some basic characteristics for each database, the reported results in (Blake & Merz, 1998), the obtained results using the Standard and Enhanced SCH-DBNNs, along with the average CPU time used for training. From this table, we can extract some preliminary conclusions: (1) Our proposed modification improves the results obtained using the standard algorithm in all databases; (2) the differences are greater in problems with higher number of classes, which indicates that inter-class

---

[1] The 8-fold cross–validation uses 7/8 of the data for training and 1/8 for validation purposes. This procedure is repeated eight times with different, randomly selected validation sets.

Table 1: *Characteristics and validation results of the selected UCI databases. From left to right columns: database name, number of patterns (♯P), number of features (♯F), number of classes (♯C), best reported accuracy in (Blake and Merz, 1998), best accuracy obtained using the standard DBNN, the obtained accuracy using the Enhanced DBNN, and the average CPU time for the standard DBNN and our proposal.*

	♯P	**♯F.**	**♯C.**	**ACCURACY [%]**			**CPU TIME [s.]**	
DATA BASE				**REPORTED IN (Blake & Merz, 1998)**	**SCH DBNN**	**ESCH DBNN**	**SCH DBNN**	**ESCH DBNN**
wdbc	569	30	2	-	97	98	34.10	40.40
glass	214	10	7	-	69	73	14.98	12.55
ionosphere	351	34	2	97	96	98	23.88	33.88
iris	150	4	3	-	95	97	1.80	2.25
liver-disorders	345	6	2	-	59	61	4.14	5.13
pima-diabetes	768	8	2	76	75	77	12.28	16.77
sonar	208	60	2	83	90	92	24.96	28.25
vehicle	846	18	4	-	68	74	60.92	71.15
vowel	990	10	11	56	71	81	108.91	150.54
wine	178	13	3	100	95	100	6.94	7.96

competition has been successfully enhanced; and (3) the differences are not numerically significant as the number of features increases, which makes our method especially convenient for large scale problems. In addition, the results obtained by our proposal are at the expense of a moderate increase in the computational burden (average increase of 21% for all databases). This result was expected since more comparisons and thus more weight updates must be performed. It is worthnoting that the CPU time increases almost linearly with the number of classes, which is due to the fact that each class is modeled by a set of dedicated subnets.

Table 2 shows a comparison of the most updated results reported in the literature for some databases (Duch, 2002). We can see that in most cases, our method outperforms some of the state-of-the-art results provided by neural networks and support vector machines (ionosphere and sonar datasets), or decision trees (diabetes, ionosphere, and glass datasets).

4.2 Channel equalization

When a binary signal is transmitted through a real dispersive channel, the received signal is affected by inter-symbol interference. Moreover, if noise is present, further corruption ensues. Therefore, in many

Table 2: *Benchmark with the most updated results (see (Duch, 2002)) provided by different methods in a subset of the databases selected.*

pima-diabetes		**ionosphere**		**sonar**		**glass**	
Method	**Acc.**	**Method**	**Acc.**	**Method**	**Acc.**	**Method**	**Acc.**
SVM	77.6	3-NN + simplex	98.7	1-NN	97.1	Adaptive metric NN	75.2
ESCH-DBNN	**77.1**	**ESCH-DBNN**	**98.3**	TAP MFT Bayesian	92.3	**ESCH-DBNN**	**73.3**
Semi-Naive Bayes	76.0	MLP+BP	96.0	**ESCH-DBNN**	**92.1**	Discriminant Adaptive NN	72.9
C4.5	76.0	C4.5	94.9	SVM	90.4	kNN	72.0
Naive Bayes	74.5	SVM	93.2	MLP+BP	90.4	C4.5	68.2

practical cases, equalization is necessary to recover the information from the received signal. Under adverse conditions such as low signal to noise ratio or when the distribution of the received samples is not linearly separable, non-linear techniques are more fitted, and neural networks are becoming a common choice (Gibson, Siu, & Cowan, 1991).

The system which we shall consider is depicted in Fig. 2. The input to the channel is assumed to be a sequence, $\{y_i\}$, of independent symbols extracted from a specific alphabet. The channel output is corrupted by random noise, $\{n_i\}$, which is considered to be an additive white Gaussian process. The task of the equalizer is to recover the input sequence, $\{y_i\}$, from the received sequence $\{r_i\}$.

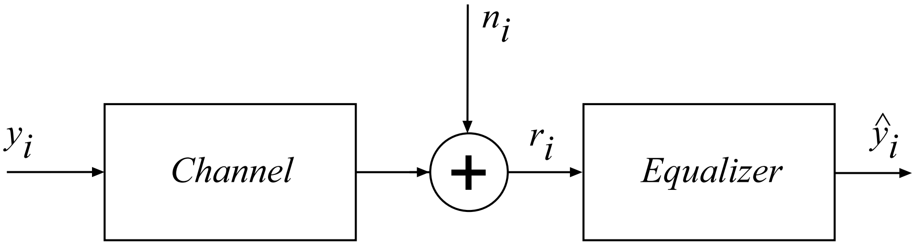

Figure 2: *Schematic of a data transmission system.*

The experiment consisted of equalizing a binary pulse amplitude modulation signal at the output of a dispersive channel, whose model was a tapped-delay line with an impulse response

$$h_n = \delta_n + 0.6\delta_{n-1} + 0.2\delta_{n-2} - 0.1\delta_{n-3} + 0.1\delta_{n-4}. \tag{9}$$

This impulse response can represent an example of minimum-phase dispersive channel, which is common in suburban and hilly terrain environments (Jeruchim, Balaban, & Shanmugan, 1992). At the receiver, a block of m received signals, $[y_n, y_{n-1}, \ldots, y_{n-m+1}]$ where n is the current discrete time index, is used to estimate the transmitted signal. A possible way to solve the problem is to consider that the set of

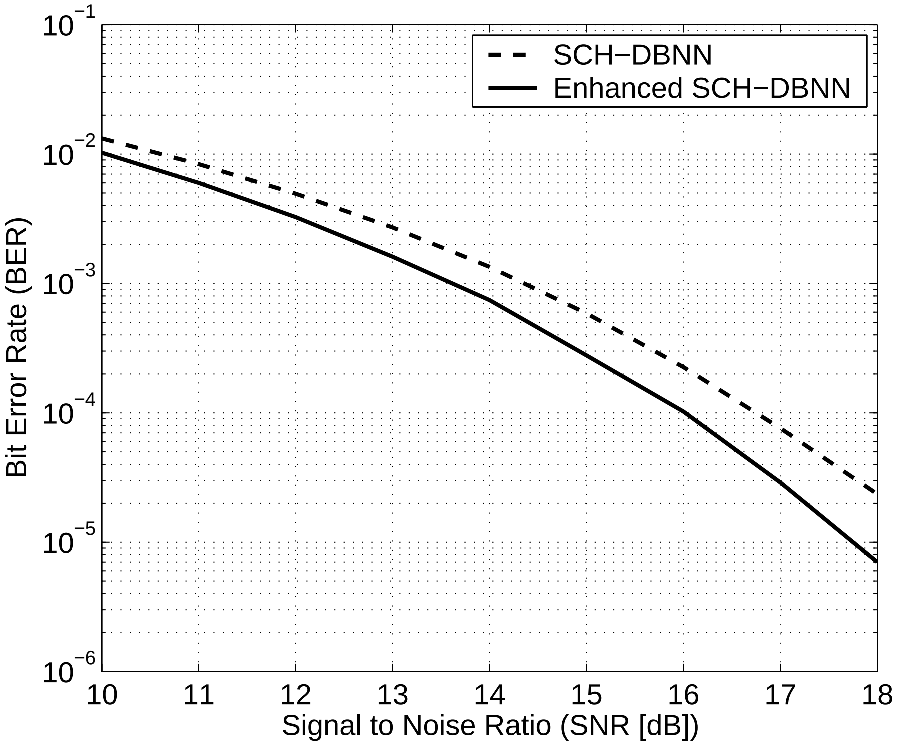

Figure 3: *BER vs. Averaged SNR for SCH-DBNN (dotted) and Enhanced SCH-DBNN (solid) equalizers.*

received samples has to be classified according to the two possible classes (states) of the input: in this case ±1. A set of 128 randomly generated samples was transmitted; 64 samples were used as training data set and the remaining samples for validation. In order to measure the bit error rate (BER), an independent test burst of 10^6 samples was also used; the experiment was done 10 times for each SNR from 10 to 18 dB (1 dB steps), which represents a reasonable confidence margin for the least measured BER.

BER is depicted in Figure 3. The SCH-DBNN performance was poorer than our proposal, mainly due to the low number of samples used to train the model; these samples generate a channel output which does not have enough information to estimate the statistical distribution of the process when a limited number of weight updates are used in the training methodology. In this sense, the Enhanced SCH-DBNN algorithm appears to be a better choice for dealing with small training data sets; experiments show a difference of at least 1.1 dB.

4.3 Texture classification

During the past few decades, the analysis of texture in digital images has received considerable attention and numerous approaches have been proposed (Tuceryan & Jain, 1993). Texture classification is defined as the problem of classifying pixels in an image according to their textural cues. This problem

Table 3: *Success classification rates [%] for the SCH-DBNN and the Enhanced SCH-DBNN for different size of the window in the validation set.*

Window Size	**SCH-DBNN**	**Enhanced SCH-DBNN**
5×5	78.2%	80.3%
7×7	80.5%	84.1%
15×15	75.3%	75.7%

is different from conventional image segmentation as the texture is characterized using both the gray value for a given pixel and the gray-level pattern in the neighborhood surronding the pixel, and thus it takes advantages of the spatial information (Greenspan, Godman, Chellappa, & Anderson, 1994; Muhamad & Deravi, 1994; Jain & Karu, 1996; Patel, 1996; Chen, Nixon, & Thomas, 1997).

Nevertheless, the texture classification problem is conventionally divided into the two subproblems of feature extraction and classification (Muhamad & Deravi, 1994; Chen et al., 1997). Recently, several works have embedded the filtering scheme in a neural network architecture to produce texture segmentation (Jain & Karu, 1996; Patel, 1996). However, this poses the question of where the gain in recognition rates comes from. This section is more concerned on model capabilities and thus no filtering or transformation is performed to data.

In many applications of texture classification, the use of committees of experts is a common practice. A fundamental tenet of the theory of mixture of experts is that an effective ensemble should consist of networks that are not only highly correct, but ones that make their errors on different parts of the input space as well. Therefore uncorrelated predictions are desired and thus using different subnets to model each class could be more appropiate to this purpose, especially if they are embedded in an effective competition scheme. In this context, our motivation is to show that dedicated models (subnets) can do a good job when their competition is enhanced.

To show the effectiveness of the proposed method, an artificially generated texture image from the Brodatz texture set was segmented. All textures were originally gray-scale images with 256 levels. This image was composed of 4 different textures (Fig. 4[top]). The size of the image was 300×300 and the size of each textured sub-image was 75×75. The experiments were performed with input window sizes 5×5, 7×7, and 15×15, which were built using the standard autoregressive (AR) features configuration (Randen & J.H. Husoy, 1999). The training data set was obtained by randomly selecting 125 of the input patterns from each texture. The desired classes for the patterns were then assigned.

Table 3 summarizes the results obtained using the standard and the Enhanced SCH-DBNN. The low number of samples selected for training (500) induces a difficult problem but (1) encloses the application of a method to real-time conditions and (2) serves as robustness test. Figure 4[(a)–(d)] shows the result

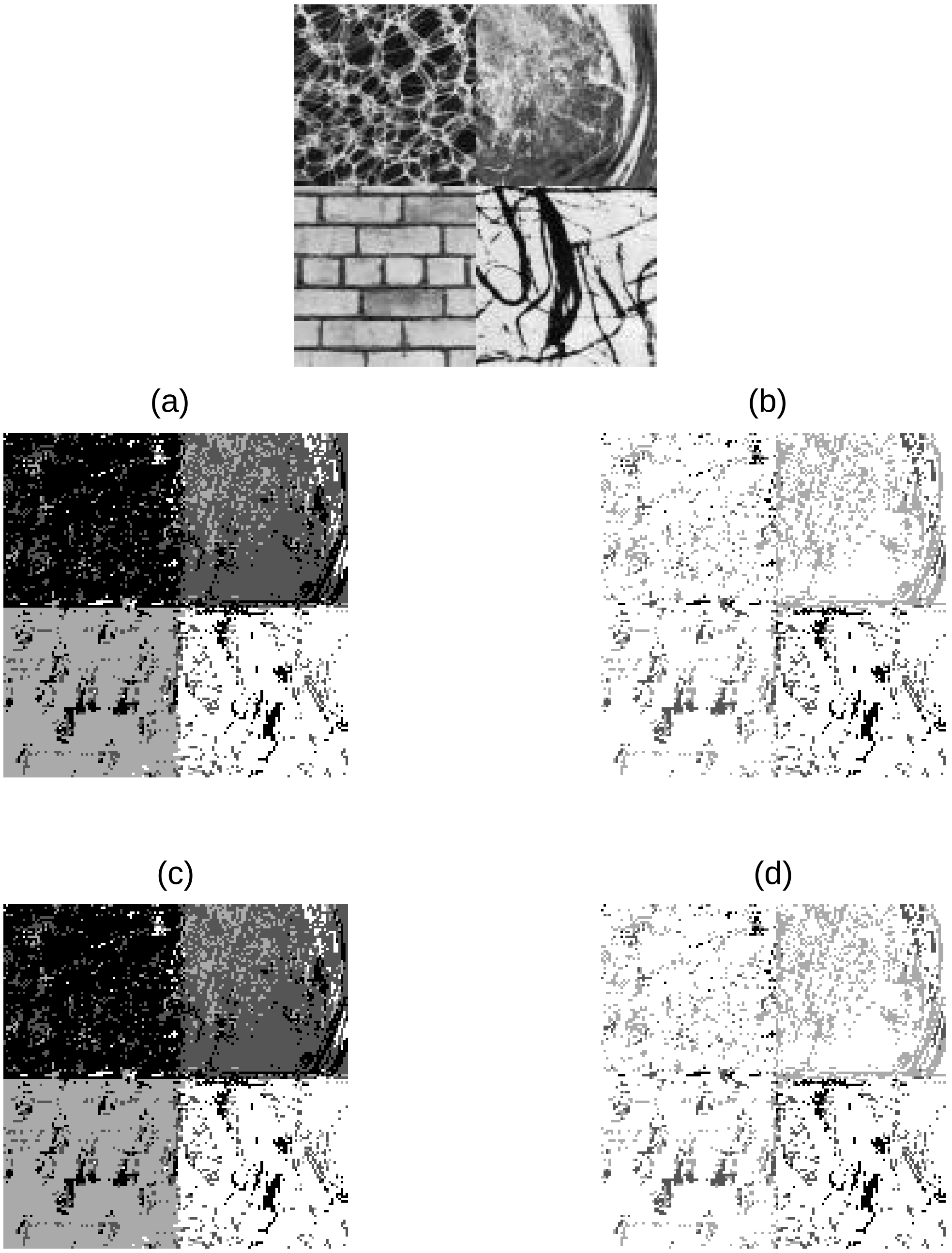

Figure 4: **Top:** *Original four Brodatz textures selected.* **Bottom:** *Texture classification and misclassified pixels illustrated as black gray levels for the SCH-DBNN ((a) and (b)) and the Enhanced SCH-DBNN ((c) and (d)). All results are obtained using a 7×7 window size.*

of texture classification using the SCH-DBNN and the Enhanced version. The best results are obtained using a 7×7 input window with our proposal. The low classification rate for the classifier that used 15×15 input window can be attributed to the instability of classification for the patterns lying on the texture boundaries and frequency changes (see committed errors in texture four, Fig. 4[d]). Note that these results are poor compared to state-of-the-art approaches, which is due to the fact that the algorithm runs without any additional filtering as in (Kim, Jung, Park, & Kim, 1999, 2002), or wavelet transformation (Li, Kwok, Zhu, & Wang, 2003) for the sake of a fair comparison.

4.4 Digoxin intoxication detection

Digoxin (DGX) is a drug widely used for treating congestive cardiac failure and symptomatic alterations of the heart rate such as auricular fibrillation and paroxysmal supraventricular tachycardia. It improves the effective behavior of the heart and its immediate consequence is to relax the heartbeat. The main drawback derived from using this drug refers to the possibility of producing intoxication in the patient due to its narrow therapeutical range of application, whose value is usually accepted between 0.8 and 2 ng/mL (Evans, 1993).

Digoxin intoxication is one of the most usual pathologies at Hospital Emergency Services with dangerous consequences for the patients, whom sometimes need to be admitted to an Intensive Care Unit. That is the main reason why developing models for detecting patients with risk of being intoxicated becomes essential. Patients can be thus classified into two categories: patients with high risk of being intoxicated (DGX plasma levels $\geq$2ng/mL) and patients with low risk of being intoxicated (DGX plasma levels $<$2ng/mL). The main difficulty to predict patients' state appears when plasmatic digoxin concentration not only depends on the externally administered dose but also on the renal activity, the global treatment, and the patient's characteristics. Finding a mathematical model capable to achieve an optimal classification of the patients with risk of intoxication is difficult using linear models, as assessed in a preliminary work. These limitations carried us to use more sophisticated tools such as neural networks.

Table 4 shows the confusion matrix provided by the best classifiers. Several success estimations are indicated: success ratio (SR), sensitivity (SE) and specificity (SP). Sensitivity is defined as the correct classification percentage on intoxicated patients and specificity is the correct classification percentage on non-intoxicated patients. The issue of false positives is a big problem in clinical practice in general, and for detecting potentially intoxicated patients treated with digoxin in particular. Thus, it is essential to achieve high rates of joint SE and SP. We can observe that results are better in training and validation when using our proposal. Despite it is a slight improvement, the model shows more robust in terms of sensitivity, yielding an improvement of 12% for training and 11% for validation.

Finally, we decided to analyze the rate of updates that our algorithm perform through the training process. We measured the squared variation of the weights in the network *per* subcluster. This anal-

Table 4: *Results for the best SCH-DBNN and Enhanced SCH-DBNN (in brackets) for training and validation. Both networks had a structure of two subclusters dedicated to the low risk class and four subclusters to the high risk class.*

Training	**PC > 2ng/ml**	**PC < 2ng/ml**
Test +	7(9)	8(7)
Test -	10(8)	58(59)
SR = 78.31% (81.93%)	**SE = 41.18% (52.94%)**	**SP = 87.88% (89.39%)**
Validation	**PC > 2ng/ml**	**PC < 2ng/ml**
Test +	6(7)	3(3)
Test -	3(2)	16(16)
SR = 78.57% (82.14%)	**SE = 66.67% (77.78%)**	**SP = 84.21% (84.21%)**

Table 5: *Average squared variation* per *subcluster (l) in a subnet,* $\Delta\mathbf{W}_l$.

Subnet 1 (Class 0)		**Subnet 2 (Class 1)**			
l=1	l=2	l=3	l=4	l=5	l=6
66.88	68.70	67.63	59.23	56.06	56.93

ysis can yield useful information about which subcluster was more efficient and which one had lower discriminative power for a dedicated class. In addition, this way we could track the updating rules and the need of a higher or smaller γ parameter. Further work will consider the adaptive variation of γ as a function of the local committed errors and the local winner penalization. The average weight variation function is computed as follows:

$$\Delta\mathbf{W}_l = \sum_i \left[\frac{w_{i,l}(n) - w_{i,l}(n-1)}{max(w_{i,l}(n)) - min(w_{i,l}(n))} \right] \tag{10}$$

where $w_{i,l}(n)$ represent the weights w_i in subcluster l at training epoch n.

We can observe that learning of class '1' requires lower average efforts, in terms of rate of weight updates, than that of class '0'. This is due to the fact that class '1' corresponds to patients potentially intoxicated who constitute the 23% of the whole cohort. Hence, the relative success rate tends to be higher and thus the nodes in the corresponding subclusters require a lower number of updates. In general terms, the best results have been obtained using simple structures but always employing more nodes for class '1'. In any case, from our experience, it does not appear a clear relationship between the number of training patterns in a class and the necessary subclusters.

5 Conclusions

In this paper, a straigthforward modification of the subcluster hierachical DBNN has been proposed. The idea relies on the concepts of local and global competitions, by including additional reinforcement/antireinforcement rules at a local scale. We tested the algorithm in terms of robustness and accuracy in many pattern recognition problems. In particular, experiments with ten UCI databases demonstrated that the algorithm is not drastically affected by the output dimension and the number of patterns. Also, a substantial gain in terms of BER was obtained in channel equalization of a minimum-phase channel. Our algorithm showed good recognition rates when working with a reduced dataset in texture classification experiments. Finally, good results were also obtained in the complex problem of identification of possibly intoxicated patients treated with digoxin.

In general, improved results have been obtained at the cost of a relatively low increase of the computational burden. These outcomes show that our method could be useful in other credit-assignment and multi-expert learning schemes. At present, we are working on the extension and applicability of our method to other basis functions, such as radial-based kernel functions.

References

Blake, C. L., & Merz, C. J. (1998). *UCI repository of machine learning databases.* (http://www.ics.uci.edu/~mlearn/MLRepository.html)

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and Regression Trees.* Monterey, CA: Wadsworth and Brooks.

Chen, Y., Nixon, M., & Thomas, D. (1997). On texture classification. *Int'l J. Systems Science*, *28*(7), 669-682.

Duch, W. (2002). *Datasets used for classification: comparison of results.* (http://www.phys.uni.torun.pl/kmk/projects/datasets.html)

Evans, e. a. (1993). *Applied pharmacokinetics. principles of therapeutic drug monitoring. applied therapeutics.*

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning. data mining, inference and prediction.* Springer-Verlag.

Gibson, G. J., Siu, S., & Cowan, C. F. N. (1991). The application of nonlinear structures to the reconstruction of binary signals. *IEEE Trans. Signal Proc.*, *39*(8), 1877–1884.

Greenspan, H., Godman, R., Chellappa, R., & Anderson, C. (1994). Learning texture-discrimination rules in a multiresolution system. *IEEE Trans. Pattern Analysis Machine Intelligence*, *16*(9), 894-901.

Haykin, S. (1999). *Neural networks: A comprehensive foundation.* Englewood Cliffs, NJ: Prentice Hall.

Jain, A. K., & Karu, K. (1996). Learning texture discrimination masks. *IEEE Trans. Trans. Pattern Analysis Machine Intelligence*, *18*(2).

Jeruchim, M. C., Balaban, P., & Shanmugan, K. S. (1992). *Simulation of communication systems.* New York: Plenum Press.

Kim, K. I., Jung, K., Park, S. H., & Kim, H. J. (1999). Supervised texture segmentation using support vector machines. *IEE Electronics Letters*, *35*(22), 1935-1936.

Kim, K. I., Jung, K., Park, S. H., & Kim, H. J. (2002). Support vector machines for texture classification. *IEEE Trans. Pattern Analysis Machine Intelligence*, *24*(11), 1542-1550.

Kung, S. Y. (1995). *Digital neural networks.* Prentice Hall.

Kung, S. Y., & Mao, W. (1991). Competition-based supervised learning algorithm for nonlinear discriminant functions. En *International conference on acoustics, speech, and signal processing, ICASSP91* (Vol. 2, p. 1073 - 1076).

Kung, S. Y., & Taur, J. S. (1992). Hierarchical perceptron (HiPer) networks for signal/image classifications. En *Proceedings of the 1992 IEEE-SP workshop on neural networks for signal processing* (Vol. 1, p. 267 - 278).

Kung, S. Y., & Taur, J. S. (1995). Decision-based neural networks with signal/image classification applications. *IEEE Transactions on Neural Networks*, *6*(1), 170-181.

Li, S., Kwok, J. T., Zhu, H., & Wang, Y. (2003). Texture classification using the support vector machines. *Pattern Recognition*, *36*(3), 2883-2893.

Lin, S.-H., & Kung, S. (1995). Probabilistic DBNN via expectation-maximization with multi-sensor classification applications. En *Proceedings of the international conference on image processing* (Vol. 3, p. 236 - 239). Washington, DC USA.

Muhamad, A. K., & Deravi, F. (1994). Neural networks for the classification of image texture. *Eng. Applications of Artificial Intelligence*, *7*, 381-393.

Patel, D. (1996). Page segmentation for document image analysis using a neural network. *Opt. Eng.*, *35*(7), 1-7.

Randen, T., & J.H. Husoy, J. H. (1999). Filtering for texture classification: A comparative study. *IEEE Trans. Pattern Analysis Machine Intelligence*, *21*(4), 291-310.

Rumelhart, D. E., McClelland, J. L., & PDP Research Group the. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 1). Cambridge, MA: MIT Press.

Taur, J. S., & Kung, S. Y. (1993). Fuzzy-decision neural networks. En *IEEE international conference on acoustics, speech, and signal processing* (Vol. 1, p. 577 - 580).

Tuceryan, M., & Jain, A. K. (1993). *Handbook pattern recognition and computer vision.* World Scientific.

Vapnik, V. N. (1998). *Statistical learning theory.* New York: John Wiley & Sons.