computing@tanet.edu.te.ua
www.tanet.edu.te.ua/computing

ISSN 1727-6209
International Scientific Journal of Computing

MULTIPLE NEURAL NETWORK MODELS GENERATOR WITH COMPLEXITY ESTIMATION AND SELF-ORGANIZATION ABILITIES

El-Khier Bouyoucef, Abdennasser Chebira, Mariusz Rybnik, Kurosh Madani

Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956), Senart Institute of Technology, University PARIS XII, Av. Pierre Point, F-77127 Lieusaint, France, {madani, chebira}@univ-paris12.fr, http://www.univ-paris12.fr/

Abstract: *In this article we present a self-organizing hybrid modular approach that is aimed at reduction of processing task complexity by decomposition of an initially complex problem into a set of simpler sub-problems. This approach hybridizes Artificial Neural Networks based artificial intelligence and complexity estimation loops in order to reach a higher level intelligent processing capabilities. In consequence, our approach mixtures learning, complexity estimation and specialized data processing modules in order to achieve a higher level self-organizing modular intelligent information processing system. Experimental results validating the presented approach are reported and discussed.*

Keywords - *Artificial Neural Networks, Complexity Estimation, Self-Organization, Intelligent Decomposer, Universal Information processing.*

1. INTRODUCTION

In a very large number of cases dealing with real world dilemmas and applications (system identification, industrial processes, manufacturing regulation, optimization, decision, pattern recognition, systems, plants safety, etc), information is available as data stored in files (databases etc.). So, the efficient data processing becomes a chief condition to solve problems related to above-mentioned areas. In the most of those cases, processing efficiency is closely related to several issues among which are:

- Data nature: including data complexity, data quality and data representative features.
- Processing technique related issues: including model choice, processing complexity and intrinsic processing delay.

Data complexity (frequently related to nonlinearity or subjective nature of data) may affect the processing efficiency. Quality (noisy data, etc.) may influence processing success and expected results quality. Representative features concerning scarcity of pertinent data could affect processing achievement or resulted precision. On the other hand, choice or availability of appropriated model describing the behaviour related to data to process is of major importance. Processing technique or algorithms' complexity (designing, precision, etc.) shapes the processing effectiveness. Intrinsic processing delay or processing time, related to the implementation issues (software or hardware related issues) or processing models parameterization could affect not only processing quality (results quality) but also the technique's viability to offer an adequate solution for a complex problem. Of course, unfortunately real world and industrial problems are never as comfortable as could be "toy problems". They are often complex problems with a large number of parameters (which should be considered). That's why conventional solutions (based on mathematical and analytical models) reach serious limitation for solving this category of dilemmas.

One of the key points on which one can act is the complexity reduction. Complexity reduction could act not only at problem's solution level but also at processing procedure's level. An issue could be model complexity reduction by splitting a complex problem into a set of simpler problems: this leads to "multi-modelling" where a set of simple models is used to sculpt a complex behaviour [1]. Another promising approach to reduce complexity takes advantage from hybridization [2]. Several Artificial Neural Networks (ANN) based approaches were suggested allowing complexity and computing time reduction. Among proposed approaches, one can note the Intelligent Hybrids Systems [2], Neural Network Ensemble concept [3], Models or experts mixture ([4], [5]), Dynamic Cell Structure architecture [6] and active learning approaches [7].

In this paper, we present an ANN based data driven treelike Multiple Model generator, that we called T-DTS (Treelike Divide To Simplify), able to reduce complexity on both data and processing chain levels. The main idea of T-DTS is based on the notion "Divide et impera"1 (Julius Caesar), transformed here as "Divide To Simplify" (DTS) [8]. The purpose is based on the use of a set of small and specialized mapping neural networks, called Neural Network based Models (NNM), supervised by a Scheduling Unit (SU). Scheduling Unit could be a prototype based neural network, Markovian decision process, etc.. Leafs of the obtained tree are Artificial Neural Networks (models). At the node's level, the input space is decomposed, using complexity estimation based decomposer, into a set of sub-spaces of smaller sizes. While, at the leaf's level the aim is to learn the relations between inputs and outputs in sub-spaces, obtained from splitting. Combination of complexity estimation, splitting and learning capabilities confers to the issued intelligent system self-organizing ability.

The paper is organized in following way. Section 2 will present the Tree-like multiple neural network models generator. Different parameters related to this structure will be presented and discussed. Scheduling function, related dilemmas and proposed techniques will be introduced and discussed. Section 3 will be dedicated to the splitting dilemma and complexity estimation. Splitting strategies and associated complexity estimation techniques are exposed in section 4. Section 5 will be dedicated to implementation and validation issues. Experimental results, validating the presented concept will be presented and discussed. The same section will analyze the efficiency of such approach on the basis of a classification benchmark including a set of databases representative of a set of classification problems with gradually increasing difficulty. In the same section, T-DTS is also applied to real industrial problem: a drilling rubber process used in manufacturing industry. Finally, the last section will conclude the present paper.

2. TREE-LIKE MULTIPLE NEURAL NETWORK MODELS GENERATOR

Tree-like multiple neural network models generator, that we called "Treelike Divide To Simplify" (T-DTS) is a data driven neural networks based Multiple Processing (multiple model) structure designed to reduce complexity on both data and processing chain levels. T-DTS and associated algorithm construct a tree-like evolutionary neural architecture automatically where nodes, called also "Splitting Units" (SU), are decision units, and leafs, called also "Neural Network based Models" (NNM), correspond to neural based processing units.

The T-DTS includes two main operation modes. The first is the learning phase, when T-DTS system decomposes the input data and provides processing sub-structures and tools for decomposed sets of data. The second phase is the operation phase. There could be also a pre-processing phase at the beginning, which arranges (prepare) data to be processed. Pre-processing phase could include several steps (conventional or neural stages). Figures 1 gives the general bloc diagram of T-DTS operational steps. As shows this figure, T-DTS could be characterized by three main operations: "data pre-processing", "learning process" and "generalization process" (called also "operation phase").

The first one performs data pre-processing in order to ease the processing of data. This step could operate with each of two other T-DTS's operational modes. During pre-processing several operations such as data normalizing, data scaling, data dimensionality reduction could be performed. Pre-processing could also include other kind of operations, as removing outliers or Principal Component Analysis ([9]) to enhance input data quality or eliminate redundancy in data.

The learning phase is an important phase during which T-DTS performs several key operations: splitting the learning database into several sub-databases, constructing (dynamically) a treelike Supervision/Scheduling Unit (SU) and building a set of sub-models (NNM) corresponding to each sub-database. Figure 2 represents the division and NNM construction process bloc diagrams. As this figure shows, after the learning phase, a set of neural network based models (trained from sub-databases) are available and cover (model) the behaviour region-by-region in the problem's feature space. In this way, a complex problem is decomposed recursively into a set of simpler sub-problems: the initial feature space is divided into *M* sub-spaces. For each subspace *k*, T-DTS constructs a neural based model describing the relations between inputs and outputs. If a neural based model cannot be built for an obtained sub-database, then, a new decomposition will be performed on the concerned sub-space, dividing it into several other sub-spaces.

The second operation mode corresponds to the use of the constructed neural based Multi-model system for processing unlearned (work) data. The Operation Phase is depicted by figure 3. In fact, the learning phase could be considered as a self-organizing model generation process, which leads to a set of ANN based models (or processing units) managed by a tree-like Supervisor/Scheduler Unit (SU).

[1] "divide and rule".

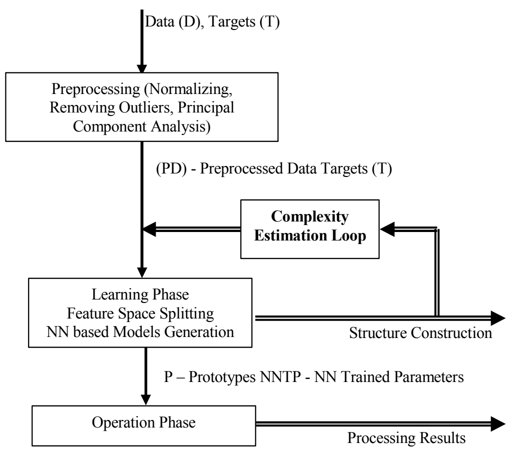
Data (D), Targets (T)
Preprocessing (Normalizing, Removing Outliers, Principal Component Analysis)
(PD) - Preprocessed Data Targets (T)
Complexity Estimation Loop
Learning Phase Feature Space Splitting NN based Models Generation
Structure Construction
P – Prototypes NNTP - NN Trained Parameters
Operation Phase
Processing Results

Fig. 1. General bloc diagram of T-DTS, presenting main operation levels.

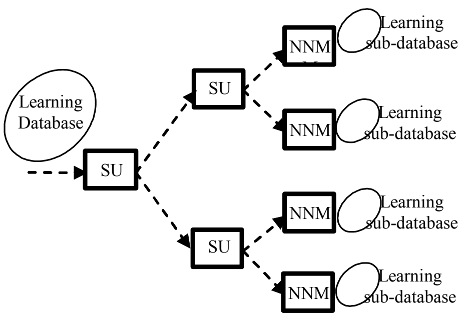

Fig. 2. General bloc diagram of T-DTS learning phase and its tree-like splitting process.

The SU, constructed during the learning phase, receives data (unlearned input vector) and classifies that data (pattern) as corresponding to one of the processing subset. Then, the most appropriated neural processing unit (NNM) is authorized (activated) to process that pattern.

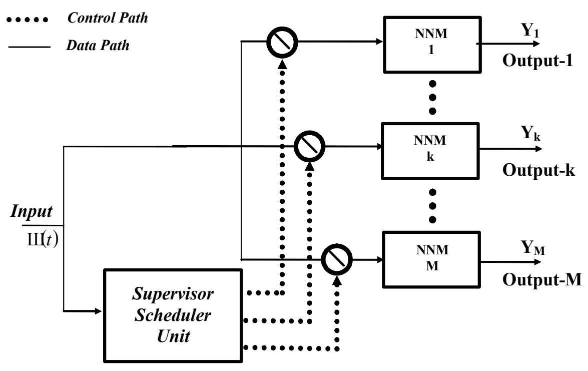

Fig.3 – General bloc diagram of T-DTS generalization phase.

Let Ψ(t) be the input ($\Psi(t) \in \Re^{n_\Psi}$), a n_Ψ-Dimensional vector and $Y_k(t) \in \Re^{n_Y}$ be the k-th ($k \in \{1, \cdots, M\}$) model's output vector of dimension n_y. Let $F_k(.): \Re^{n_\Psi} \to \Re^{n_Y}$ be the k-th NNM's transfer function. Let $S(\Psi(t), p, \xi) \in B^M$, where $B = \{0,1\}$, be the Supervisor/Scheduler Unit's (SSU) output, called also Scheduling Function, which depends on Ψ(t), but which may also depend on some parameters p and/or conditions ξ. p_k represents some particular values of parameter p and ξ_k denotes some particular value of condition ξ, respectively, obtained from learning phase process for the k-th sub-dataset. Taking into account the above-defined notation, the Supervisor/Scheduler Unit's (SU) output could be formalized as relation (1).

$$S(\Psi(t), p, \xi) = \begin{pmatrix} s_1 & \cdots & s_k & \cdots & s_M \end{pmatrix}^T \tag{1}$$

$$\text{with} \begin{bmatrix} s_k = 1 & \textit{if } p = p_k \textit{ and } \xi = \xi_k \\ s_k = 0 & \textit{else} \end{bmatrix}$$

The scheduling vector S(Ψ, p_k, C_k) will activate (select) the k-th NNM, and so the processing of an unlearned input data conform to parameter p_k and condition C_k will be given by the output of the selected NNM (relation (2)).

$$Y(\Psi, t) = Y_k(t) = F_k(\Psi(t)) \tag{2}$$

It is important to emphasize that the SU uses a complexity indicator in order to handle the splitting process. So, complexity estimation is among the most important operations performed by TDTS. The next section introduces and describes this major point in the T-DTS paradigm.

3. SPLITTING AND COMPLEXITY ESTIMATION

The goal is to estimate the processing task's difficulty and modify the splitting (initial complex problem's division into a set of sub-problems with reduced complexity) and processing algorithms (models) in order to handle the whole task more efficiently. The modification may include among others:

- task decomposition up to some degree dependant on task or data complexity,
- choice of appropriate processing structure (i.e. appropriated model) for each subset of decomposed data,
- choice of processing architecture (i.e. models parameters).

These concepts are quite different and complicated so we focus only on the first aspect:

measurements supporting task decomposition.

Complexity estimation is not a very popular subject. These methods are used to evaluate a problem's processing complexity level. We will present the methods that compute indirectly Bayes error, non-parametric Bayes error estimation methods and other methods.

CHERNOFF BOUND

The Bayes error for the two class case can be expressed by relation (3) where **c_k** represent the class *k* and *x* is the feature vector.

$$\varepsilon = \int \min_k [P(c_k)p(x \mid c_k)]dx \tag{3}$$

Through modifications, we can obtain a Chernoff bound [10] ε_u, which is an upper bound on ε for the two class case. The tightness of bound varies with the parameters.

$$\varepsilon_u = P(c_1)^s P(c_2)^{1-s} \int p(x^s \mid c_1) p(x^{1-s} \mid c_2) dx \tag{4}$$

BHATTACHARYYA BOUND

The Bhattacharyya bound [11] is a special case of Chernoff bound for s = 1/2. Empirical evidence indicates that optimal value for Chernoff bound is close to 1/2 when the majority of separation comes from the difference in class means. Under a Gaussian assumption, the expression of the Bhattacharyya bound is expressed by (5).

$$\varepsilon_b = \sqrt{P(c_1)P(c_2)}\, e^{-\mu(1/2)} \tag{5}$$

where:

$$\mu(1/2) = \frac{1}{8}(\mu_2 - \mu_1)^T \left[\frac{\Sigma_1 + \Sigma_2}{2}\right]^{-1} (\mu_2 - \mu_1) + \frac{1}{2}\ln \frac{\left|\frac{\Sigma_1 + \Sigma_2}{2}\right|}{\sqrt{|\Sigma_1||\Sigma_2|}} \tag{6}$$

μ_i and Σ_i are respectively the means and covariance of classes *k* in {1,2}.

DIVERGENCE

Measure of divergence [12] (separability) is related to verisimilitude ratio. Verisimilitude ratio L_{12} between two classes' c_1 and c_2 is defined as:

$$L_{12}(X) = \frac{P(X \mid c_1)}{P(X \mid c_2)} \tag{7}$$

Computing the divergence is significantly simplified when distributions of variables are normal. In that case divergence equals:

$$D_{12} = \tfrac{1}{2} tr \left[(\Sigma_1 - \Sigma_2)(\Sigma_2^{-1} - \Sigma_1^{-1})\right] + \tfrac{1}{2} tr \left[(\Sigma_1^{-1} + \Sigma_2^{-1})(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T\right] \tag{8}$$

where *tr* signifies trace of a matrix, μ_1 and μ_2 class means, $\sum_1$ and $\sum_2$ class covariance matrices.

Divergence measures the degree of separability between two classes. Therefore in order to evaluate multi class case one should count an average of all two-element combination of classes. Computational cost of divergence is significant.

JEFFRIES-MATUSITA DISTANCE

Jeffries-Matusita [13] distance between classes c_1 and c_2 is defined by relation (9).

$$JM_D = \int_x \left\{\sqrt{p(X \mid c_2)} - \sqrt{p(X \mid c_1)}\right\}^2 dx \tag{9}$$

If c_1 and c_2 distributions are normal Jeffries-Matusita distance reduces to:

$$JM_D = 2\left(1 - e^{-\alpha}\right) \quad \text{with}$$

$$\alpha = \frac{1}{8}(\mu_2 - \mu_1)^T \left(\frac{\Sigma_2 + \Sigma_1}{2}\right)^{-1} (\mu_2 - \mu_1) + \frac{1}{2}\log_e\left(\frac{\det \Sigma}{\det \Sigma_1 - \det \Sigma_2}\right) \tag{10}$$

Matusita distance is bounded within range [0, 2] where high values signify high separation between classes' c_1 and c_2.

PARZEN ESTIMATION

Parzen [14] techniques relay on the same concept as k-NN: setting a local region *Γ(x)* around each sample *x* and examining the ratio of the samples enclosed *k*, to the total number of samples *N*, normalized with respect to region volume *v*:

$$\hat{p}(x) = \frac{k}{vN} \tag{11}$$

The density estimation Equation becomes:

$$\hat{p}(x) = \frac{k(x)}{vN} \tag{12}$$

where *p(x)* represents density and *k(x)* represents number of samples enclosed in volume.

BOUNDARY METHODS

The boundary methods are described in the work of Pierson [15]. Data from each class is enclosed within a boundary of specified shape according to some criteria. The boundaries can be generated using general shapes like: ellipses, convex hulls, splines and others. The boundary method often uses ellipsoidal boundaries for Gaussian data, since it is a natural representation of those. Since most decision boundaries pass through overlap regions, a count of these samples may give a measure related to misclassification rate. Pierson has demonstrated that the measure of "Separability" called the Overlap

Sum is directly related to Bayes error with a much more simple computational complexity. It does not require any exact knowledge of the *a posteriori* distributions. Overlap Sum is the arithmetical mean of overlapped points with respect to progressive collapsing iterations:

$$O_S(mt_0) = \frac{1}{N}\sum_{k=1}^{m}(kt_0)\Delta s(kt_0) \tag{13}$$

where t_o is the step size, m is the maximum number of iteration (collapsing boundaries), N is the number of data points in whole dataset and $\Delta s(kt_0)$ is the number of points in the differential overlap. Boundary methods provide a measure of class separability, the overlap sum (OS), which is strongly correlated with Bayes error and easily computed. These properties suggest that BMs can be used as an alternative to traditional Bayes error estimation techniques.

INTER-INTRA CLUSTER DISTANCE

The average inter-cluster [16] distance is computed by considering all data in both clusters (classes):

$$S_{IC} = \frac{l_1 S_{wi} + l_2 S_{wj}}{l_1 + l_2} \tag{14}$$

where l_1 and l_2 are the numbers of samples in the two clusters and S_{wi} , S_{wj} represent consequently inter-cluster distance of cluster *i* and inter-cluster distance of cluster *j*.

4. COMPLEXITY ESTIMATION AND SPLITTING STRATEGIES

The complexity estimation based splitting could be performed according to two general strategies: “static” splitting strategy and “adaptive” (dynamic) one. In both cases, the issued could be binary or multiple branches tree-like structure. The main difference between two strategies remains in nature of the complexity estimation indicator and the splitting decision operator used in splitting process.

STATIC COMPLEXITY ESTIMATION STRATEGY

By “Static” we mean “not flexible” regarding different features influencing the processing task complexity. The static nature of splitting strategy could appear on complexity estimation method’s level and on splitting decision’s level.

For example, empirically determined maximum standard deviation (MaxStd: representing global data similarity) could be an interesting candidate. In this case, if the data of a given sub-database is homogenous enough then the associated complexity will be considered as “low” (no necessary to divide the database). The splitting process starts by evaluating the average and standard deviation of the learning database. If the obtained standard deviation is greater that the MaxStd, then, a two-clusters Kohonen SOM (or a distance based competitive NN) divides the learning database into two sub-databases. These operations are repeated until the standard deviation relative to each created sub-database doesn’t exceed the MaxStd value.

If Ψ_j^k represents the j-th learning prototype of the k-th learning sub-database, if $\overline{\Psi}_k$ denotes average representative prototype of this sub-database and $\square_k$ it’s standard deviation (obtained from relations (15)), then, a threshold based decision will determine if the concerned database should be divided (or not) into two (or more) sub-databases.

$$\overline{\Psi}_k = \frac{1}{n_k}\sum_{j=1}^{n_k}\Psi_j^k \quad \text{and} \quad \sigma_k = \frac{1}{n_k}\sum_{j=1}^{n_k}\left(\Psi_j^k - \overline{\Psi}_k\right)^2 \tag{15}$$

$$MaxStd = Max\ (\sigma_k) \tag{16}$$

The principle is based on the following assumption: more dissimilar the problem’s representative data (learning database) is, more complex will be the needed processing effort. So, after a global data similarity estimation (giving a rough estimation of the treated problem’s complexity), a threshold based decision will determine if the concerned database should be divided (or not) into two (or more) sub-databases.

$$\left[\begin{array}{lll}\textbf{If}\ \sigma_k > MAXSTD & \textbf{Then} & \text{"Split" the current database} \\ \quad\textbf{Else} & \textbf{Generate} & \text{an ANN based model}\end{array}\right. \tag{17}$$

ADAPTIVE COMPLEXITY ESTIMATION STRATEGY

By “Adaptive” we mean “flexible” regarding different features influencing the processing task complexity. Concerning data complexity, it could be seen from Bayes error estimation angle, which basically reflects the “data classification” task’s difficulty. However, direct estimation of the Bayes error is a difficult task in practice. That’s why in practice, indirect approaches or approximation techniques are used to approach the Bayes error. There are two general ways to estimate Bayes error: the first, belonging to indirect way, consists to measure lower or higher bound of this error, which are easier to compute than direct estimation of the Bayes error, the second is to estimate this error by non-parametric method. Some other methods could be used based on space partitioning or other heuristically obtained hypothesis (rules). Figure 4 gives taxonomy of “Classification Complexity Estimation” methods.

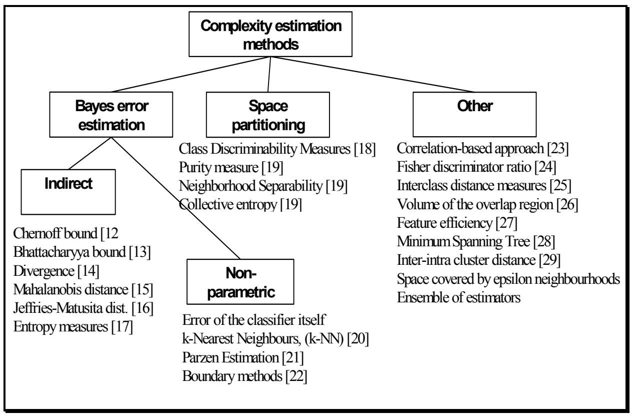
Complexity estimation methods
Bayes error estimation
Space partitioning
Other
Indirect
Non-parametric
Chernoff bound [12
Bhattacharyya bound [13]
Divergence [14]
Mahalanobis distance [15]
Jeffries-Matusita dist. [16]
Entropy measures [17]
Class Discriminability Measures [18]
Purity measure [19]
Neighborhood Separability [19]
Collective entropy [19]
Error of the classifier itself
k-Nearest Neighbours, (k-NN) [20]
Parzen Estimation [21]
Boundary methods [22]
Correlation-based approach [23]
Fisher discriminator ratio [24]
Interclass distance measures [25]
Volume of the overlap region [26]
Feature efficiency [27]
Minimum Spanning Tree [28]
Inter-intra cluster distance [29]
Space covered by epsilon neighbourhoods
Ensemble of estimators

Fig.4. Taxonomy of Classification Complexity Estimation methods.

Indirect Bayes error estimation methods are well established theoretically but they have some drawbacks: they assume data normality, construction of models could be time consuming, model verification could be difficult, they are susceptible to data dimensionality and finally a large number of samples may be needed to estimate accurately class conditional probabilities. Non-parametric Bayes error estimation methods make no assumptions about the specific distributions involved. They use some intuitive methods. Their drawback is due to computing cost. *Measures related to space partitioning* are connected to space partitioning algorithms: specific space partitioning algorithms that divide the feature space into sub-spaces. They also suffer from high computing cost.

CHOICE OF THE ADAPTIVE COMPLEXITY ESTIMATION STRATEGY

Taking into account the above-discussion, we have designed a specific benchmark to investigate adaptive complexity estimation strategy choice. The benchmark is based on classification problem and has been defined in following way: two databases of 1000 vectors each, represent three different problems with gradually increasing classification difficulties (see Figure 5). In the first database, data is distributed according to“circle” geometry (symmetrical). In the second database, data distribution acts conformably to a two spirals-like geometry. Each database contains data from two classes. Three databases of 1000 vectors each, each database contains data from two classes. Each database is divided into two equal parts (learning and generalization databases) of 500 vectors each. Databases are normalized (to obtain mean equal to 0 and variance equal to 1).

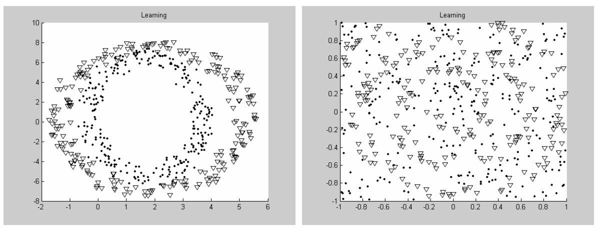

Fig.5 – Benchmark problems with gradually increasing classification complexity.

Decomposition is performed by competitive network of 2 neurons. The maximum number of epochs has been set up to 100 epochs and learning rate to 0.1. The splitting criterion is the following: “**If** (threshold is >= measure) **then** the database is divided”. Issued databases (with reduced complexity) are then used to train a set of linear networks to minimize the sum of squared error for learning database. The output of each linear network is rounded to nearest integer number in order to obtain categorized values (classes). The observed values are number of generated models and number of correctly classified prototypes in generalization phases. Figures 6 and 7 give the above mentioned characteristics for each benchmark database.

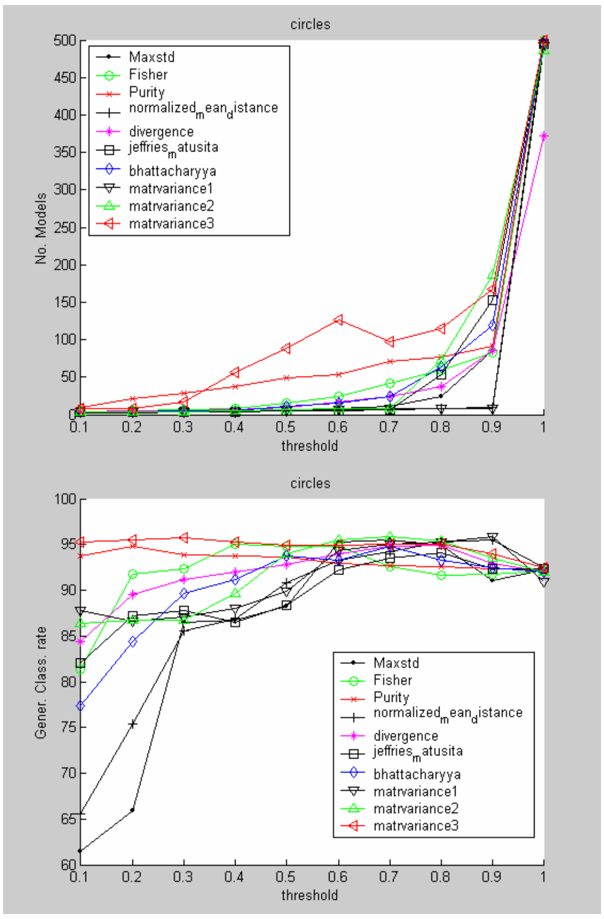

Fig.6 – Observed values versus threshold for the circular distribution data: number of generated models (upper) and correct classification rate in generalization phase (lower).

One can remark that for both cases the best classification rate is obtained when the decomposition (splitting) is decided using "purity" measurement [17] based complexity indicator. However, in the same time, "Fisher's" discriminator based complexity estimation [18] achieves performances close to the previous one. Regarding the number of generated models, the first complexity estimation indicator (purity) leads to much greater number of models. Finally, "low computational cost" criteria should also be taken into account. In this context we have chosen Fisher discriminator ratio as complexity estimation indicator.

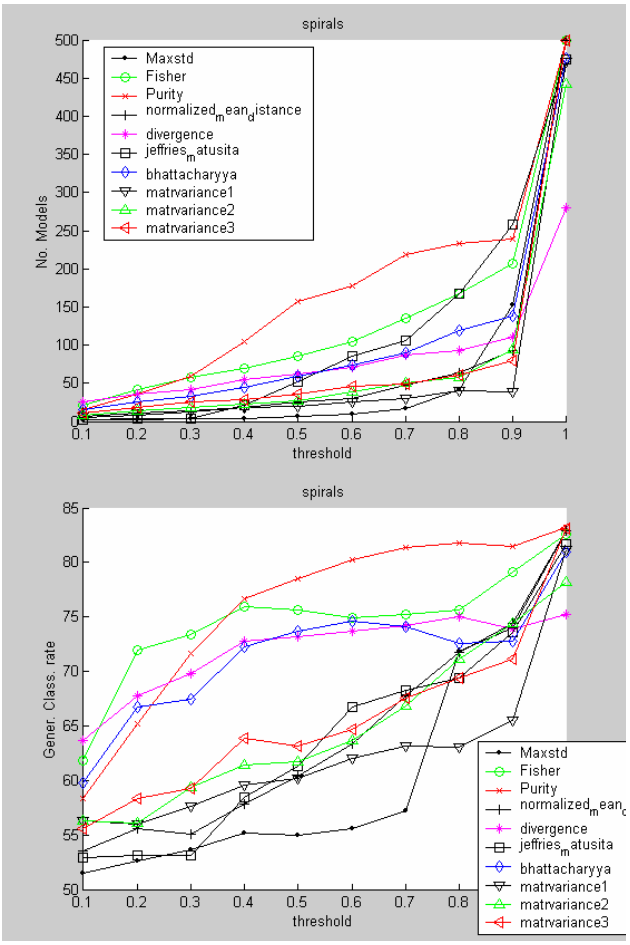

Fig.7 – Observed values versus threshold for the two-spiral-like distribution data: number of generated models (upper) and correct classification rate in generalization phase (lower).

The measure is calculated in each dimension separately and afterwards the maximum of the values is taken. To use it for more than two class problem it is necessary to compute Fisher discriminator for each two-element combination of classes and later average it. Important features of this measure are: its strong relation with the structure of data and low computational requirements. Thus it can be efficiently used as decomposition criterion (complexity estimation criteria). In fact, as the Fisher's discriminator based complexity estimation indicator measures distance between two classes (versus the averages and dispersions of data representative of each class), it could be used to adjust the splitting decision proportionally to processing's difficulty: a short distance between two classes (of data) reflects higher difficulty, while, well separated classes (high distance) could be interpreted as needing a lower processing complexity.

5. IMPLEMENTATION AND VALIDATION RESULTS

A software implementation of T-DTS including different splitting strategies, different complexity estimation methods and different ANN models has been achieved. The implementation has been performed under MathLab environment. A specific validation benchmark database including a set of databases with gradually increasing complexity has been designed. The problems difficulty starts from linearly separable classification (qualified as easy) and ends with two spirals problem (qualified as the hardest): the easiest problem is labelled as problem **1** and the most complex one is labelled as problem **12**.

Based on the above-presented set of benchmark problems, two self-organizing modular systems have been generated in order to solve the issued classification problem. The first one using the "static" complexity estimation method with a threshold based decomposition decision rule, and the second one, using "adaptive" complexity estimation criterion based on Fisher's decimator (figure 8).

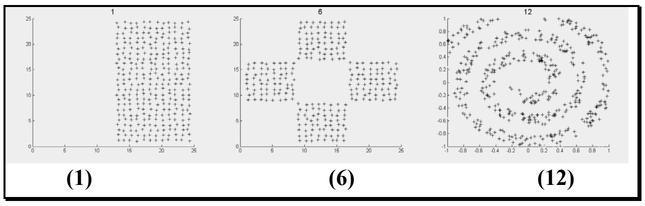

Fig.8 – Benchmark: databases examples with gradually increasing complexity.

Figures 9 gives classification results obtained for each both static and dynamic cases. One can note from the figure 9 that in the case of the static strategy based splitting, the classification rate drops significantly for more complicated datasets. That proves: when databases complexity is increasing such modular system cannot maintain the processing quality.

When the splitting is performed on the basis of Fisher's complexity estimation based indicator, there is only a small dropping tendency of the classification rate when the classification's difficulty increases. However, in this case, processing time (essentially the learning phase) increases significantly for more complex datasets.

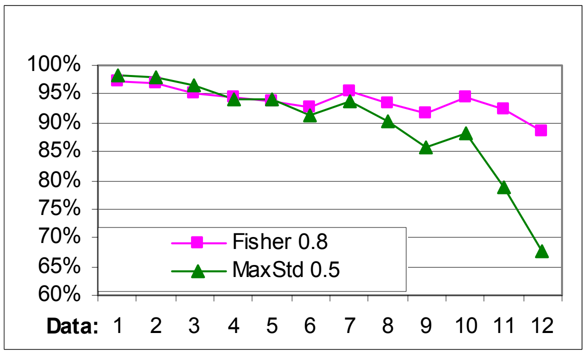

Fig.9 – Results for modular structure with static and Fisher's discriminator adaptive complexity estimation strategy.

In fact, in this case adaptive structure adapts decomposition (and so, the modularity) in order to reduce processing complexity, by creating the number of models proportionally to the processed data's complexity.

APPLICATION TO INDUSTRIAL NON LINEAR PROCESS IDENTIFICATION

The validation has been performed using Self Organization Map (SOM) [19] as SU. In this case, splitting process dividing the initial complex problem into M reduced sub-problems takes advantage from Kohonen SOM properties (Similarity Matching). The activation of an appropriate NNM will be issued from similarity measure between an unlearned input vector Ψ(t) and the k-th SOM cluster (W_k). If the initial feature space has been decomposed into M clusters by a Kohonen like SOM process, then, the Scheduling vector (SU output) will be conform to relation (18).

$$s_k(\text{Щ}\, W_k) = \begin{matrix} 1 & if & |\text{Ш}(t) \quad W_k| = \underset{M}{Min}(|\text{Ш}(t) \quad W_k|) \\ 0 & & Else \end{matrix} \qquad (18)$$

In case when Kohonen maps have a grid 2x1 topology, T-DTS builds a binary decision tree. The implemented splitting optimization criterion (loop) is quite simple. The parameter is MaxStd, which defines the standard deviation maximum value (in each dimension) in a given subset. Concerning Neural Networks based models (processing units) they are MLP-like (Multi-Layers Perceptron) units.

We have applied T-DTS based Identifier to a real world industrial process control problem. The process is a drilling rubber process used in plastic manufacturing industry. Several non-linear parameters influence the manufacturing process. To perform an efficient control of the manufacturing quality (process quality), one should identify the global process.

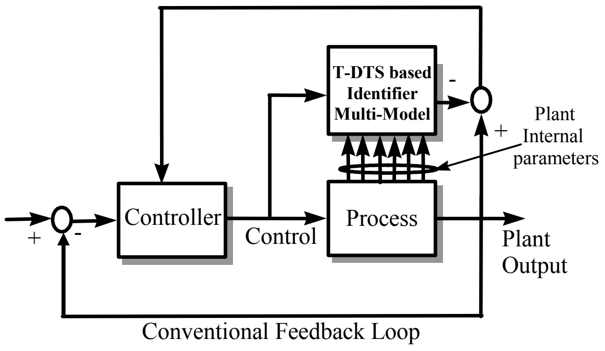
T-DTS based Identifier Multi-Model
-
+
Plant Internal parameters
+
-
Controller
Control
Process
Plant Output
Conventional Feedback Loop

Fig. 10. Implemented industrial processing loop using a Kohonen DU based T-DTS identifier.

Kohonen SOM based Scheduling Unit (SU) uses a 4x3 grid leading to 12 feature sub-spaces. So, 12 Neural Network based Models (NNM) have been generated and trained (from learning database). Figure 11 shows the learning phase validation presenting the identified output . Figure 12 shows system output in the generalization phase. One can conclude that estimated output is in accord with the measured one.

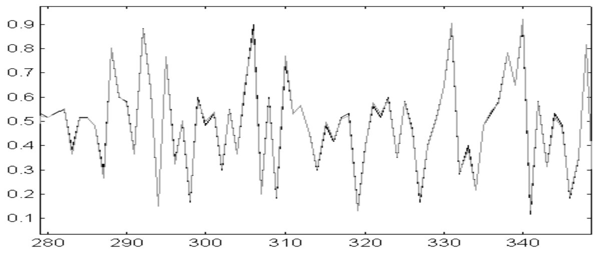

Fig. 11. Real plant's and T-DTS based identification issued outputs after the learning process.

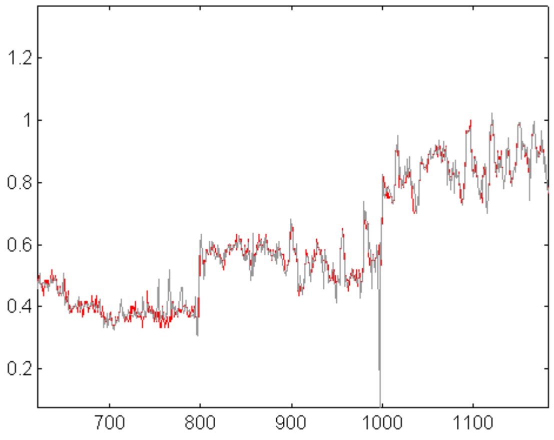

Fig. 12. Prediction of an unlearned plant's output sequence (generalization phase).

On the one hand, by dividing the initial problem into several sub-problems with reduced sizes, the proposed Tree-like multiple neural network models generator (T-DTS) simplifies the learning complexity and so, the duration time. On the other hand, by embedding "complexity estimation" capabilities (as regulation loops) at the splitting decision level, T-DTS obtains smart self-organization capabilities, acquiring some processing universality potential. Finally, it decreases globally implementation, parameterization and models parameters optimization constraints. Moreover, the neural character of the proposed structure makes it adaptable for different kinds of applications, offering solution to a wide range of complex processing problems.

6. CONCLUSION

Very promising results, obtained for the presented benchmark set of classification problems, show efficiency of such multiple model structure to enhance processing capability by reducing complexity on both processing and data levels. Moreover, the neural nature of generated models and complexity estimation based self-organization ability of our approach lead to additional attractive features which are adaptability, modularity and universality, opening new dimensions in bio-inspired artificial intelligence. T-DTS concept has successfully been applied for real world industrial plant identification (a chief step in adaptive control of complex systems). We are currently working on the splitting criterion based on advanced complexity measurement techniques.

7. REFERENCES

[1]. *Multiple Model Approaches to Modeling and Control*, edited by R. Murray-Smith and T.A. Johansen, Taylor & Francis Publishers, 1997, ISBN 0-7484-0595-X.

[2]. S. Goonatilake and S. Khebbal, "Intelligent Hybrid Systems: Issues, Classification and Future Directions", in *Intelligent Hybrid Systems*, John Wiley & Sons, pp 1-20, ISBN 0 471 94242 1.

[3]. Krogh A., Vedelsby J.: Neural Network Ensembles, Cross Validation, and Active Learning, *in Adv in Neural Inf Processing Syst. 7*, The MIT Press, Ed by G. Tesauro, pp 231-238, 1995.

[4]. Sridhar D.V.,Bartlett E.B., Seagrave R.C., "An information theoretic approach for combining neural network process models", *Neural Networks*, Vol. 12, pp 915-926, Pergamon, Elsevier, 1999.

[5]. Jordan M. I. and Xu L., "Convergence Results for the EM Approach to Mixture of Experts Architectures", *Neural Networks*, Vol. 8, N° 9, pp 1409-1431, Pergamon, Elsevier, 1995.

[6]. Bruske J., Sommer G., Dynamic Cell Structure, *Adv in Neural Inf Processing Systems 7*, The MIT Press, Ed by G. Tesauro, pp 497-504, 1995.

[7]. Sang K. K. and Niyogi P., Active learning for function approximation, *in Neural Information Processing Systems 7*, The MIT Press, Ed by G. Tesauro, pp 497-504.

[8]. Madani K., Chebira A., "*A Data Analysis Approach Based on a Neural Networks Data Sets Decomposition and it's Hardware Implementation*", PKDD 2000, Lyon, France, 2000.

[9]. Jollifee I.T., "*Principle Component Analysis*", New York, Springer Verlag 1986.

[10]. Chernoff, "Estimation of a multivariate density", *Annals of the Institute of Statistical Mathematics*, vol. 18, pp. 179-189, 1966.

[11]. Bhattacharya, "On a measure of divergence between two statistical populations defined by their probability distributions", *Bulletin of Calcutta Maths Society*, vol. 35, pp. 99-110, 1943.

[12]. J. Lin, "Divergence measures based on the Shannon entropy", *IEEE Transactions on Information Theory*, 37(1):145-151, 1991.

[13]. K. Matusita, "On the notion of affinity of several distributions and some of its applications", *Annals Inst. Statistical Mathematics*, 19:181-192, 1967.

[14]. E. Parzen, "On estimation of a probability density function and mode", *Annals of Math. Statistics*, vol. 33, pp. 1065-1076, 1962.

[15]. W.E. Pierson, "*Using boundary methods for estimating class separability*", PhD Thesis, Dept of Elec. Eng., Ohio State Univ., 1998.

[16]. Kohn, L. G. Nakano, and V. Mani, "A class discriminability measure based on feature space partitioning", *Pattern Recognition*, 29(5):873-887, 1996.

[17]. Singh S., "Pattern Recognition Using Information Slicing model", *16th Internat. Conf. on Pattern Recog. (ICPR 2002)*, 11-15 August 2002, Quebec, Canada, IEEE Computer Society, ISBN 0-7695-1695-X, 2002.

[18]. Fisher, "*The mathematical theory of probabilities*", John Wiley, 2000.

[19]. T. Kohonen, "*Self-Organization and Associative Memory*", 2-nd Ed., Springer-Verlag, New York, 1988.

El Khier Bouyoucef
Received his Master of Telecommunication Science degree from UVHC University, Valenciennes, France, in 2003. Since octobre 2003 he works as Ph.D student in Images, Signals and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII – Val de Marne University. His research works deals

with data processing, artificial learning, complexity estimation methods, classification techniques and hybrid neural based information processing systems.

Dr. Abdennasser Chebira
Received his Ph.D. degree in Electrical Engineering and Computer Sciences from PARIS XI University, Orsay, France, in 1994. Since September 1994 he works as Professor Assistant at Senart Institute of Technology of PARIS XII – Val de Marne University. He is a staff researcher at Images, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of this University. His current research works concern self-organizing neural network based multi-modeling, hybrid neural based information processing systems; Neural based data fusion and complexity estimation.

Dr. Mariusz Rybnik
Received his Master of Computer Science degree from Bialystok Technical University, Bialystok, Poland, in 2001. Received his Ph.D. degree in December 2004 from PARIS XII – Val de Marne University, Créteil, France. He worked as Assistant at Images, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of this University until October 2005. His research work deals with self-organizing neural network based multi-modeling and hybrid neural based information processing systems.

***Prof. Kurosh Madani**:*
Received his Ph.D. degree in Electrical Engineering and Computer Sciences from University PARIS XI (PARIS-SUD), Orsay, France, in 1990. From 1989 to 1990, he worked as assistant professor at Institute of Fundamental Electronics of PARIS XI University. In 1990, he joined Creteil-Senart Institute of Technology of University PARIS XII – Val de Marne, Lieusaint, France, where he worked from 1990 to 1998 as assistant professor. In 1995, he received the DHDR Doctor Habilitate degree (senior research Dr. Hab. degree) from University PARIS XII – Val de Marne. Since 1998 he works as Chair Professor in Electrical Engineering of Senart Institute of Technology of University PARIS XII – Val de Marne. From 1992 to 2004 he has been head of Intelligence in Instrumentation and Systems Laboratory of PARIS XII – Val de Marne University located at Senart Institute of Technology. Since 2005, he is head of one of the three research teams of Image, Signal and Intelligent Systems Laboratory (LISSI / EA 3956) of PARIS XII University. He has worked on both digital and analog implementation of processors arrays for image processing by stochastic relaxation, electro-optical random number generation, and both analog and digital Artificial Neural Networks (ANN) implementation. His current research interests include large ANN structures behavior modeling and implementation, hybrid neural based information processing systems and their software and hardware implementations, design and implementation of real-time neuro-control and neural based fault detection and diagnosis systems. Since 1996 he is a permanent member (elected Academician) of International Informatization Academy. In 1997, he was also elected as Academician of International Academy of Technological Cybernetics.