Chapter 0

Combining Evolutionary and Fuzzy Techniques in Medical Diagnosis

Carlos Andrés Peña-Reyes and **Moshe Sipper**
Logic Systems Laboratory
Swiss Federal Institute of Technology in Lausanne
Switzerland
Carlos.Pena@epfl.ch, Moshe.Sipper@epfl.ch

In this chapter we focus on the Wisconsin breast cancer diagnosis (WBCD) problem, combining two methodologies—fuzzy systems and evolutionary algorithms—to automatically produce diagnostic systems. We present two hybrid approaches: (1) a fuzzy-genetic algorithm, and (2) Fuzzy CoCo, a novel cooperative coevolutionary approach to fuzzy modeling. Both methods produce systems exhibiting high classification performance, and which are also human-interpretable. Fuzzy CoCo obtains higher-performance systems than the standard fuzzy-genetic approach while expending less computational effort.

1 Introduction

A major class of problems in medical science involves the diagnosis of disease, based upon various tests performed upon the patient. When several tests are involved, the ultimate diagnosis may be difficult to obtain, even for a medical expert. This has given rise, over the past few decades, to computerized diagnostic tools, intended to aid the physician in making sense out of the welter of data.

A prime target for such computerized tools is in the domain of cancer diagnosis. Specifically, where breast cancer is concerned, the treating physician is interested in ascertaining whether the patient under examination exhibits the symptoms of a benign case, or whether her case is a malignant one.

A good computerized diagnostic tool should possess two characteristics, which are often in conflict. First, the tool must attain the highest possible *performance*, i.e., diagnose the presented cases correctly as being either *benign* or *malignant*. Second, it would be highly beneficial for such a diagnostic system to be human-friendly, exhibiting so-called *interpretability*. This means that the physician is not faced with a black box that simply spouts answers (albeit correct) with no explanation; rather, we would like for the system to provide some insight as to *how* it derives its outputs.

In this chapter we present the combination of two methodologies—fuzzy systems and evolutionary algorithms—to automatically produce systems for breast cancer diagnosis. The major advantage of fuzzy systems is that they favor interpretability, however, finding good fuzzy systems can be quite an arduous task. This is where evolutionary algorithms step in, enabling the automatic production of fuzzy systems, based on a database of training cases. There are several recent examples of the application of fuzzy systems and evolutionary algorithms in the medical domain [28][1], though only a few combine both methodologies in a hybrid way—as we do in this chapter.

This chapter is organized as follows: In the next section we provide an overview of fuzzy modeling, evolutionary computation, and evolutionary fuzzy modeling. In Section 3 we describe the Wisconsin breast cancer diagnosis (WBCD) problem, which is the focus of our interest herein. Section 4 then describes a fuzzy-genetic approach to the WBCD problem. Section 5 presents Fuzzy CoCo, our cooperative coevolutionary approach to fuzzy modeling, and its application to the WBCD problem. Finally, we present concluding remarks in Section 6.

[1]This article provides over one hundred references to works in the medical domain using evolutionary computation.

2 Background

2.1 Fuzzy modeling

Fuzzy logic is a computational paradigm that provides a mathematical tool for representing and manipulating information in a way that resembles human communication and reasoning processes [43]. It is based on the assumption that, in contrast to Boolean logic, a statement can be *partially* true (or false), and composed of imprecise concepts. For example, the expression "I live near Geneva," where the fuzzy value "near" applied to the fuzzy variable "distance," in addition to being imprecise, is subject to interpretation. A *fuzzy variable* (also called a *linguistic variable*; see Figure 1) is characterized by its name tag, a set of *fuzzy values* (also known as *linguistic values* or *labels*), and the membership functions of these labels; these latter assign a membership value $\mu_{label}(u)$ to a given real value $u \in \Re$, within some predefined range (known as the universe of discourse). While the traditional definitions of Boolean-logic operations do not hold, new ones can be defined. Three basic operations, **and**, **or**, and **not**, are defined in fuzzy logic as follows:

$$\mu_{A\mathbf{and}B}(u) = \mu_A(u) \wedge \mu_B(u) = min\{\mu_A(u), \mu_B(u)\},$$
$$\mu_{A\mathbf{or}B}(u) = \mu_A(u) \vee \mu_B(u) = max\{\mu_A(u), \mu_B(u)\},$$
$$\mu_{\mathbf{not}A}(u) = \neg\mu_A(u) = 1 - \mu_A(u),$$

where A and B are fuzzy variables. Using such fuzzy operators one can combine fuzzy variables to form fuzzy-logic expressions, in a manner akin to Boolean logic. For example, in the domain of control, where fuzzy logic has been applied extensively, one can find expressions such as: **if** room temperature **is** Warm, **then** increase slightly the ventilation-fan speed.

A *fuzzy inference system* is a rule-based system that uses fuzzy logic, rather than Boolean logic, to reason about data [43]. Its basic structure consists of four main components, as depicted in Figure 2: (1) a fuzzifier, which translates crisp (real-valued) inputs into fuzzy values; (2) an inference engine that applies a fuzzy reasoning mechanism to obtain a fuzzy output; (3) a defuzzifier, which translates this latter output into a crisp value; and (4) a knowledge base, which contains both an ensemble of fuzzy rules, known as the rule base, and an ensemble of membership functions known as the database.

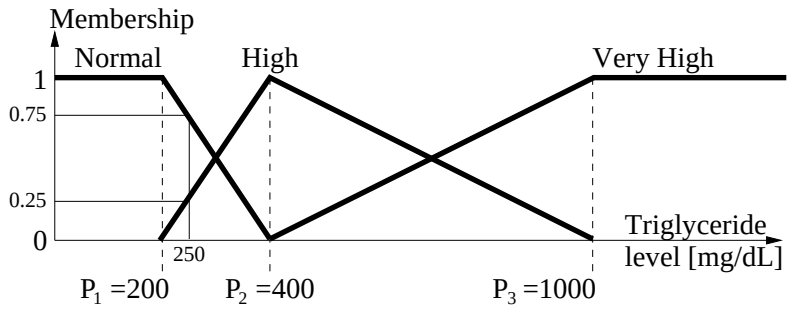

Figure 1. Example of a fuzzy variable: *Triglyceride level* has three possible fuzzy values, labeled **Normal**, **High**, and **Very High**, plotted above as degree of membership versus input value. The values P_i, setting the trapezoid and triangle apices, define the membership functions. In the figure, an example input value 250 mg/dL is assigned the membership values $\mu_{Normal}(250) = 0.75$, $\mu_{High}(250) = 0.25$, and $\mu_{VeryHigh}(250) = 0$. Note that $\mu_{Normal}(250) + \mu_{High}(250) + \mu_{VeryHigh}(250) = 1$.

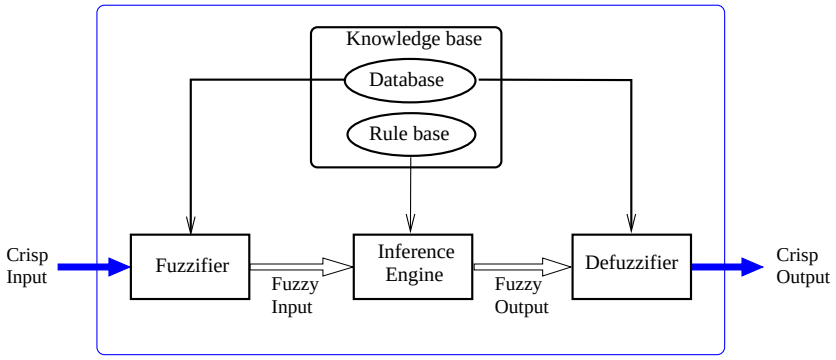

Figure 2. Basic structure of a fuzzy inference system.

The decision-making process is performed by the inference engine using the rules contained in the rule base. These fuzzy rules define the connection between input and output fuzzy variables. A fuzzy rule has the form:

if $antecedent$ **then** $consequent$,

where $antecedent$ is a fuzzy-logic expression composed of one or more simple fuzzy expressions connected by fuzzy operators, and $consequent$ is an expression that assigns fuzzy values to the output variables. The inference engine evaluates all the rules in the rule base and combines the weighted consequents of all relevant rules into a single fuzzy set using

the *aggregation* operation. This operation is the analog in fuzzy logic of the average operator in arithmetic [42] (aggregation is usually performed with the *max* operator).

Fuzzy modeling is the task of identifying the parameters of a fuzzy inference system so that a desired behavior is attained [42]. Note that the fuzzy-modeling process has to deal with an important trade-off between the *accuracy* and the *interpretability* of the model. In other words, the model is expected to provide high numeric precision while incurring as little a loss of linguistic descriptive power as possible. With the *direct* approach a fuzzy model is constructed using knowledge from a human expert. This task becomes difficult when the available knowledge is incomplete or when the problem space is very large, thus motivating the use of *automatic* approaches to fuzzy modeling. There are several approaches to fuzzy modeling, based on neural networks [14, 22, 41], evolutionary algorithms [2, 7, 26], and hybrid methods [35, 37]. Selection of relevant variables and adequate rules is critical for obtaining a good system. One of the major problems in fuzzy modeling is the *curse of dimensionality*, meaning that the computation requirements grow exponentially with the number of variables.

The parameters of fuzzy inference systems can be classified into four categories (Table 1) [26]: logical, structural, connective, and operational. Generally speaking, this order also represents their relative influence on performance, from most influential (logical) to least influential (operational).

In fuzzy modeling, logical parameters are usually predefined by the designer based on experience and on problem characteristics. Typical choices for the reasoning mechanism are Mamdani-type, Takagi-Sugeno-Kang (TKS)-type, and singleton-type [42]. Common fuzzy operators are min, max, product, probabilistic sum, and bounded sum. The most common membership functions are triangular, trapezoidal, and bell-shaped. As for defuzzification, several methods have been proposed, with the Center of Area (COA) and the Mean of Maxima (MOM) being the most popular [19, 42].

Table 1. Parameter classification of fuzzy inference systems.

Class	Parameters
Logical	Reasoning mechanism
	Fuzzy operators
	Membership function types
	Defuzzification method
Structural	Relevant variables
	Number of membership functions
	Number of rules
Connective	Antecedents of rules
	Consequents of rules
	Rule weights
Operational	Membership-function values

Structural, connective, and operational parameters may be either predefined, or obtained by synthesis or search methodologies. Generally, the search space, and thus the computational effort, grows exponentially with the number of parameters. Therefore, one can either invest more resources in the chosen search methodology, or infuse more *a priori*, expert knowledge into the system (thereby effectively reducing the search space). The aforementioned trade-off between accuracy and interpretability is usually expressed as a set of constraints on the parameter values, thus complexifying the search process.

2.2 Evolutionary computation

The domain of evolutionary computation involves the study of the foundations and the applications of computational techniques based on the principles of natural evolution. Evolution in nature is responsible for the "design" of all living beings on earth, and for the strategies they use to interact with each other. Evolutionary algorithms employ this powerful design philosophy to find solutions to hard problems.

Generally speaking, evolutionary techniques can be viewed either as search methods, or as optimization techniques. As written by Michalewicz [21]:

> Any abstract task to be accomplished can be thought of as solving

a problem, which, in turn, can be perceived as a search through a space of potential solutions. Since usually we are after 'the best' solution, we can view this task as an optimization process.

The first works on the use of evolution-inspired approaches to problem solving date back to the late 1950s [4, 5, 8, 10, 11]. Independent and almost simultaneous research conducted by Rechenberg and Schwefel on *evolution strategies* [34, 36], by Holland on *genetic algorithms* [13], and by Fogel on *evolutionary programming* [9] triggered the study and the application of evolutionary techniques.

Three basic mechanisms drive natural evolution: *reproduction*, *mutation*, and *selection*. The first two act on the *chromosomes* containing the genetic information of the *individual* (the *genotype*), rather than on the individual itself (the *phenotype*) while selection acts on the phenotype. Reproduction is the process whereby new individuals are introduced into a *population*. During sexual reproduction, *recombination* (or *crossover*) occurs, transmitting to the offspring chromosomes that are a melange of both parents' genetic information. Mutation introduces small changes into the inherited chromosomes; it often results from copying errors during reproduction. Selection, acting on the phenotype, is a process guided by the Darwinian principle of survival of the fittest. The fittest individuals are those best adapted to their environment, which thus survive and reproduce.

Evolutionary computation makes use of a metaphor of natural evolution, according to which a problem plays the role of an environment wherein lives a population of individuals, each representing a possible solution to the problem. The degree of adaptation of each individual (i.e., candidate solution) to its environment is expressed by an adequacy measure known as the *fitness function*. The phenotype of each individual, i.e., the candidate solution itself, is generally encoded in some manner into its *genome* (genotype). Evolutionary algorithms potentially produce progressively better solutions to the problem. This is possible thanks to the constant introduction of new "genetic" material into the population, by applying so-called genetic operators which are the computational equivalents of natural evolutionary mechanisms.

There are several types of evolutionary algorithms, among which the best known are *genetic algorithms*, *genetic programming*, *evolution strategies*, and *evolutionary programming*; though different in the specifics they are all based on the same general principles. The archetypal evolutionary algorithm proceeds as follows: An initial population of individuals, $P(0)$, is generated at random or heuristically. Every evolutionary step t, known as a *generation*, the individuals in the current population, $P(t)$, are *decoded* and *evaluated* according to some predefined quality criterion, referred to as the fitness, or fitness function. Then, a subset of individuals, $P'(t)$—known as the *mating pool*—is selected to reproduce, with selection of individuals done according to their fitness. Thus, high-fitness ("good") individuals stand a better chance of "reproducing," while low-fitness ones are more likely to disappear.

Selection alone cannot introduce any new individuals into the population, i.e., it cannot find new points in the search space. These points are generated by altering the selected population $P'(t)$ via the application of crossover and mutation, so as to produce a new population, $P''(t)$. Crossover tends to enable the evolutionary process to move toward "promising" regions of the search space. Mutation is introduced to prevent premature convergence to local optima, by randomly sampling new points in the search space. Finally, the new individuals $P''(t)$ are introduced into the next-generation population, $P(t+1)$; usually $P''(t)$ simply becomes $P(t+1)$. The termination condition may be specified as some fixed, maximal number of generations or as the attainment of an acceptable fitness level. Figure 3 presents the structure of a generic evolutionary algorithm in pseudo-code format.

As they combine elements of directed and stochastic search, evolutionary techniques exhibit a number of advantages over other search methods. First, they usually need a smaller amount of knowledge and fewer assumptions about the characteristics of the search space. Second, they can more easily avoid getting stuck in local optima. Finally, they strike a good balance between *exploitation* of the best solutions, and *exploration* of the search space. The strength of evolutionary algorithms relies on their population-based search, and on the use of the genetic mechanisms described above. The existence of a population of candidate solutions entails a parallel search, with the selection mechanism directing the

```
begin EA
    t:=0
    Initialize population P(t)
    while not done do
        Evaluate P(t)
        P'(t) := Select[P(t)]
        P''(t) := ApplyGeneticOperators[P'(t)]
        P(t + 1) := Introduce[P''(t),P(t)]
        t:=t+1
    end while
end EA
```

Figure 3. Pseudo-code of a standard evolutionary algorithm.

search to the most promising regions, the crossover operator encouraging the exchange of information between these search-space regions, and the mutation operator enabling the exploration of new directions.

The application of an evolutionary algorithm involves a number of important considerations. The first decision to take when applying such an algorithm is how to encode candidate solutions within the genome. The representation must allow for the encoding of all possible solutions while being sufficiently simple to be searched in a reasonable amount of time. Next, an appropriate fitness function must be defined for evaluating the individuals. The (usually scalar) fitness value must reflect the criteria to be optimized and their relative importance. Representation and fitness are thus clearly problem-dependent, in contrast to selection, crossover, and mutation, which seem *prima facie* more problem-independent. Practice has shown, however, that while standard genetic operators can be used, one often needs to tailor these to the problem as well.

We noted above that there are several types of evolutionary algorithms. The distinction is mainly due to historical reasons and the different types of evolutionary algorithms are in fact quite similar. One could argue that there is but a single general evolutionary algorithm, or just the opposite– that "there are as many evolutionary algorithms as the researchers working in evolutionary computation" [31]. The frontiers among the widely accepted classes of evolutionary algorithms have become fuzzy over the

years as each technique has attempted to overcome its limitations, by imbibing characteristics of the other techniques. To design an evolutionary algorithm one must define a number of important parameters, which are precisely those that demarcate the different evolutionary-computation classes. Some important parameters are: representation (genome), selection mechanism, crossover, mutation, size of populations P' and P'', variability or fixity of population size, and variability or fixity of genome length.

2.3 Evolutionary Fuzzy Modeling

Evolutionary algorithms are used to search large, and often complex, search spaces. They have proven worthwhile on numerous diverse problems, able to find near-optimal solutions given an adequate performance (fitness) measure. Fuzzy modeling can be considered as an optimization process where part or all of the parameters of a fuzzy system constitute the search space. Works investigating the application of evolutionary techniques in the domain of fuzzy modeling had first appeared about a decade ago [15, 16]. These focused mainly on the tuning of fuzzy inference systems involved in control tasks (e.g., cart-pole balancing, liquid-level system, and spacecraft rendezvous operation). Evolutionary fuzzy modeling has since been applied to an ever-growing number of domains, branching into areas as diverse as chemistry, medicine, telecommunications, biology, and geophysics. For a detailed bibliography on evolutionary fuzzy modeling up to 1996, the reader is referred to [1, 6].

Depending on several criteria—including the available *a priori* knowledge about the system, the size of the parameter set, and the availability and completeness of input/output data—artificial evolution can be applied in different stages of the fuzzy-parameter search. Three of the four categories of fuzzy parameters in Table 1 can be used to define targets for evolutionary fuzzy modeling: structural parameters, connective parameters, and operational parameters [26]. As noted in Section 2.1, logical parameters are usually predefined by the designer based on experience.

Knowledge tuning (operational parameters). The evolutionary algorithm is used to tune the knowledge contained in the fuzzy system by finding membership-function values. An initial fuzzy system is defined

by an expert. Then, the membership-function values are encoded in a genome, and an evolutionary algorithm is used to find systems with high performance. Evolution often overcomes the local-minima problem present in gradient descent-based methods. One of the major shortcomings of knowledge tuning is its dependency on the initial setting of the knowledge base.

Behavior learning (connective parameters). In this approach, one supposes that extant knowledge is sufficient in order to define the membership functions; this determines, in fact, the maximum number of rules [42]. The genetic algorithm is used to find either the rule consequents, or an adequate subset of rules to be included in the rule base.

As the membership functions are fixed and predefined, this approach lacks the flexibility to modify substantially the system behavior. Furthermore, as the number of variables and membership functions increases, the curse of dimensionality becomes more pronounced and the interpretability of the system decreases rapidly.

Structure learning (structural parameters). In many cases, the available information about the system is composed almost exclusively of input/output data, and specific knowledge about the system structure is scant. In such a case, evolution has to deal with the simultaneous design of rules, membership functions, and structural parameters. Some methods use a fixed-length genome encoding a fixed number of fuzzy rules along with the membership-function values. In this case the designer defines structural constraints according to the available knowledge of the problem characteristics. Other methods use variable-length genomes to allow evolution to discover the optimal size of the rule base.

Both behavior and structure learning can be viewed as rule-base learning processes with different levels of complexity. They can thus be assimilated within other methods from machine learning, taking advantage of experience gained in this latter domain. In the evolutionary-algorithm community there are two major approaches for evolving such rule systems: the Michigan approach and the Pittsburgh approach [21]. A more recent method has been proposed specifically for fuzzy modeling: the iterative rule learning approach [12]. These three approaches are briefly

described below.

The Michigan approach. Each individual represents a *single* rule. The fuzzy inference system is represented by the *entire population*. Since several rules participate in the inference process, the rules are in constant competition for the best action to be proposed, and cooperate to form an efficient fuzzy system. The cooperative-competitive nature of this approach renders difficult the decision of which rules are ultimately responsible for good system behavior. It necessitates an effective credit-assignment policy to ascribe fitness values to individual rules.

The Pittsburgh approach. Here, the evolutionary algorithm maintains a population of candidate fuzzy systems, each individual representing an *entire* fuzzy system. Selection and genetic operators are used to produce new generations of fuzzy systems. Since evaluation is applied to the entire system, the credit-assignment problem is eschewed. This approach allows to include additional optimization criteria in the fitness function, thus affording the implementation of multi-objective optimization. The main shortcoming of this approach is its computational cost, since a population of full-fledged fuzzy systems has to be evaluated each generation.

The iterative rule learning approach. As in the Michigan approach, each individual encodes a single rule. An evolutionary algorithm is used to find a single rule, thus providing a partial solution. The evolutionary algorithm is used iteratively for the discovery of new rules, until an appropriate rule base is built. To prevent the process from finding redundant rules (i.e., rules with similar antecedents), a penalization scheme is applied each time a new rule is added. This approach combines the speed of the Michigan approach with the simplicity of fitness evaluation of the Pittsburgh approach. However, as with other incremental rule-base construction methods, it can lead to a non-optimal partitioning of the antecedent space.

As mentioned before, the accuracy-interpretability trade-off faced by fuzzy modelers implies the assumption of constraints acting on the parameter values, mainly on the membership-function shapes. The following semantic criteria represent conditions driving fuzzy modeling toward human-interpretable systems [26, 30]:

- *Distinguishability.* Each linguistic label should have semantic meaning and the fuzzy set should clearly define a range in the universe of discourse. In the example of Figure 1, to describe variable *Triglyceride level* we used three meaningful labels: **Normal**, **High**, and **Very High**. Their membership functions are defined using parameters P_1, P_2, and P_3.

- *Justifiable number of elements.* The number of membership functions of a variable should be compatible with the number of conceptual entities a human being can handle. This number should not exceed the limit of 7 ± 2 distinct terms. The same criterion is applied to the number of variables in the rule antecedent.

- *Coverage.* Any element from the universe of discourse should belong to at least one of the fuzzy sets. That is, its membership value must be different than zero for at least one of the linguistic labels. Referring to Figure 1, we see that any value along the x-axis belongs to at least one fuzzy set; no value lies outside the range of all sets.

- *Normalization.* Since all labels have semantic meaning, then, for each label, at least one element of the universe of discourse should have a membership value equal to one. In Figure 1, we observe that all three sets **Normal**, **High**, and **Very High** have elements with membership value equal to 1.

- *Orthogonality.* For each element of the universe of discourse, the sum of all its membership values should be equal to one (as in the example in Figure 1).

3 Fuzzy Systems for Breast Cancer Diagnosis

In this section we present the medical-diagnosis problem which is the object of our study, and the fuzzy system we propose to solve it with.

3.1 The WBCD problem

Breast cancer is the most common cancer among women, excluding skin cancer. The presence of a breast mass[2] is an alert sign, but it does not always indicate a malignant cancer. Fine needle aspiration (FNA)[3] of breast masses is a cost-effective, non-traumatic, and mostly non-invasive diagnostic test that obtains information needed to evaluate malignancy.

The Wisconsin breast cancer diagnosis (WBCD) database [20] is the result of the efforts made at the University of Wisconsin Hospital for accurately diagnosing breast masses based solely on an FNA test [17]. Nine visually assessed characteristics of an FNA sample considered relevant for diagnosis were identified, and assigned an integer value between 1 and 10. The measured variables are as follows:

1. Clump Thickness (v_1);
2. Uniformity of Cell Size (v_2);
3. Uniformity of Cell Shape (v_3);
4. Marginal Adhesion (v_4);
5. Single Epithelial Cell Size (v_5);
6. Bare Nuclei (v_6);
7. Bland Chromatin (v_7);
8. Normal Nucleoli (v_8);
9. Mitosis (v_9).

The diagnostics in the WBCD database were furnished by specialists in the field. The database itself consists of 683 cases, with each entry representing the classification for a certain ensemble of measured values:

[2] Most breast cancers are detected as a lump or mass on the breast, by self-examination, by mammography, or by both [18].
[3] Fine needle aspiration is an outpatient procedure that involves using a small-gauge needle to extract fluid directly from a breast mass [18].

$case$	v_1	v_2	v_3	$\cdots$	v_9	$diagnostic$
1	5	1	1	$\cdots$	1	$benign$
2	5	4	4	$\cdots$	1	$benign$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
683	4	8	8	$\cdots$	1	$malignant$

Note that the diagnostics do not provide any information about the degree of benignity or malignancy.

There are several studies based on this database. Bennet and Mangasarian [3] used linear programming techniques, obtaining a 99.6% classification rate on 487 cases (the reduced database available at the time). However, their solution exhibits little understandability, i.e., diagnostic decisions are essentially black boxes, with no explanation as to how they were attained. With increased interpretability in mind as a prior objective, a number of researchers have applied the method of extracting Boolean rules from neural networks [38, 39]. Their results are encouraging, exhibiting both good performance and a reduced number of rules and relevant input variables. Nevertheless, these systems use Boolean rules and are not capable of furnishing the user with a measure of confidence for the decision made. Our own work on the evolution of fuzzy rules for the WBCD problem has shown that it is possible to obtain diagnostic systems exhibiting high performance, coupled with interpretability and a confidence measure [24–27].

3.2 Fuzzy-system setup

The solution scheme we propose for the WBCD problem is depicted in Figure 4. It consists of a fuzzy system and a threshold unit. The fuzzy system computes a continuous appraisal value of the malignancy of a case, based on the input values. The threshold unit then outputs a *benign* or *malignant* diagnostic according to the fuzzy system's output.

Our previous knowledge about the WBCD problem represents valuable information to be used for our choice of fuzzy parameters (Table 1). When defining our setup we took into consideration the following three results concerning the composition of potential high-performance systems: (1) small number of rules; (2) small number of variables; and (3) monotonicity of the input variables [26]. Some fuzzy models forgo in-

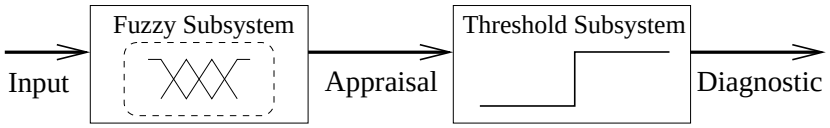

Figure 4. Proposed diagnosis system. Note that the fuzzy subsystem displayed to the left is in fact the entire fuzzy inference system of Figure 2.

terpretability in the interest of improved performance. Where medical diagnosis is concerned, interpretability is the major advantage of fuzzy systems. This motivated us to take into account the five semantic criteria presented in Section 2.3, defining constraints on the fuzzy parameters: (1) distinguishability, (2) justifiable number of elements, (3) coverage, (4) normalization, and (5) orthogonality.

Referring to Table 1, and taking into account these five criteria, we delineate below the fuzzy-system setup:

- Logical parameters: singleton-type fuzzy systems; min-max fuzzy operators; orthogonal, trapezoidal input membership functions (see Figure 5); weighted-average defuzzification.
- Structural parameters: two input membership functions (*Low* and *High*; see Figure 5); two output singletons (*benign* and *malignant*); a user-configurable number of rules. The relevant variables are one of the evolutionary objectives.
- Connective parameters: the antecedents and the consequent of the rules are searched by the evolutionary algorithm. The algorithm also searches for the consequent of the default rule which plays the role of an `else` condition (note that for the fuzzy-genetic approach presented in Section 4, the consequents are predefined instead of evolved, thus reducing the search space). All rules have unitary weight.
- Operational parameters: the input membership-function values are to be found by the evolutionary algorithm. For the output singletons we used the values 2 and 4, for *benign* and *malignant*, respectively.

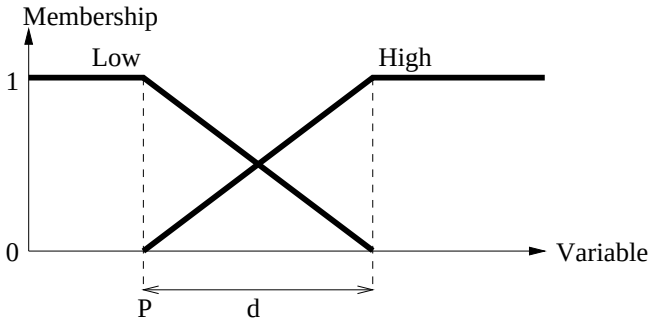

Figure 5. Input fuzzy variables for the WBCD problem. Each fuzzy variable has two possible fuzzy values labeled **Low** and **High**, and orthogonal membership functions, plotted above as degree of membership versus input value. *P* and *d* define the start point and the length of membership function edges, respectively. The orthogonality condition means that the sum of all membership functions at any point is one. In the figure, an example value u is assigned the membership values $\mu_{Low}(u) = 0.8$ and $\mu_{High}(u) = 0.2$ (as can be seen $\mu_{Low}(u) + \mu_{High}(u) = 1$).

4 A Fuzzy-Genetic Approach

The problem, at this stage, consists of searching for three fuzzy-system parameters: input membership functions, antecedents of rules, and relevant variables (consequents of rules are predefined; see Section 3.2). We applied a Pittsburgh-like approach, using a simple genetic algorithm [40] to search for individuals whose genomes encode these three parameters. The next subsection describes the setup of the genetic algorithm, after which subsection 4.2 presents the results obtained applying this approach.

4.1 The evolutionary setup

The genome encodes three sets of parameters: input membership functions, antecedents of rules, and relevant variables. It is defined as follows:

- Membership-function parameters. There are nine variables ($v_1 - v_9$), each with two parameters *P* and *d*, defining the start point and the length of the membership-function edges, respectively (Figure 5).
- Antecedents. The i-th rule has the form:

Table 2. Parameter encoding of an individual's genome. Total genome length is 54 + 18N_r, where N_r denotes the number of rules (N_r is set *a priori* to a value between 1–5, and is fixed during the genetic-algorithm run).

Parameter	Values	Bits	Qty	Total bits
P	{1,2,...,8}	3	9	27
d	{1,2,...,8}	3	9	27
A	{0,1,2,3}	2	$9 \times N_r$	$18 \times N_r$

if (v_1 **is** A_1^i) **and ... and** (v_9 **is** A_9^i) **then** ($output$ **is** *benign*),

where A_j^i represents the membership function applicable to variable v_j. A_j^i can take on the values: 1 (*Low*), 2 (*High*), or 0 or 3 (*Other*).

- Relevant variables are searched for implicitly by letting the algorithm choose non-existent membership functions as valid antecedents; in such a case the respective variable is considered irrelevant. For example, the rule

  if (v_1 **is** *High*) **and** (v_2 **is** *Other*) **and** (v_3 **is** *Other*) **and** (v_4 **is** *Low*) **and** (v_5 **is** *Other*) **and** (v_6 **is** *Other*) **and** (v_7 **is** *Other*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Other*) **then** ($output$ **is** *benign*),

  is interpreted as:

  if (v_1 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_8 **is** *Low*) **then** ($output$ **is** *benign*).

Table 2 delineates the parameter encoding, which together form a single individual's genome.

To evolve the fuzzy inference system, we used a genetic algorithm with a fixed population size of 200 individuals, and fitness-proportionate selection (Subsection 2.2). The algorithm terminates when the maximum number of generations, G_{max}, is reached (we set $G_{max} = 2000 + 500 \times N_r$, i.e., dependent on the number of rules used in the run), or when the increase in fitness of the best individual over five successive generations falls below a certain threshold (in our experiments we used threshold values between 2×10^{-7} and 4×10^{-6}).

Our fitness function combines three criteria: (1) F_c: classification performance, computed as the percentage of cases correctly classified; (2) F_e: the quadratic difference between the continuous appraisal value (in the range $[2, 4]$) and the correct discrete diagnosis given by the WBCD database (either 2 or 4); and (3) F_v: the average number of variables per active rule. The fitness function is given by $F = F_c - \alpha F_v - \beta F_e$, where $\alpha = 0.05$ and $\beta = 0.01$ (these latter values were derived empirically). F_c, the ratio of correctly diagnosed cases, is the most important measure of performance. F_v measures the linguistic integrity (interpretability), penalizing systems with a large number of variables per rule (on average). F_e adds selection pressure towards systems with low quadratic error.

4.2 Results

This section describes the results obtained when applying the methodology described in Section 4.1. We first delineate the success statistics relating to the evolutionary algorithm. Then, we describe in full a three-rule evolved fuzzy system that exemplifies our approach.

A total of 120 evolutionary runs were performed, all of which found systems whose classification performance exceeds 94.5%. In particular, considering the best individual per run (i.e., the evolved system with the highest classification success rate), 78 runs led to a fuzzy system whose performance exceeds 96.5%, and of these, 8 runs found systems whose performance exceeds 97.5%; these results are summarized in Figure 6.

Table 3 shows the results of the best systems obtained with the fuzzy-genetic approach. The number of rules per system was fixed at the outset to be between one and five, i.e., evolution seeks a system with an *a priori* given number of rules. A comparison of these systems with other approaches is presented in Section 5.4 (see also [26]).

We next describe our top-performance system, which serves to exemplify the solutions found by our evolutionary approach. The system, delineated in Figure 7, consists of three rules (note that the `else` condition is not counted as an active rule). Taking into account all three criteria of performance—classification rate, number of rules per system, and average number of variables per rule— this system can be considered the top

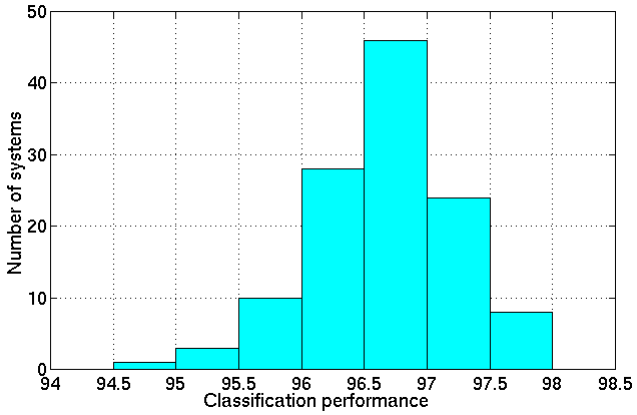

Figure 6. Summary of results of 120 evolutionary runs. The histogram depicts the number of systems exhibiting a given performance level at the end of the evolutionary run. The performance considered is that of the best individual of the run, measured as the overall percentage of correctly classified cases over the entire database.

Table 3. Results of the best systems evolved by the fuzzy-genetic approach. Shown below are the classification performance values of the top systems obtained by these approaches, along with the average number of variables-per-rule. Results are divided into five classes, in accordance with the number of rules-per-system, going from one-rule systems to five-rule ones.

Rules-per-system	Performance	variables-per-rule
1	97.07%	4
2	97.36%	3
3	97.80%	4.7
4	97.80%	4.8
5	97.51%	3.4

one over all 120 evolutionary runs. It obtains an overall classification rate (i.e., over the entire database) of 97.8%.

A thorough test of this three-rule system revealed that the second rule (Figure 7) is never actually used; in the fuzzy literature this is known as a rule that never *fires*, i.e., is triggered by none of the input cases. Thus, it can be eliminated altogether from the rule base, resulting in a two-rule system (also reducing the average number of variables-per-rule from 4.7 to 4).

Database

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
P	3	5	2	2	8	1	4	5	4
d	5	2	1	2	4	7	3	5	2

Rule base

Rule 1 **if** (v_3 **is** *Low*) **and** (v_7 **is** *Low*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** ($output$ **is** *benign*)

Rule 2 **if** (v_1 **is** *Low*) **and** (v_2 **is** *Low*) **and** (v_3 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_5 **is** *High*) **and** (v_9 **is** *Low*) **then** ($output$ **is** *benign*)

Rule 3 **if** (v_1 **is** *Low*) **and** (v_4 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **then** ($output$ **is** *benign*)

Default **else** ($output$ **is** *malignant*)

Figure 7. The best evolved, fuzzy diagnostic system with three rules. It exhibits an overall classification rate of 97.8%, and an average of 4.7 variables per rule. Thorough testing revealed that Rule 2 can be dropped.

5 A Fuzzy Coevolutionary Approach: Fuzzy CoCo

The fuzzy-genetic approach, even though it obtained good diagnostic systems, plateaued at a certain performance level. In this section we present Fuzzy CoCo, a cooperative coevolutionary approach to fuzzy modeling, capable of obtaining higher-performance systems while requiring less computation than the fuzzy-genetic approach. The next subsection briefly explains cooperative coevolution; after which Section 5.2 presents Fuzzy CoCo; Section 5.3 then describes the setup of Fuzzy CoCo when applied to the WBCD probem, and, finally, Section 5.4 presents the results obtained.

5.1 Cooperative coevolution

Coevolution refers to the simultaneous evolution of two or more species with coupled fitness. Such coupled evolution favors the discovery of complex solutions whenever complex solutions are required [23]. Simplistically speaking, one can say that coevolving species can either compete (e.g., to obtain exclusivity on a limited resource) or cooperate (e.g.,

to gain access to some hard-to-attain resource). Cooperative (also called symbiotic) coevolutionary algorithms involve a number of independently evolving species which together form complex structures, well-suited to solve a problem. The fitness of an individual depends on its ability to collaborate with individuals from other species. In this way, the evolutionary pressure stemming from the difficulty of the problem favors the development of cooperative strategies and individuals. Single-population evolutionary algorithms often perform poorly—manifesting stagnation, convergence to local optima, and computational costliness—when confronted with problems presenting one or more of the following features: (1) the sought-after solution is complex, (2) the problem or its solution is clearly decomposable, (3) the genome encodes different types of values, (4) strong interdependencies among the components of the solution, (5) component-ordering drastically affects fitness. Cooperative coevolution effectively addresses these issues, consequently widening the range of applications of evolutionary computation. Potter [32, 33] developed a model in which a number of populations explore different decompositions of the problem. Below we detail this framework as it forms the basis of our own approach.

In Potter's system, each species represents a subcomponent of a potential solution. Complete solutions are obtained by assembling *representative* members of each of the species (populations). The fitness of each individual depends on the quality of (some of) the complete solutions it participated in, thus measuring how well it cooperates to solve the problem. The evolution of each species is controlled by a separate, independent evolutionary algorithm. Figure 8 shows the general architecture of Potter's cooperative coevolutionary framework, and the way each evolutionary algorithm computes the fitness of its individuals by combining them with selected representatives from the other species. A greedy strategy for the choice of representatives of a species is to use one or more of the fittest individuals from the last generation.

5.2 The coevolutionary algorithm

Fuzzy CoCo is a cooperative coevolutionary approach to fuzzy modeling wherein two coevolving species are defined: database (membership functions) and rule base [27]. This approach is based primarily on the

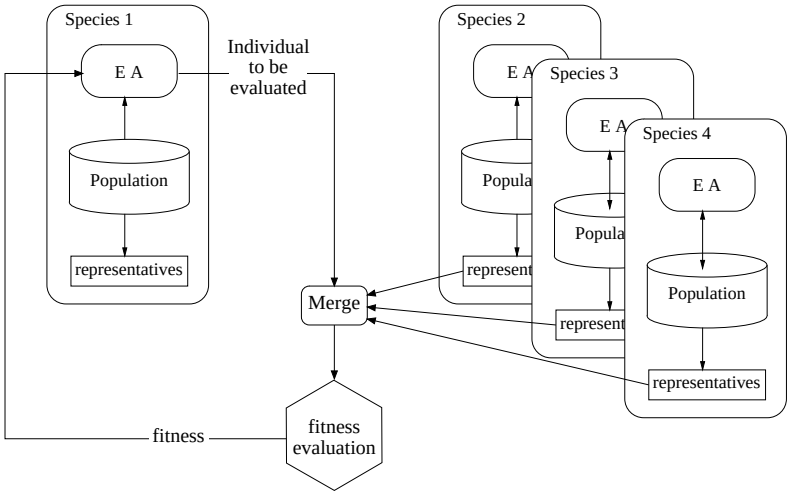

Figure 8. Potter's cooperative coevolutionary system. The figure shows the evolutionary process from the perspective of Species 1. The individual being evaluated is combined with one or more *representatives* of the other species so as to construct several solutions which are tested on the problem. The individual's fitness depends on the quality of these solutions.

framework defined by Potter [32, 33].

A fuzzy modeling process has usually to deal with the simultaneous search for operational and connective parameters (Table 1). These parameters provide an almost complete definition of the linguistic knowledge describing the behavior of a system, and the values mapping this symbolic description into a real-valued world (a complete definition also requires logical and structural parameters whose definition is best suited for human skills). Thus, fuzzy modeling can be thought of as two separate but intertwined search processes: (1) the search for the membership functions (i.e., operational parameters) that define the fuzzy variables, and (2) the search for the rules (i.e., connective parameters) used to perform the inference.

Fuzzy modeling presents several features discussed earlier which justify the application of a cooperative-coevolutionary approach: (1) The required solutions can be very complex, since fuzzy systems with a few dozen variables may call for hundreds of parameters to be defined. (2) The proposed solution—a fuzzy inference system—can be decomposed

into two distinct components: rules and membership functions. (3) Membership functions are continuous and real-valued, while rules are discrete and symbolic. (4) These two components are interdependent because the membership functions defined by the first group of values are indexed by the second group (rules).

Consequently, in Fuzzy CoCo, the fuzzy modeling problem is solved by two coevolving, cooperating species. Individuals of the first species encode values which define completely all the membership functions for all the variables of the system. For example, with respect to the variable $Triglyceridelevel$ shown in Figure 1, this problem is equivalent to finding the values of P_1, P_2, and P_3.

Individuals of the second species define a set of rules of the form:

if (v_1 **is** A_1) **and ... and** (v_n **is** A_n) **then** ($output$ **is** C),

where the term A_v indicates which of the linguistic labels of fuzzy variable v is used by the rule. For example, a valid rule could contain the expression:

if ... and ($Triglyceridelevel$ **is** $High$) **and ... then ...**

which includes the membership function $High$ whose defining parameters are contained in the first species (population).

The two evolutionary algorithms used to control the evolution of the two populations are instances of a simple genetic algorithm [40]. Figure 9 presents the Fuzzy CoCo algorithm in pseudo-code format. The genetic algorithms apply fitness-proportionate selection to choose the mating pool, and apply an elitist strategy with an elitism rate Er to allow some of the best individuals to survive into the next generation. Standard crossover and mutation operators are applied with probabilities P_c and P_m, respectively.

We introduced elitism to avoid the divergent behavior of Fuzzy CoCo, observed in preliminary trial runs. Non-elitist versions of Fuzzy CoCo

begin Fuzzy CoCo
g:=0
for each species S
Initialize populations $P_S(0)$
Evaluate population $P_S(0)$
end for
while not done **do**
for each species S
g:=g+1
$E_S(g) =$ elite-select $P_S(g-1)$
$P'_S(g) =$ select $P_S(g-1)$
$P''_S(g) =$ crossover $P'_S(g)$
$P'''_S(g) =$ mutate $P''_S(g)$
$P_S(g) = P'''_s(g) + E_S(g)$
Evaluate population $P_S(g)$
end for
end while
end Fuzzy CoCo

Figure 9. Pseudo-code of Fuzzy CoCo. Two species coevolve in Fuzzy CoCo: membership functions and rules. The elitism strategy extracts E_S individuals to be reinserted into the population after evolutionary operators have been applied (i.e., selection, crossover, and mutation). Selection results in a reduced population $P'_S(g)$ (usually, the size of $P'_S(g)$ is $\|P'_S\| = \|P_S\| - \|E_S\|$). The line "Evaluate population $P_S(g)$" is elaborated in Figure 10.

tended to lose the genetic information of good individuals found during evolution, consequently producing populations with mediocre individuals scattered throughout the search space. This is probably due to the relatively small size of the population which renders difficult the preservation of good solutions while exploring the search space. The introduction of simple elitism produces an undesirable effect on Fuzzy CoCo's performance: populations converge prematurely even with reduced values of the elitism rate E_r. To offset this effect without losing the advantages of elitism, it was necessary to increase the mutation probability P_m by an order of magnitude so as to improve the exploration capabilities of the algorithm. (Increased mutation rates were also reported by Potter [32, 33] in his coevolutionary experiments.)

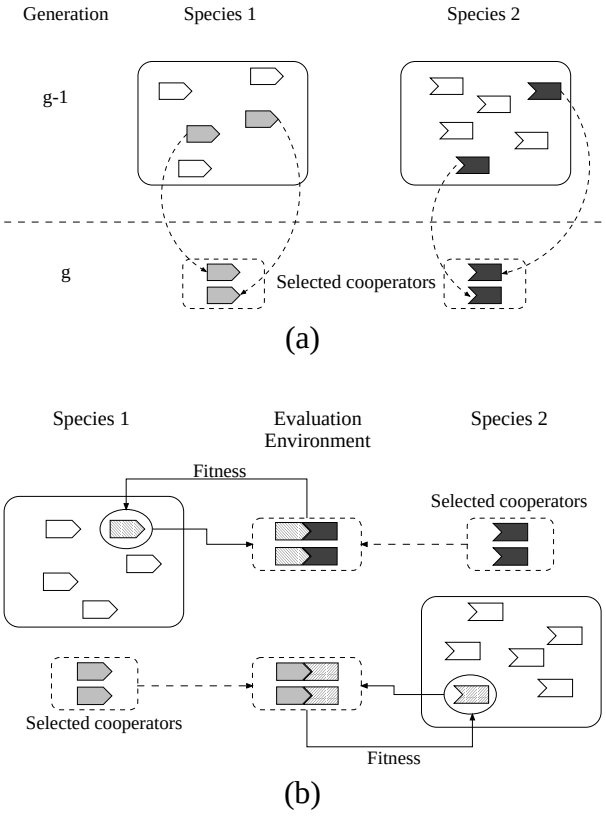

Figure 10. Fitness evaluation in Fuzzy CoCo. (a) Several individuals from generation $g-1$ of each species are selected according to their fitness to be the representatives of their species during generation g; these representatives are called "cooperators." (b) During the evaluation stage of generation g (after selection, crossover, and mutation—see Figure 9), individuals are combined with the selected cooperators of the other species to construct fuzzy systems. These systems are then evaluated on the problem domain and serve as a basis for assigning the final fitness to the individual being evaluated.

A more detailed view of the fitness evaluation process is depicted in Figure 10. An individual undergoing fitness evaluation establishes cooperations with one or more representatives of the other species, i.e., it is combined with individuals from the other species to construct fuzzy systems. The fitness value assigned to the individual depends on the performance of the fuzzy systems it participated in (specifically, either the average or the maximal value).

Representatives, called here *cooperators*, are selected both fitness-

proportionally and randomly from the previous generation since they have already been assigned a fitness value (see Figure 9). In Fuzzy CoCo, N_{cf} cooperators are selected according to their fitness, usually the fittest individuals, thus favoring the exploitation of known good solutions. The other N_{cr} cooperators are selected randomly from the population to represent the diversity of the species, maintaining in this way exploration of the search space.

5.3 The evolutionary setup

Fuzzy CoCo was set to search for four parameters: input membership-function values, relevant input variables, and antecedents and consequents of rules. These search goals are more ambitious than those defined for the fuzzy-genetic approach (Section 4) as the consequents of rules are added to the search space. The genomes of the two species are constructed as follows:

- Species 1: Membership functions. There are nine variables ($v_1 - v_9$), each with two parameters, *P* and *d*, defining the start point and the length of the membership-function edges, respectively (Figure 5).
- Species 2: Rules. The i-th rule has the form:

  if (v_1 **is** A_1^i) **and ... and** (v_9 **is** A_9^i) **then** ($output$ **is** C^i),

  A_j^i can take on the values: 1 (*Low*), 2 (*High*), or 0 or 3 (*Other*). C^i bit can take on the values: 0 (*Benign*) or 1 (*Malignant*). Relevant variables are searched for implicitly by letting the algorithm choose non-existent membership functions (0 or 3) as valid antecedents; in such a case the respective variable is considered irrelevant.

Table 4 delineates the parameter encoding for both species' genomes, which together describe an entire fuzzy system. Note that in the fuzzy-genetic approach (Section 4) both membership functions and rules were encoded in the same genome, i.e., there was only one species.

To evolve the fuzzy inference system, we applied Fuzzy CoCo with the same evolutionary parameters for both species. Values and ranges of values used for these parameters were defined according to preliminary tests

Table 4. Genome encoding of parameters for both species. Genome length for membership functions is 54 bits. Genome length for rules is $19 \times N_r + 1$, where N_r denotes the number of rules.

Species 1: Membership functions

Parameter	Values	Bits	Qty	Total bits
P	{1,2,...,8}	3	9	27
d	{1,2,...,8}	3	9	27
	Total			54

Species 2: Rules

Parameter	Values	Bits	Qty	Total bits
A	{0,1,2,3}	2	$9 \times N_r$	$18 \times N_r$
C	{1,2}	1	$N_r + 1$	$N_r + 1$
	Total			$19 \times N_r + 1$

Table 5. Fuzzy CoCo set-up for the WBCD problem.

Parameter	Values
Population size $\|Ps\|$	[30-90]
Maximum generations G_{max}	$1000 + 100N_r$
Crossover probability P_c	1
Mutation probability P_m	[0.02-0.3]
Elitism rate E_r	[0.1-0.6]
"Fit" cooperators N_{cf}	1
Random cooperators N_{cr}	{1,2,3,4}

performed on benchmark problems (mostly function-optimization problems found in Potter [32]); Table 5 delineates these values. The algorithm terminates when the maximum number of generations, G_{max}, is reached (we set $G_{max} = 1000 + 100 \times N_r$, i.e., dependent on the number of rules used in the run), or when the increase in fitness of the best individual over five successive generations falls below a certain threshold (10^{-4} in our experiments).

Our fitness function combines two criteria: 1) F_c—classification performance, computed as the percentage of cases correctly classified, and 2) F_v—the maximum number of variables in the longest rule. The fitness function is given by $F = F_c - \alpha F_v$, where $\alpha = 0.0015$. F_c, the percentage of correctly diagnosed cases, is the most important measure of performance. F_v measures the linguistic integrity (interpretability), pe-

nalizing systems with a large number of variables in their rules. The value α was calculated to allow F_v to occasion a fitness difference only among systems exhibiting similar classification performance. (We did not apply F_e as it proved of little use.)

We stated earlier that cooperative coevolution reduces the computational cost of the search process. In order to measure this cost we calculated the number of fuzzy-system evaluations performed by a single run of Fuzzy CoCo. Each generation, the $\|Ps\|$ individuals of each population are evaluated N_c times (where $N_c = N_{cf} + N_{cr}$). The total number of fuzzy-system evaluations per run is thus $2 \times G_{max} \times \|Ps\| \times N_c$. This value ranged from 5.28×10^5 evaluations for a one-rule system search, up to 8.16×10^5 evaluations for a seven-rule system (using typical parameter values: $\|Ps\| = 80$, $N_{cf} = 1$, and $N_{cr} = 2$). The number of fuzzy-system evaluations required by our single-population approach was, on the average, 5×10^5 for a one-rule system and 11×10^5 for a seven-rule system [26].

5.4 Results

A total of 495 evolutionary runs were performed, all of which found systems whose classification performance exceeds 96.7%. In particular, considering the best individual per run (i.e., the evolved system with the highest classification success rate), 241 runs led to a fuzzy system whose performance exceeds 98.0%, and of these, 81 runs found systems whose performance exceeds 98.5%.

Table 6 compares our best systems with the top systems obtained by the fuzzy-genetic approach (Section 4) [26] and with the systems obtained by Setiono's NeuroRule approach [38]. The evolved fuzzy systems described in this paper can be seen to surpass those obtained by other approaches in terms of performance, while still containing simple, interpretable rules. As shown in Table 6, we obtained higher-performance systems for all rule-base sizes but one, i.e., from two-rule systems to seven-rule ones, while all our one-rule systems perform as well as the best system reported by Setiono.

We next describe two of our top-performance systems, which serve to ex-

Table 6. Comparison of the best systems evolved by Fuzzy CoCo with the top systems obtained using single-population evolution [26] and with those obtained by Setiono's NeuroRule approach [38]. Shown below are the classification performance values of the top systems obtained by these approaches, along with the number of variables of the longest rule in parentheses. Results are divided into seven classes, in accordance with the number of rules per system, going from one-rule systems to seven-rule ones.

Rules per system	Neuro-Rule [38]	Single population GA [26]	Fuzzy CoCo	
	best	best	average	best
1	97.36% (4)	97.07% (4)	97.36% (4)	97.36% (4)
2	–	97.36% (4)	97.73% (3.9)	98.54% (5)
3	98.10% (4)	97.80% (6)	97.91% (4.4)	98.54% (4)
4	–	97.80% (-)	98.12% (4.2)	98.68% (5)
5	98.24% (5)	97.51% (-)	98.18% (4.6)	98.83% (5)
6	–	–	98.18% (4.3)	98.83% (5)
7	–	–	98.25% (4.7)	98.98% (5)

emplify the solutions found by Fuzzy CoCo. The first system, delineated in Figure 11, presents the highest classification performance evolved to date. It consists of seven rules (note that the `else` condition is not counted as an active rule) with the longest rule including 5 variables. This system obtains an overall classification rate (i.e., over the entire database) of 98.98%.

In addition to the above seven-rule system, evolution found systems with between 2 and 6 rules exhibiting excellent classification performance, i.e., higher than 98.5% (Table 6). Among these systems, we consider as the most interesting the system with the smallest number of conditions (i.e., total number of variables in the rules). Figure 12 presents one such two-rule system, containing a total of 8 conditions, and which obtains an overall classification rate of 98.54%; its longest rule has 5 variables.

The improvement attained by Fuzzy CoCo, while seemingly slight (0.5-1%) is in fact quite significant. A 1% improvement implies 7 additional cases which are classified correctly. At the performance rates in question (above 98%) every additional case is hard-won. Indeed, try as we

Database

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
P	2	1	1	1	6	1	3	5	2
d	7	8	4	8	1	4	8	4	1

Rule base

Rule 1 **if** (v_1 **is** *Low*) **and** (v_3 **is** *Low*) **then** ($output$ **is** *benign*)

Rule 2 **if** (v_4 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** ($output$ **is** *benign*)

Rule 3 **if** (v_1 **is** *Low*) **and** (v_3 **is** *High*) **and** (v_5 **is** *High*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** ($output$ **is** *benign*)

Rule 4 **if** (v_1 **is** *Low*) **and** (v_2 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_5 **is** *Low*) **and** (v_8 **is** *High*) **then** ($output$ **is** *benign*)

Rule 5 **if** (v_2 **is** *High*) **and** (v_4 **is** *High*) **then** ($output$ **is** *malignant*)

Rule 6 **if** (v_1 **is** *High*) **and** (v_3 **is** *High*) **and** (v_6 **is** *High*) **and** (v_7 **is** *High*) **then** ($output$ **is** *malignant*)

Rule 7 **if** (v_2 **is** *High*) **and** (v_3 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_5 **is** *Low*) **and** (v_7 **is** *High*) **then** ($output$ **is** *malignant*)

Default **else** ($output$ **is** *malignant*)

Figure 11. The best evolved, fuzzy diagnostic system with seven rules. It exhibits an overall classification rate of 98.98%, and its longest rule includes 5 variables.

did with the fuzzy-genetic approach—tuning parameters and tweaking the setup—we arrived at a performance impasse. Fuzzy CoCo, however, readily churned out better-performance systems, which were able to classify a significant number of additional cases; moreover, these systems were evolved in less time.

6 Concluding remarks

We presented our recent work which combines the search power of evolutionary algorithms with the expressive power of fuzzy systems to design high-performance, human-interpretable medical diagnostic systems. In particular, we described two approaches for automatically designing systems for breast-cancer diagnosis: (1) a fuzzy-genetic approach and (2) Fuzzy CoCo, our novel cooperative coevolutionary approach to fuzzy modeling.

Database

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
P	3		1	3	4	5		7	2
d	8		3	1	2	2		4	1

Rule base

Rule 1 **if** (v_1 **is** *Low*) **and** (v_3 **is** *Low*) **and** (v_5 **is** *Low*) **then** ($output$ **is** *benign*)

Rule 2 **if** (v_1 **is** *Low*) **and** (v_4 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** ($output$ **is** *benign*)

Default **else** ($output$ **is** *malignant*)

Figure 12. The best evolved, fuzzy diagnostic system with two rules. It exhibits an overall classification rate of 98.54%, and a maximum of 5 variables in the longest rule.

We applied the two aforementioned algorithms to the Wisconsin breast cancer diagnosis problem. Our evolved systems exhibit both characteristics outlined in Section 1: first, they attain high classification *performance* (the best shown to date); second, the resulting systems involve a few simple rules, and are therefore *interpretable*.

We are currently investigating the expansion of Fuzzy CoCo, with two short-term goals in mind: 1) Study the tuning of the genetic-algorithm parameters according to each species characteristics (e.g., encoding schemes, elitism rates, or mutation probabilities). 2) Explore the application of different evolutionary algorithms for each species (e.g., evolution strategies for the evolution of membership functions). In the long term we plan to test some novel ideas that could improve Fuzzy CoCo: 1) Coevolution of $N_r + 1$ species, one species for each of the N_r rules in addition to the membership-function species. 2) Coexistence of several Fuzzy CoCo instances (each one set to evolve systems with a different number of rules), permitting migration of individuals among them so as to increase the exploration and the diversity of the search process. 3) Apply the strategy of rising and death of species proposed by Potter and DeJong [33] in order to evolve systems with variable numbers of rules and membership functions.

References

[1] J. T. Alander. An indexed bibliography of genetic algorithms with fuzzy logic. In Pedrycz [29], pages 299–318.

[2] A. Bastian. Identifying fuzzy models utilizing genetic programming. *Fuzzy Sets and Systems*, 113(3):333–350, August 2000.

[3] K. P. Bennett and O. L. Mangasarian. Neural network training via linear programming. In P. M. Pardalos, editor, *Advances in Optimization and Parallel Computing*, pages 56–57. Elsevier Science, 1992.

[4] G. E. P. Box. Evolutionary operation: A method for increasing industrial productivity. *Applied Statistics*, 6(2):81–101, 1957.

[5] G. E. P. Box and J. S. Hunter. Condensed calculations for evolutionary operation programs. *Technometrics*, 1:77–95, 1959.

[6] O. Cordón, F. Herrera, and M. Lozano. On the combination of fuzzy logic and evolutionary computation: A short review and bibliography. In Pedrycz [29], pages 33–56.

[7] O. Cordón, F. Herrera, and M. Lozano. A two-stage evolutionary process for designing TSK fuzzy rule-based systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 29(6):703–714, December 1999.

[8] D. B. Fogel, editor. *Evolutionary Computation: The Fossil Record*. IEEE Press, Piscataway, NJ, 1998.

[9] L. J. Fogel. Autonomous automata. *Industrial Research*, 4:14–19, 1962.

[10] R. M. Friedberg. A learning machine: I. *IBM Journal of Research and Development*, 2:2–13, 1958.

[11] R. M. Friedberg, B. Dunham, and J. H. North. A learning machine. II. *IBM Journal of Research and Development*, 3:282–287, 1959.

[12] F. Herrera, M. Lozano, and J. L. Verdegay. Generating fuzzy rules from examples using genetic algorithms. In B. Bouchon-Meunier, R. R. Yager, and L. A. Zadeh, editors, *Fuzzy Logic and Soft Computing*, pages 11–20. World Scientific, 1995.

[13] J. H. Holland. Outline for a logical theory of adaptive systems. *Journal of the ACM*, 9(3):297–314, July 1962.

[14] J.-S. R. Jang and C.-T. Sun. Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, 83(3):378–406, March 1995.

[15] C. L. Karr. Genetic algorithms for fuzzy controllers. *AI Expert*, 6(2):26–33, February 1991.

[16] C. L. Karr, L. M. Freeman, and D. L. Meredith. Improved fuzzy process control of spacecraft terminal rendezvous using a genetic algorithm. In G. Rodriguez, editor, *Proceedings of Intelligent Control and Adaptive Systems Conference*, volume 1196, pages 274–288. SPIE, February 1990.

[17] O. L. Mangasarian, R. Setiono, and W.-H Goldberg. Pattern recognition via linear programming: Theory and application to medical diagnosis. In T. F. Coleman and Y. Li, editors, *Large-Scale Numerical Optimization*, pages 22–31. SIAM, 1990.

[18] O. L. Mangasarian, W. N. Street, and W. H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. Mathematical Programming Technical Report 94-10, University of Wisconsin, 1994.

[19] J. M. Mendel. Fuzzy logic systems for engineering: A tutorial. *Proceedings of the IEEE*, 83(3):345–377, March 1995.

[20] C. J. Merz and P. M. Murphy. UCI repository of machine learning databases, 1996.

[21] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Heidelberg, third edition, 1996.

[22] D. Nauck and R. Kruse. Neuro-fuzzy systems for function approximation. *Fuzzy Sets and Systems*, 101(2):261–271, January 1999.

[23] J. Paredis. Coevolutionary computation. *Artificial Life*, 2:355–375, 1995.

[24] C. A. Peña-Reyes and M. Sipper. Evolving fuzzy rules for breast cancer diagnosis. In *Proceedings of 1998 International Symposium on Nonlinear Theory and Applications (NOLTA'98)*, volume 2, pages 369–372. Presses Polytechniques et Universitaires Romandes, Lausanne, 1998.

[25] C. A. Peña-Reyes and M. Sipper. Designing breast cancer diagnostic systems via a hybrid fuzzy-genetic methodology. In *1999 IEEE International Fuzzy Systems Conference Proceedings*, volume 1, pages 135–139. IEEE Neural Network Council, 1999.

[26] C. A. Peña-Reyes and M. Sipper. A fuzzy-genetic approach to breast cancer diagnosis. *Artificial Intelligence in Medicine*, 17(2):131–155, October 1999.

[27] C. A. Peña-Reyes and M. Sipper. Applying Fuzzy CoCo to breast cancer diagnosis. In *Proceedings of the 2000 Congress on Evolutionary Computation (CEC00)*, volume 2, pages 1168–1175. IEEE Press, Piscataway, NJ, USA, 2000.

[28] C. A. Peña-Reyes and M. Sipper. Evolutionary computation in medicine: An overview. *Artificial Intelligence in Medicine*, 19(1):1–23, May 2000.

[29] W. Pedrycz, editor. *Fuzzy Evolutionary Computation*. Kluwer Academic Publishers, 1997.

[30] W. Pedrycz and J. Valente de Oliveira. Optimization of fuzzy models. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 26(4):627–636, August 1996.

[31] R. Poli. Introduction to evolutionary computation. http://www.cs.bham.ac.uk/~rmp/slide_book/, October 1996. visited: 16 March 1999.

[32] M. A. Potter. *The Design and Analysis of a Computational Model of Cooperative Coevolution*. PhD thesis, George Mason University, 1997.

[33] M. A. Potter and K. A. DeJong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, spring 2000.

[34] I. Rechenberg. Cybernetic solution path of an experimental problem. Farborough Hants: Royal Aircraft Establishment. Library Translation 1122, August 1965. English Translation of lecture given at the Annual Conference of the WGLR at Berlin in September, 1964.

[35] F. Russo. Evolutionary neural fuzzy systems for noise cancellation in image data. *IEEE Transactions on Instrumentation and Measurement*, 48(5):915–920, October 1999.

[36] H.-P. Schwefel. Kybernetische evolution als strategie der experimentelen forschung in der stromungstechnik. Master's thesis, Technical University of Berlin, March 1965.

[37] T. L. Seng, M. Bin Khalid, and R. Yusof. Tuning of a neuro-fuzzy controller by genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 29(2):226–236, April 1999.

[38] R. Setiono. Generating concise and accurate classification rules for breast cancer diagnosis. *Artificial Intelligence in Medicine*, 18(3):205 – 219, 2000.

[39] I. Taha and J. Ghosh. Evaluation and ordering of rules extracted from feedforward networks. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 221–226, 1997.

[40] M. D. Vose. *The Simple Genetic Algorithm*. MIT Press, Cambridge, MA, August 1999.

[41] P. Vuorimaa. Fuzzy self-organizing map. *Fuzzy Sets and Systems*, 66:223–231, 1994.

[42] R. R. Yager and D. P. Filev. *Essentials of Fuzzy Modeling and Control*. John Wiley & Sons., New York, 1994.

[43] R. R. Yager and L. A. Zadeh. *Fuzzy Sets, Neural Networks, and Soft Computing*. Van Nostrand Reinhold, New York, 1994.