An Introduction to Intelligent System Algorithms

G. Papadourakis* and I. Kyriakidis*

* Department of Applied Informatics & Multimedia
Technological Educational Institute of Crete
Heraklion, Greece GREECE
papadour@cs.teicrete.gr, kyriakidis@teicrete.gr

***Abstract*—Huge successes have been achieved through modeling the biological and natural systems to develop new algorithmic models to solve complex problems. Those systems are so-called "intelligent systems". Intelligent Systems include algorithms like artificial neural networks, evolutionary computing, swarm intelligence, and fuzzy systems. These intelligent algorithms, included with logic, deductive reasoning, expert systems, case-based reasoning and symbolic machine learning systems, form part of the field of Artificial Intelligence (AI). Artificial intelligence is the established name for the field that has defined as Computational Intelligence (CI). This paper presents an introduction to some of these intelligent algorithms, under the umbrella of computational intelligence.**

I. EXPERT SYSTEMS

Expert systems [1-2] encode human expertise in limited domains by representing it using if-then rules. Imagine a piece of software that runs on your PC which provides the same sort of interaction and advice as a career counselor helping you decide what education field to go into and perhaps what course to pursue. It can be a piece of software which asks you questions about your defective TV and provides a diagnosis about what is wrong with it. Actually such software exists and called expert systems. Expert systems are a part of the larger area of Artificial Intelligence.

One of the goals of Artificial Intelligence is to develop systems which exhibit "intelligent" human-like behavior. Expert systems are systems which encode human expertise in limited domains. One way of representing this human knowledge is using If-then rules.

A. Components of a Expert System

A typical expert system consists of five components:

1) The user interface.
2) The working memory.
3) The knowledge base.
4) The inference engine.
5) The explanation system.

The knowledge base and the working memory (WM) are the data structures which the system uses and the inference engine is the basic program which is used. The explanation system answers questions the user has and provides an explanation of its reasoning. Each of these components are briefly described below.

B. Working Memory

The working memory represents the set of facts known about the domain. The elements of the WM reflect the current state of the world. In an expert system, the WM typically contains information about the particular instance of the problem being addressed. For example, in a TV troubleshooting expert system, the WM could contain the details of the particular TV being looked at.

The actual data represented in the WM depends on the type of application. The initial WM, for instance, can contain a priori information known to the system. The inference engine uses this information in conjunction with the rules in the knowledge base to derive additional information about the problem being solved.

C. Knowledge Base

The knowledge base (also called rule base when If-then rules are used) is a set of rules which represents the knowledge about each domain. The general form of a rule is:

```
If cond1 and cond2 and cond3 ...
then action1, action2, ...
```

The conditions cond1, cond2, cond3, etc., (also known as antecedents) are evaluated based on what is currently known about the problem being solved (i.e., the contents of the working memory).

Each antecedent of a rule typically checks if the particular problem instance satisfies some condition. For example, an antecedent of a rule in a TV troubleshooting expert system could be: the picture on the TV display flickers.

The consequents of a rule typically alter the WM, to incorporate the information obtained by application of the rule. This could mean adding more elements to the WM, modifying an existing WM element or even deleting WM elements. They could also include actions such as reading input from a user, printing messages, accessing files, etc. When the consequents of a rule are executed, the rule is said to have been fired.

D. Inference Engine

The inference engine is the program part of an expert system. It represents a problem solving model which uses the rules in the knowledge base and the situation-specific knowledge in the WM to solve a problem.

Given the contents of the WM, the inference engine determines the set of rules which should be considered.

These are the rules for which the consequents match the current goal of the system. The set of rules which can be fired is called the conflict set. Out of the rules in the conflict set, the inference engine selects one rule based on some predefined criteria. This process is called conflict resolution. For example, a simple conflict resolution criterion could be to select the first rule in the conflict set.

A rule can be fired if all its antecedents are satisfied. If the value of an antecedent is not known (in the WM) the system checks if there are any other rules with that as a consequent; thus setting up a sub-goal. If there are no rules for that antecedent, the user is prompted for the value and the value is added to the WM.

If a new sub-goal has been set up, a new set of rules will be considered in the next cycle. This process is repeated till, in a given cycle, there are no sub-goals or alternatively, the goal of the problem-solving has been derived.

This inferencing strategy is called backward chaining (since it reasons backward from the goal to be derived). There is another strategy, called forward chaining where the system works forward from the information it has in the working memory. In forward chaining, the conflict set will be created by rules which have all their antecedents true in a given cycle. The system continues till the conflict set becomes empty.

E. Explanation System

Expert systems typically need to be able to provide explanations regarding the conclusions they make. Most expert systems provide a mechanism whereby the user can ask questions about:

- Why a particular question is being asked.
- How the system came to a particular conclusion.

Providing explanations is essential in all non-trivial domains for the user to understand how the system works and determine whether its reasoning is correct or not. Typically the system will keep track of what rules (knowledge) it is using and provide explanations based on a translation of these rules into English.

F. Expert system shells

A shell is a complete development environment for building and maintaining knowledge-based applications. It provides a step-by-step methodology and ideally a user-friendly interface such as a graphical interface, for a knowledge engineer. That allows the domain experts to be directly involved in structuring and encoding the knowledge. Examples of shells include:

F.1 Prolog

Prolog (programming in logic) [3-6] is one of the most widely used programming languages in artificial intelligence research. As opposed to imperative languages such as C or Java (which also happens to be object-oriented) it is a declarative programming language. That means, when implementing the solution to a problem, instead of specifying how to achieve a certain goal in a certain situation, we specify what the situation (rules and facts) and the goal (query) are and let the Prolog interpreter derive the solution for us.

F.2 CLIPS

CLIPS (C Language Integrated Production System) [7] is an expert system tool consisting of a programming language and an interpreting environment. The programming language has a Lisp-like syntax, using prefix notation and parentheses to delimit commands and constructs. Commands are used the way LISP functions are and operate on constructs such as facts and rules etc. As in Lisp, it is essential to remember to balance the parentheses. The language is actually a multi-paradigm language, it provides rule–based, object–oriented and procedural programming all in one package.

The interpreting environment contains an inference engine, a fact base and a rule base. The facts in the fact base can be thought of as data representing the state of the world. The rules are used to store knowledge about how to act in certain situations or knowledge about conclusions to draw from data. Rules have a condition part which describes some state of the world, and an action part which is to be executed when that state of the world is at hand. The action may either be internal (e.g. change the fact base) or external (e.g. printing something or reading information). When the inference engine is run, all the rules in the rule base check the facts in the fact base and rules whose condition part is matched are executed until no more rules can fire.

II. NEURAL NETWORKS

Work on artificial neural networks, commonly referred to as “neural networks”, has been motivated right from its inception by the recognition that the human brain computes in an entirely different way from the conventional digital computer. The brain is a highly complex, nonlinear and parallel computer (information-processing system). It has the capability to organize its structural constituents, known as neurons, so as to perform certain computations (e.g. pattern recognition, perception and motor control) many times faster than the fastest digital computer in existence today. Consider, for example, human vision, which is an information-processing task (Mark, 1982; Levine, 1985; Churchland and Sejnowski, 1992). [8]

In brain theory, the complexities of real neurons are abstracted in many ways to aid in understanding different aspects or neural network development, learning or function. In neural computing (technology based on networks of “neuron-like” units), the artificial neurons are designed as variations on the abstractions of brain theory and are implemented in software or VLSI or other media [8-9].

A. Neural Network Architectures

The manner in which the neurons of a neural network are structured is intimately linked with the learning algorithm used to train the network. In general, it can be identified three fundamentally different classes of network architectures:

A.1 Single-Layer Feedforward Networks

In a layered neural network the neurons are organized in the form of layers. In the simplest form of a layered network, we have as input layer of source nodes that projects onto an output layer of neurons (computation nodes), but not vice versa. In other words, this network is strictly a feedforward or acyclic type. Such a network is

called a single-layer network, with the designation "single-layer" referring to the output layer of computation nodes (neurons). We do not count the input layer of source nodes because no computation is performed there.

A.2 Multilayer Feedforward Networks

The second class of a feedforward neural network distinguishes itself by the presence of one or more hidden layers, whose computation nodes are correspondingly called hidden neurons or hidden units. The function of hidden neurons is to intervene between the external input and the network output in some useful manner. By adding one or more hidden layers, the network is enabled to extract higher-order statistics. The ability of hidden neurons to extract higher-order statistics is particularly valuable when the size if the input layer is large.

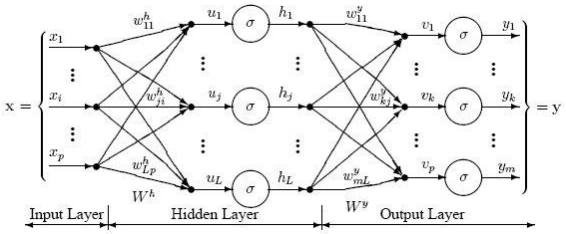

Figure 1. Example of a multilayer feedforward network

Backpropagation is a family of methods for training a multilayer perceptron. Perceptrons are neural nets that use an error-correction rule to change the weights of each unit that makes erroneous responses to stimuli that are presented to the network [9-10]. Rumelhart, Hinton and Williams (1986) in the most influential paper on the error backpropagation method, providing a formula for propagating back the gradient of error evaluation from a unit to the units that provide its inputs. Since the formulas involve derivatives, the input and output of each unit must take continues values in some range, here taken to be [0, 1]. The response is a sigmoidal function of the weighted sum [9].

Consider a layered loop-free net with error $E=\Sigma_k(t_k-o_k)^2$, where k ranges over designated output units, and let the weights w_{ij} be changed according to the gradient rule:

$$\Delta w_{ij} = -\frac{\partial E}{\partial w_{ij}} = 2 \sum_k (t_k - o_k) \frac{\partial o_k}{\partial w_{ij}} \qquad (1)$$

A.3 Recurrent Networks

A recurrent neural network distinguishes itself from a feedforward neural network in that it has at least one feedback loop. For example, a recurrent network may consist of a single layer of neurons with each neuron feeding its output signal back to the inputs of all the other neurons.

B. Radial Basis Functions

The radial basis function (RBF) [11] network or the potential function network is conceptually a very simple yet intrinsically powerful network structure. The radial basis function network (RBFN) as an alternative to the multilayered feedfowrard neural networks has been studied intensively. A RBFN is a multidimensional nonlinear function mapping that depends on the distance between the input vector and the center vector. A RBFN with n-dimensional input $x \in \Re^n$ and a single output $y \in \Re$ can be represented, as shown in the next figure, by the weighted summation of a finite number of radial basis functions.

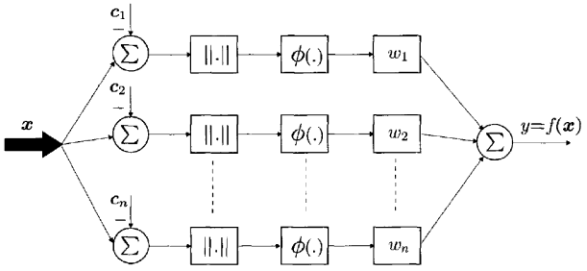

Figure 2. Block diagram representation of the radial basis function network (RBFN) with input $x \in \Re^n$ and output $y \in \Re$.

C. Self-Organizing Map (SOM)

The Self-Organizing Map (SOM) is a type of artificial neural network for the visualization of high-dimensional data. In its basic form it produces a similarity graph of input data. It converts the nonlinear statistical relationships between high-dimensional data into simple geometric relationships of their image points on a low-dimensional display, usually a regular two-dimensional grid of nodes. As the SOM thereby compresses information of the primary data elements on the display, it may also be thought to produce some kind of abstraction. These two aspects, visualization and abstraction, can be utilized in a number of ways in complex tasks such as process analysis, machine perception, control and communication [12].

D. Support Vector Machines

The support vector machine (SVM) [13] is a supervised learning method that generates input-output mapping functions from a set of labeled training data. The mapping function can be either a classification function, i.e., the category of the input data, or a regression function. For classification, nonlinear kernel functions are often used to transform input data to a high-dimensional feature space in which the input data becomes more separable compared to the original input space. Maximum-margin hyper planes are then created. The model thus produced depends on only a subset of the training data near the class boundaries. Similarly, the model produced by Support Vector Regression ignores any training data that is sufficiently close to the model prediction. SVMs are also said to belong to "kernel methods".

In addition to its solid mathematical foundation in statistical learning theory, SVMs have demonstrated highly competitive performance in numerous real-world applications, such as bioinformatics, text mining, face recognition, and image processing, which has established SVMs as one of the state-of the- art tools for machine learning and data mining, along with other soft computing techniques, e.g., neural networks and fuzzy systems.

III. PATTERN RECOGNITION

A pattern is essentially an arrangement or an ordering in which some organization of underlying structure can be said to exist. We can view the world as made up of patterns. Watanabe (1985) defines a pattern as an entity,

vaguely defined, that could be given a name. A pattern can be referred to as a quantitative or structural description of an object or some other item of interest. A set of patterns that share some common properties can be regarded as a pattern class [8].

Pattern recognition as a field of study developed significantly in the 1960s [9]. It is a process of categorizing any sample of measured or observed data as a member of one of the several classes or categories. Due to the fact that pattern recognition is a basic attribute of human beings and other living things, it has been taken for granted for long time [10]. There are different ways to design a chart of how a classifier is designed, depending the details level [9-14]. Figure 3 shows the various stages followed for the design of a classification system.

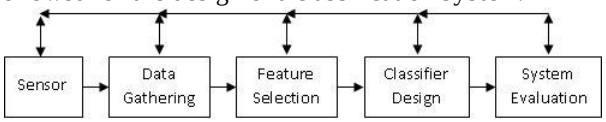

Figure 3. The basic stages involved in the design of a classification system.

The large numbers of applications, ranging from the classical ones such as automatic character recognition and medical diagnosis to the more recent ones in *data mining*, have attracted considerable research effort, with many methods developed and advances made. Other researchers were motivated by the development of machines with "brain-like" performance that in some way could emulate human performance. There were many over-optimistic and unrealistic claims made, and to some extent there exist strong parallels with the growth of research on knowledge-based systems in the 1970s and neural networks in the 1980s.

Nevertheless, within these areas significant progress has been made, particularly where the domain overlaps with probability and statistics, and within recent years there have been many exciting new developments, both in methodology and applications. These build on the solid foundations of earlier research and take advantage of increased computational resources readily available nowadays. These developments include, for example, kernel-based methods and Bayesian computational methods [9].

A. Bayesian

The Bayes [15] likelihood ratio test has been shown to be optimal in the sense that it minimizes the cost or the probability of error. However, in order to construct the likelihood ratio, we must have the conditional probability density function for each class. In most applications, we must estimate these density functions using a finite number of sample observation vectors. However, they may be very complex or require a large number of samples to give accurate results.

Even if we can obtain the densities, the likelihood ratio test may be difficult to implement; time and storage requirements for the classification process may be excessive. Therefore, we are often led to consider a simpler procedure for designing a pattern classifier. In particular, we may specify the mathematical form of the classifier, leaving a finite set of parameters to be determined. The most common choices are linear, quadratic or piecewise classifiers [12]

B. Supervised learning

Supervised learning requires a training set which consists of input vectors and a target vector associated with each input vector. The supervised learning process also requires a trainer that submits both the input and the target patterns for the objects to get recognized. The trainer uses the target vector to determine how well it has learned, and to guide adjustments to weight values to reduce its overall error. Among the supervised learning algorithms, most common are the back-propagation training and Widrow-Hoff's MADALINEs [14].

The advantage of using a supervised learning system to perform pattern classification is that it can construct a linear or a nonlinear decision boundary between different classes in a nonparametric fashion, and therefore offer a practical method for solving highly complex pattern classification problems.

C. Unsupervised learning

The process of unsupervised learning is required in many recognition problems, where the target pattern is unknown. The unsupervised learning process attempts to generate a unique set of weights for one particular class of patterns. The objective of unsupervised learning is to discover patterns or features in the input data with no help from a "teacher", basically performing a clustering of input space.

Among the typical class of unsupervised learning, neural nets, Hopfield nets [16], associative memory, and cognitive neural nets need special mention. One form of unsupervised learning is clustering. Another example is blind source separation based on Independent Component Analysis (ICA). Among neural network models, the Self-organizing map (SOM) [17] and Adaptive resonance theory (ART) are commonly used unsupervised learning algorithms.

IV. EVOLUTIONARY SYSTEMS

One of the strongest challenges facing the current computer is in the difficulty of maintaining updated information systems. It is common when the build of a new system is finished the requirements in the organization have suffered changes (in its structure) that are not reflected in the system. On the other hand, even if the system is updated, it is difficult to include changes or necessary modifications to maintain an image of the organization, since the outside is changing dynamically.

Within these mechanisms have been designed to develop a system based on artificial intelligence tools, able to capture changes in the environment and reconfigured internally to represent these changes, this mechanism may fall into one of the following levels:

- Static: Able to stay in medium changes or with little change (ecological niche).
- Adaptive Static: They are capable of adapting to the environment stable.
- Transformers: Capable of influence to change the means.
- Adaptive Dynamic: Capable of staying in half being transformed.
- Evolutionary: Capable of becoming a medium that is being transformed.

A. *Characteristics of an evolutionary system*

The construction of an evolutionary system has many interesting problems, since it comes finding a mechanism, develop "grow" the user independently and among other observed the following problems:

The system must have a mechanism to grasp the reality that surrounds it, because you need to know and studying the environment in order to detect differences and required changes to adapt and evolve in that medium.

The system must be able to store and display the knowledge, to build its own representation of reality and explore it.

The system must be able to "generate" new knowledge from stored knowledge and capture the outside knowledge, in order to propose changes or amendments to his picture of reality including that new knowledge.

The system must be able to abstract general rules from a set of knowledge that represents them in a synthetic form.

The system must be capable of establishing a dialogue with the outside so that it can transmit its knowledge and through feedback promote change on the outside.

B. *Techniques involved*

Expert systems are capable to attack and solve problems in areas of knowledge specific grounds that are a representation of knowledge driven in that area and use a mechanism to infer new knowledge from stored knowledge.

Other techniques are evolutionary algorithms (comprising genetic algorithms, evolutionary programming, evolution strategy and genetic programming) and swarm intelligence (comprising ant colony optimization and particle swarm optimization)

V. FUZZY LOGIC

This fuzzy logic [18] theory allows us to handle and process certain types of information which are managed in terms of inaccurate, imprecise or subjective. In a similar way as does the human brain, it is possible to order a rule-based reasoning imprecise and incomplete data.

To do this we must expand the set theory and Boolean logic so that an individual may belong partially to a set and logic operations in addition to ones and zeros, can be 0.01 or 0.75. We communicate and coordinate actions with data as "... you are too young to do that ..."; how much is "too"?; What is "young"?.

With fuzzy sets we can define sub-sets; in such a way that anything they may belong to different degrees. Fuzzy rules process the relations between the variables and produce a fuzzy output. From those outputs, binary quantities and continuous quantities can be provided, as the state of a switch or a fee. The rules for the disposal of the inference engine of a fuzzy system can be made by experts or learned by the system itself, in this case is making use of neural networks to strengthen future decision making.

Some programming languages that have embedded fuzzy logic would be for example the different implementations of Fuzzy Prolog or the language Frill. The fuzzy mathematic involve the next operations:

A. *Fuzzyfication*

Fuzzyfication is the translation of real-world values to Fuzzy environment by using membership functions. The membership functions of the Figure 4 reflect a steady **rate = 55** in the fuzzy values (degrees of membership), **SLOW = 0.25**, **MEDIUM = 0.75** and **FAST = 0**.

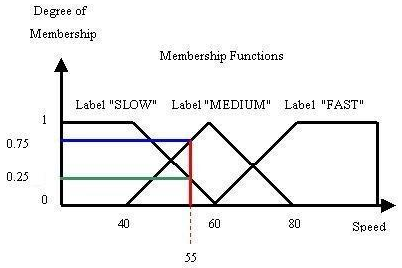

Figure 4. Fuzzyfication process

B. *Rule evaluation*

The Rule evaluation is the determination of the strength of the rules based on input values and rules.

For example:

If SPEED = MEDIUM **and** HIGHER = SECURE

then GAS = INCREASE

Assume in this case, **MEDIUM = 0.75** and **SECURE = 0.5**. Now the validity of the rule will be 0.5 (The minimum value between the background) and then become INCREASE fuzzy variable equal to 0.5.

Thus, we find two rules involving the fuzzy variable INCREASE. A "OR" fuzzy between the results of the two rules will be 0.5 (maximum value between two operands). **INCREASE = 0.5**

C. *Defuzzyfication*

Defuzzyfication is translating the results back diffuse to real world values. After computing the fuzzy rules and evaluate fuzzy variables, the values must moved out into the real world. Then we will require a membership function (membership functions) for each of the output variables, as shown in the next figure.

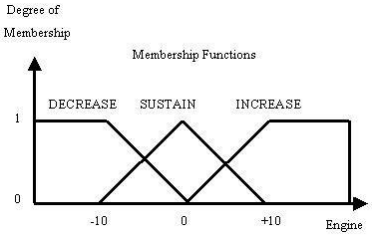

Figure 5. Deffuzyfication process

VI. AMBIENT INTELLIGENCE

Ambient Intelligence [19] is a model of interaction in which people are surrounded by a digital environment aware of our presence, context sensitive, which adaptively responds to our needs and habits, to make life easier daily at home, leisure and work. In addition, Ambient Intelligence is a vision that places the human being as a center for future development in society knowledge and information and communication technologies. These technologies are embedded in everyday objects and nearly invisible to those who use

and their interfaces will be simple and usable in a natural way. For example, a Smart Object in AmI environment could be a cup of coffee to take different colors depending on the temperature of the beverage contained or level of sugar according to one's diet.

But for this vision, somewhat futuristic, which should increase our lives at home, work or during leisure time, a reality is needed in a number of technologies, which engineers will contribute in the coming years:

Hardware discrete miniaturized using nanotechnology, smart devices and sensors that capture the environment.

A communication infrastructure based fixed and mobile web, where networks wireless and wired communication interoperate and converge.

Networks of Dynamic and massively distributed device, without central servers capable of cooperation.

User interfaces more natural and close to humans as voice or gestures.

Sensitivity to the context where our position, identity, or serve as activity Implicit input parameters to enable an intelligent environment know our current situation and act accordingly.

VII. DIMENSIONALITY REDUCTION

Dimensionality reduction can be achieved by two different ways, feature selection and feature extraction [14-17]. All systems that deal with datasets with large dimensionality, feature selection and extraction have found wide applicability. Some of the main areas of application are shown in Figure 6.

Feature selection methods select a subset of the original features based on a subset evaluation function. Basically feature selection methods identify those variables that do not contribute to the classification task. In a discrimination problem, we would neglect those variables that do not contribute to class separability.

Feature extraction methods transform the underline meaning of the data. This transformation may be a linear or nonlinear combination of the original variables and may be supervised or unsupervised. In the supervised case, the task is to find the transformation for which a particular criterion of class separability is maximized.

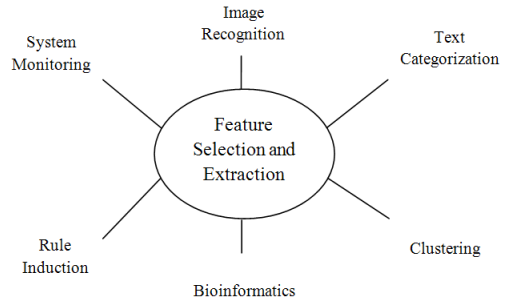

Figure 6. Typical feature selection and extraction application areas

REFERENCES

[1]. C.S. Krishnamoorthy, S. Rajeev, "Artificial Intelligence and Expert Systems for Engineers", CRC Press, CRC Press LLC 1996.

[2]. Jay Liebowitz, "The Handbook of Applied Expert Systems", *CRC Press LLC* 1997.

[3]. Dennis Merritt, "Building Expert Systems in Prolog", Springer; 1st edition (1989)

[4]. Ivan Bratko, "Prolog Programming for Artificial Intelligence", Addison Wesley; 3rd edition (2000)

[5]. Neil C. Rowe, "Artificial Intelligence through Prolog", Prentice-Hall (1988)

[6]. M.M. Huntbach, G.A. Ringwood: Agent-Oriented Programming, Springer (1999).

[7]. Joseph C. Giarratano, Gary D. Riley, "Expert Systems: Principles and Programming", Course Technology; 3rd edition (1998)

[8]. [16] Simon Haykin, "Neural Networks, A Comprehensive Foundation", Second edition, © 1999 by Pearson Education, Inc. ISBN: 81-7808-300-0.

[9]. [17] Michael A. Arbib, "The Handbook of Brain Theory and Neural Networks", Second Edition, ©2003 by Massachusetts Institute of Technology, ISBN: 0-262-01197-2.

[10]. [18] Madan M. Gupta, Liang Jin, and Noriyasu Homma, "Static and Dynamic Neural Networks, From Fundamentals to Advanced Theory", © 2003 by John Wiley & Sons, Inc. ISBN: 0-471-21948-7.

[11]. Martin D. Buhmann, "Radial Basis Functions: Theory and Implementations", Cambridge University Press (2003)

[12]. [19] Teuvo Kohonen, "Self-Organizing Maps", Information Sciences, Third Edition, © 2001 by Springer-Verlag Berlin Heidelberg New York. ISBN: 3-540-67921-9.

[13]. [20] Lipo Wang (Ed.), "Support Vector Machines: Theory and Applications", © 2005 by Springer-Verlag Berlin Heidelberg, ISBN: 3-540-24388-7.

[14]. [21] Richard Jensen, Qiang Shen, "Computational Intelligence and Feature Selection, Rough and Fuzzy Approaches", Aberystwyth University, © 2008 by Institute of Electrical and Electronics Engineers. Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

[15]. Richard E. Neapolitan, "Learning Bayesian Networks", Prentice Hall (2003)

[16]. [22] Andrew R. Webb, QinetiQ Ltd., Malvern, UK, "Statistical Pattern Recognition", Second Edition, © 2002 by John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester.

[17]. [23] Juan R. Rabunal, Julian Dorado, "Artificial Neural Networks in Real-Life Applications", University of a Caruna, Spain. © 2006 by Idea Group Inc.

[18]. George J. Klir, Bo Yuan, "Fuzzy Sets and Fuzzy Logic: Theory and Applications", Prentice Hall; 1st edition (1995)

[19]. G. Riva, F. Vatalaro, F. Davide, M. Alcaniz, "The evolution of technology, communication and cognition towards the future of human-computer interaction", IOS Press (2005)