ELSEVIER Artificial Intelligence in Medicine 17 (1999) 131–155

Artificial Intelligence in Medicine

www.elsevier.com/locate/artmed

A fuzzy-genetic approach to breast cancer diagnosis

Carlos Andrés Peña-Reyes, Moshe Sipper *

Logic Systems Laboratory, Swiss Federal Institute of Technology, IN-Ecublens, CH-1015 Lausanne, Switzerland

Received 20 September 1998; received in revised form 21 December 1998; accepted 2 February 1999

Abstract

The automatic diagnosis of breast cancer is an important, real-world medical problem. In this paper we focus on the Wisconsin breast cancer diagnosis (WBCD) problem, combining two methodologies—fuzzy systems and evolutionary algorithms—so as to automatically produce diagnostic systems. We find that our fuzzy-genetic approach produces systems exhibiting two prime characteristics: first, they obtain high classification performance (the best shown to date), with the possibility of attributing a confidence measure to the output diagnosis; second, the resulting systems involve a few simple rules, and are therefore (human-) interpretable.

Keywords: Fuzzy systems; Genetic algorithms; Breast cancer diagnosis

1. Introduction

1.1. Motivation

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.

* Corresponding author. Tel.: +41-21-6932658; fax: +41-21-6933705.
E-mail addresses: carlos.pena@di.epfl.ch (C.A. Peña-Reyes), moshe.sipper@di.epfl.ch (M. Sipper)

PII: S0933-3657(99)00019-6

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*. Moreover, it would be highly desirable to be in possession of a so-called *degree of confidence*: the system not only provides a binary diagnosis (benign or malignant), but also outputs a numeric value that represents the degree to which the system is confident about its response. 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 paper we combine two methodologies—fuzzy systems and evolutionary algorithms—so as 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, though these do not combine both methodologies in a hybrid way, as we do in this paper [2,7,14,29].

In the next two subsections we provide a brief overview of fuzzy systems and genetic algorithms (one of the central methodologies within the field of evolutionary algorithms). In Section 2 we describe the Wisconsin breast cancer diagnosis (WBCD) problem, which is the focus of our interest in this paper. This is followed by an exposition in Section 3 of evolutionary fuzzy modeling, which involves the application of genetic algorithms to the evolution of fuzzy systems. Section 4 then describes our particular evolutionary approach to the WBCD problem. In Section 5 we delineate our results, followed by concluding remarks in Section 6.

1.2. Fuzzy systems

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 [33]. A *fuzzy variable* (also called a *linguistic variable;* see Fig. 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, $\mu_{\text{label}}(u)$ to a given real value $u(\mathscr{R}$, 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:

$$\mu_{A \text{ and } B}(u) = \mu_A(u) \wedge \mu_B(u) = \min\{\mu_A(u),\ \mu_B(u)\},$$

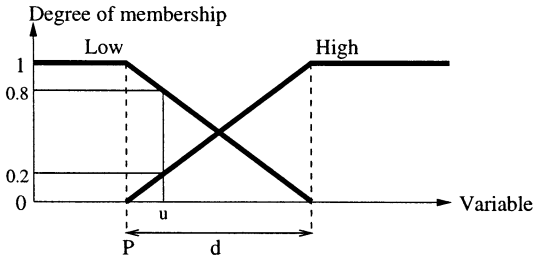

Fig. 1. Example of a fuzzy variable with two possible fuzzy values labeled *Low* and *High*, and orthogonal membership functions, plotted above as degree of membership versus input values. *P* and *d* define the start point and the length of membership function edges, respectively. The orthogonality 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 $\mu_{\text{Low}}(u) = 0.8$ and $\mu_{\text{High}}(u) = 0.2$ (as can be seen $\mu_{\text{Low}}(u) + \mu_{\text{High}}(u) = 1$).

$$\mu_{A \text{ or } B}(u) = \mu_A(u) \vee \mu_B(u) = \max\{\mu_A(u),\ \mu_B(u)\},$$

$$\mu_{\text{not } A}(u) = \ \neg\mu_A(u) = 1 - \mu_A(u),$$

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** Low, **then** increase ventilation fan speed.

A *fuzzy inference system* is a rule-based system that uses fuzzy logic, rather than Boolean logic, to reason about data [33]. Its basic structure includes four main components, as depicted in Fig. 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

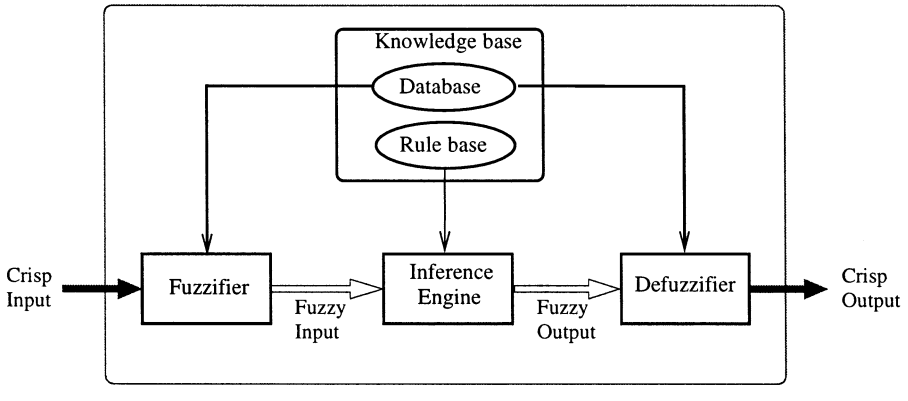

Fig. 2. Basic structure of a fuzzy inference system.

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

ensemble of fuzzy rules, known as the rule base, and an ensemble of membership functions, known as the database.

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 [32] (Aggregation is performed usually 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 [32]. 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 [11,16,18,31], genetic algorithms [9,17,24], and hybrid methods [8]. 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): 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 [32]. Common fuzzy operators are min, max, product, probabilistic sum, and bounded sum. The most common membership functions are triangular, trapezoidal, and bell-shaped. For defuzzification several methods have been proposed with the center of area (COA) and the mean of maxima (MOM) methods being the most popular [21,32].

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).

1.3. Genetic algorithms

The idea of applying the biological principle of natural evolution to artificial systems, introduced more than four decades ago, has seen impressive growth in the past few years. Usually grouped under the term *evolutionary algorithms* or *evolutionary computation*, we find the domains of genetic algorithms, evolution strategies, evolutionary programming, and genetic programming [6,15,23]. Such algorithms are common nowadays, having been successfully applied to numerous problems from different domains, including optimization, automatic programming, machine learning, economics, medicine, ecology, population genetics, and hardware design. In this paper we consider the evolutionary methodology known as genetic algorithms.

A genetic algorithm is an iterative procedure that involves a population of individuals, each one represented by a finite string of symbols, known as the genome, encoding a possible solution in a given problem space. This space, referred to as the search space, comprises all possible solutions to the problem at hand. Genetic algorithms are usually applied to spaces which are too large to be exhaustively searched. The symbol alphabet used is often binary, though this has been extended in recent years to include character-based encodings, real-valued encodings, tree representations, and other representations [23].

The standard genetic algorithm proceeds as follows: an initial population of individuals is generated at random or heuristically. Every evolutionary step, known as a generation, the individuals in the current population are decoded and evaluated according to some predefined quality criterion, referred to as the fitness, or fitness function. To form a new population (the next generation), individuals are selected according to their fitness. Many selection procedures are currently in use, one of the simplest being fitness-proportionate selection, where individuals are selected with a probability proportional to their relative fitness. This ensures that the expected number of times an individual is chosen is approximately proportional to its relative performance in the population. 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 are generated by genetically inspired operators, of which the most well known are crossover and mutation. Crossover is performed with probability p_c (the 'crossover probability' or 'crossover rate') between two selected individuals, called *parents*, by exchanging parts of their genomes (i.e. encodings) to form two new individuals, called *offspring*. In its simplest form, substrings are exchanged after a randomly selected crossover point. This operator tends to enable the evolutionary process to move toward 'promising' regions of the search space. The mutation operator is introduced to prevent premature convergence to local optima by randomly sampling new points in the search space. It is carried out by flipping bits at random, with some (usually small) probability p_m. Genetic algorithms are stochastic iterative processes that are not guaranteed to converge. The termination condition may be specified as some fixed, maximal number of generations or as the attainment of an acceptable fitness level. Fig. 3 presents the standard genetic algorithm in pseudo-code format.

2. The Wisconsin breast cancer diagnosis problem

In this section we present the medical diagnosis problem which is the object of our study.

Breast cancer is the most common cancer among women, excluding skin cancer. The presence of a breast mass[1] is an alert sign, but it does not always indicate a malignant cancer. Fine needle aspiration (FNA)[2] of breast masses is a cost-effective,

```
begin GA
   g:=0   { generation counter }
   Initialize population P(g)
   Evaluate population P(g)   { i.e., compute fitness values }
   while not done do
      g:=g+1
      Select P(g) from P(g − 1)
      Crossover P(g)
      Mutate P(g)
      Evaluate P(g)
   end while
end GA
```

Fig. 3. Pseudo-code of the standard genetic algorithm.

[1] Most breast cancers are detected as a lump or mass on the breast, by self-examination, by mammography, or by both [20].

[2] Fine needle aspiration is an outpatient procedure that involves using a small-gauge needle to extract fluid directly from a breast mass [20].

non-traumatic, and mostly non-invasive diagnostic test that obtains information needed to evaluate malignancy.

The Wisconsin breast cancer diagnosis (WBCD) database [22] is the result of the efforts made at the University of Wisconsin Hospital for accurately diagnosing breast masses based solely on an FNA test [19]. 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:

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 [27,28,30]. 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 preliminary work on the evolution of fuzzy rules showed that it is possible to obtain high performance, coupled with interpretability and a confidence measure [25].

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 [12,13]. 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.

Below we classify evolutionary fuzzy modeling techniques based on three types of parameters which compose the search space: structural, connective, and operational (Table 1). We then focus our attention on structural and connective parameters, presenting the three major evolutionary approaches: Michigan, Pittsburgh, and iterative rule learning. For a detailed bibliography on evolutionary fuzzy modeling up to 1996, the reader is referred to [1,4].

3.1. Applying evolution to fuzzy modeling

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 parameters search. Three of the four types of fuzzy parameters in Table 1 can be used to define targets for evolutionary fuzzy modeling: structural parameters, connective parameters, and operational parameters. As noted in Section 1.2, logical parameters are usually predefined by the designer based on experience.

3.1.1. 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.

Using as example the WBCD problem, an initial fuzzy rule base is defined by an expert. An example fuzzy rule in this case would be: **if** (v_2 **is** *Low)* **and** (v_6 **is** *Low)* **then** *(output* **is** *benign).* The evolutionary algorithm then fine-tunes the membership functions, i.e. the P and d values defining *Low* and *High* (Fig. 1).

One of the major shortcomings of knowledge tuning is its dependency on the initial setting of the knowledge base.

3.1.2. 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 [32]. 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.

3.1.3. 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 some 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. In the WBCD example, evolutionary structure learning is carried out by encoding within the genome an entire fuzzy system (this is known as the Pittsburgh approach, see below).

Structure learning permits to specify other criteria related to the interpretability of the system, such as the number of membership functions and the number of rules. On the other hand, the strong interdependency among the parameters involved in this form of learning may slow down, or even prevent altogether, the convergence of the genetic algorithm.

3.2. Three approaches to behavior and structure learning

Both connective and structural parameters modeling 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 [23]. A more recent method has been proposed specifically for fuzzy modeling: the iterative rule learning approach [10]. These three approaches are presented below.

3.2.1. 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

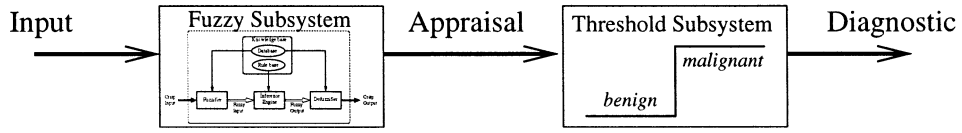

Fig. 4. Proposed diagnosis system. Note that the fuzzy subsystem displayed to the left is in fact the entire fuzzy inference system of Fig. 2.

good system behavior. It necessitates an effective credit assignment policy to ascribe fitness values to individual rules.

3.2.2. 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.

3.2.3. 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.

4. Evolving fuzzy systems for the WBCD problem

The solution scheme we propose for the WBCD problem is depicted in Fig. 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.

In order to evolve the fuzzy model we must make some preliminary decisions about the fuzzy system itself and about the genetic algorithm encoding. In this section we describe our choices, followed in the next section by a presentation of our results.

4.1. Fuzzy system parameters

Previous knowledge about the WBCD problem and about some of the extant rule-based models represents valuable information to be used for our choice of fuzzy parameters. When defining our setup we took into consideration the following results, described in previous works:

- *Small number of rules*. Systems with no more than four rules have been shown to obtain high performance [25,27].
- *Small number of variables*. Rules with no more than four antecedents have proven adequate [25,28,30].
- *Monotonicity of the input variables*. Simple observation of the input and output spaces shows that higher-valued variables are associated with malignancy.

Some fuzzy models forgo interpretability in the interest of improved performance. Where medical diagnosis is concerned, interpretability—also called linguistic integrity—is the major advantage of fuzzy systems. This motivated us to take into account the following semantic criteria, defining constraints on the fuzzy parameters [5,26]:

- *Distinguishability*. Each linguistic label should have semantic meaning and the fuzzy set should clearly define a range in the universe of discourse. For example, to describe variable *Clump thickness* (v_1) we can use two labels: *Thick* and *Diffuse* (we opted for *Low* and *High*). Their membership functions are defined using parameters P and d (Fig. 1).
- *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. For example, the following would be considered an adequate rule:

  if (v_1 **is** *High*) **and** (v_2 **is** *High*) **and** (v_4 **is** *High*) **and** (v_6 **is** *Low*)

  and (v_8 **is** *Low*) **then** (*output* **is** *benign*).

- *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 Fig. 1, we see that any value along the x-axis belongs to at least one fuzzy set *(Low*, *High*, or both); 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 Fig. 1, we observe that both *Low* and *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 (e.g. in Fig. 1 a *Low* membership value of 0.8 entails a *High* membership value of 0.2).

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

(i) Logical parameters

- Reasoning mechanism: singleton-type fuzzy system, meaning that consequents of rules (i.e. output membership functions) are real values (also called singletons), rather than fuzzy ones.
- Fuzzy operators: min and max.
- Input membership function type: orthogonal, trapezoidal (see Fig. 1).
- Defuzzification method: weighted average.

(ii) *Structural parameters*

- Relevant variables: there is insufficient a priori knowledge to define them, therefore this will be one of the genetic algorithm's objectives.
- Number of input membership functions: two membership functions, denoted *Low* and *High* are used (Fig. 1).
- Number of output membership functions: two singletons are used, corresponding to the *benign* and *malignant* diagnostics.
- Number of rules: in our approach, this is a user-configurable parameter. Based on our previous results [25], we limited the number of rules to be between 1 and 5. The rules themselves are to be found by the genetic algorithm.

(iii) Connective parameters

- Antecedents of rules: to be found by the genetic algorithm.
- Consequent of rules: the algorithm finds rules for the *benign* diagnostic; the *malignant* diagnostic is an else condition (see below).
- Rule weights: active rules have a weight of value 1, and the *else* condition has a weight of 0.25.

(iv) Operational parameters

- Input membership function values: to be found by the genetic algorithm.
- Output membership function values: following the WBCD database, we used a value of 2 for *benign* and 4 for *malignant*.

4.2. The genetic algorithm

We apply Pittsburgh-style structure learning, using a genetic algorithm to search for three parameters. The genome, encoding relevant variables, input membership function values, and antecedents of rules, is constructed 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 (Fig. 1).
- Antecedents. 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** 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),

Table 2
Parameters encoding of an individual's genome[a]

Parameter	Values	Bits	Quantity	Total bits
P	(1–8)	3	9	27
d	(1–8)	3	9	27
A	(0–3)	2	$9N_r$	$18N_r$

[a] 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 and 5, and is fixed during the genetic algorithm run).

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 parameters encoding, which together form a single individual's genome. Fig. 5 shows a sample genome.

To evolve the fuzzy inference system, we used a genetic algorithm with a fixed population size of 200 individuals, and fitness-proportionate selection (Section 1.3). The algorithm terminates when the maximum number of generations, G_{max} is reached (we set $G_{max} = 2000 + 500(N_r$, i.e. dependent on the number of rules used

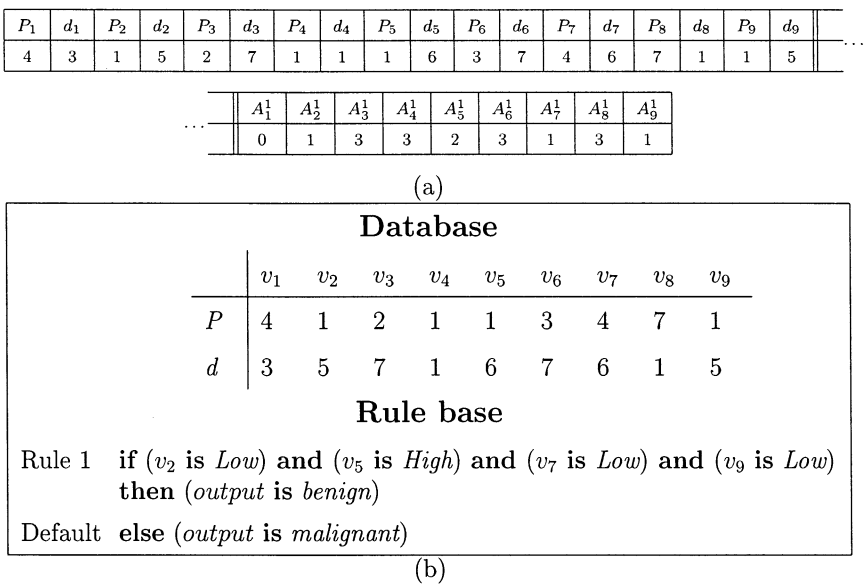

Fig. 5. Example of a genome for a single-rule system, (a) Genome encoding. The first 18 positions encode the parameters P and d for the nine variables v_1–v_9. The rest encode the membership function applicable for the nine antecedents of the rule; (b) Interpretation. Database and rule base of the single-rule system encoded by (a). The parameters P and d are interpreted as illustrated in Fig. 1.

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.

5. Results

This section describes the results obtained when applying the methodology described in Section 4. We first delineate in Section 5.1 the success statistics relating to the evolutionary algorithm. Then, in Section 5.2, we describe in full three evolved fuzzy systems—a three-rule system, a two-rule system, and a one-rule system—that exemplify our approach. In Section 5.3, we discuss the issue of obtaining a confidence measure of the system's output, going beyond a mere binary, benign–malignant classification. Finally, in Section 5.4, we briefly describe two further experiments that we carried out.

5.1. The genetic algorithm…

The evolutionary experiments performed fall into three categories, in accordance with the data repartitioning into two distinct sets: training set and test (or evaluation) set. The three experimental categories are: (1) training set contains all 683 cases of the WBCD database, while the test set is empty; (2) training set contains 75% of the WBCD cases, and the test set contains the remaining 25% of the cases; (3) training set contains 50% of the WBCD cases and the test set contains the remaining 50% of the cases. In the last two categories, the choice of training-set cases is done randomly, and is performed anew at the outset of every evolutionary run. The number of rules per system was also fixed at the outset, to be between one and five, i.e. evolution seeks a system with an a priori given number of rules (the choice of number of rules per system determines the final structure of the genome, as presented in Table 2).

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, eight runs found systems whose performance exceeds 97.5%; these results are

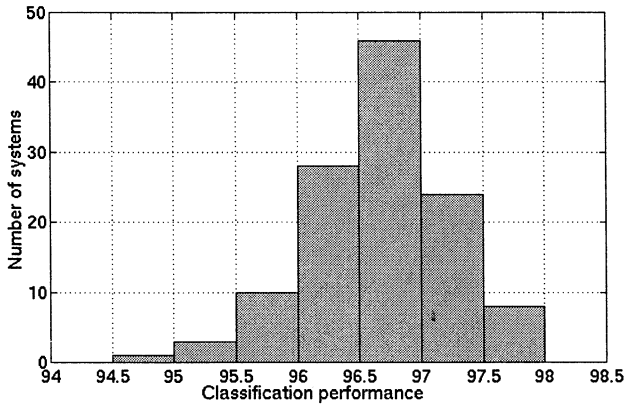

Fig. 6. Summary of results of 120 evolutionary runs. The histogram depicts the number of systems exhibiting a given performance level at the end of the evolutionary run. The performance considered is that of the best individual of the run, measured as the overall percentage of correctly classified cases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

summarized in Fig. 6. Table 3 presents the average performance obtained by the genetic algorithm over all 120 evolutionary runs, divided according to the three experimental categories discussed above (a more detailed account of our results can be found in Table 7, which lists the top evolved 45 systems).

Table 4 compares our best systems with the top systems obtained by four other rule-based diagnostic approaches. The first three approaches, those of Setiono [27], Setiono and Liu [28], and Taha and Ghosh [30], involve Boolean rule bases extracted from trained neural networks; the last approach is our own previous work [25]. The evolved fuzzy systems described in this paper can be seen to surpass those obtained by these four previous approaches in terms of both performance and simplicity of rules. As shown in Table 4, we obtained the highest-performance systems for all five rule-base sizes, i.e. from one-rule systems all the way up to

Table 3
Summary of results of 120 evolutionary runs, divided according to the three experimental categories discussed in the text (i.e. the three classes which differ in the training-set to test-set ratio)[a]

Training/test ratio (%)	Performance			Number of variables
	Training set (%)	Test set (%)	Overall (%)	
100/0	–	–	96.97	3.32
75/25	97.00	96.02	96.76	3.46
50/50	97.71	94.73	96.23	3.41

[a] The table lists the average performance over all 120 runs, where the averaging is done over the best individual of each run. The performance value denotes the percentage of cases correctly classified. Three such performance values are shown, (1) performance over the training set; (2) performance over the test set; and (3) overall performance, considering the entire database. In addition, the average number of variables per rule is also shown.

Table 4
Comparison of the best systems evolved by our approach with the top systems obtained by four other rule-based diagnostic approaches[a]

Rules-per-system	Setiono [27] (%)	Setiono and Liu [28] (%)	Taha and Ghosh [30](%)	Peña and Sipper [25](%)	This work (%)
1	95.42 (2)	–	–	96.35 (3)	97.07 (4)
2	–	–	–	96.65 (7)	97.36 (3)
3	97.14 (4)	97.21 (4)	–	–	97.80 (4.7)
4	–	–	–	–	97.80 (4.8)
5	–	–	96.19 (1.8)[b]	–	97.51 (3.4)

[a] The first three approaches—those of Setiono [27], Setiono and Liu [28], and Taha and Ghosh [30]—involve Boolean rule bases extracted from trained neural networks; the last approach is our own previous work [25]. Shown above are the classification performance values of the top systems obtained by these approaches, along with the average number of variables-per-rule given in parentheses. Results are divided into five classes, in accordance with the number of rules-per-system, going from one-rule systems to five-rule ones.

[b] Note that Taha and Ghosh [30] obtained slightly better results for the five-rules case by directly using their trained neural networks, rather than the extracted rule-based systems. Herein, our interest lies with these latter, rule-based systems.

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*)

Fig. 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.

five-rule systems. Not only is high performance exhibited, but, moreover, our fuzzy approach enables the introduction of a confidence measure of the diagnostic decision (see Section 5.3). In contrast, the Boolean rule-based systems [27,28,30] provide but a single binary value, indicating whether the case in question is benign or malignant. Compared with our previous work [25], the current approach not only improves performance, but also obtains systems with less antecedents per rule (which are thus more easily comprehensible).

5.2. …and the fuzzy systems it discovered

We next describe three of our top-performance systems, which serve to exemplify the solutions found by our evolutionary approach. The first system, delineated in Fig. 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 one over all 120 evolutionary runs. It obtains 98.7% correct classification rate over the benign cases, 97.07% correct classification rate over the malignant cases[3], and 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 (Fig. 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).

[3] The WBCD database contains 444 benign cases and 239 malignant cases.

Database

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

Rule base

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

Rule 2 **if** (v_2 **is** *Low*) **and** (v_5 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **then** (*output* **is** *benign*)

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

Fig. 8. The best evolved, fuzzy diagnostic system with two rules. It exhibits an overall classification rate of 97.36%, and an average of three variables per rule.

Can the genetic algorithm automatically discover a two-rule system, i.e. without recourse to any post-processing (such as that described in the previous paragraph)? Our results have shown that this is indeed the case; one such solution is presented in Fig. 8. It obtains 97.3% correct classification rate over the benign cases, 97.49% correct classification rate over the malignant cases, and an overall classification rate of 97.36%.

Finally, Fig. 9 delineates the best one-rule system found through our evolutionary approach. It obtains 97.07% correct classification rate over the benign cases, 97.07% correct classification rate over the malignant cases, and an overall classification rate of 97.07%.

5.3. Diagnostic confidence

Up until now we have been using the evolved fuzzy systems to ultimately produce a binary classification value—*benign* or *malignant*—with no finer gradations. Going back to Fig. 4, we note that the diagnostic system comprises in fact two subsystems: the first subsystem consists of the (evolved) fuzzy system, which, upon

Database

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

Rule base

Rule 1 **if** (v_1 **is** *Low*) **and** (v_2 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **then** (*output* **is** *benign*)

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

Fig. 9. The best evolved, fuzzy diagnostic system with one rule. It exhibits an overall classification rate of 97.07%, and a rule with four variables.

presentation of an input (in our case, a WBCD database entry) proceeds to pro-duce a *continuous* appraisal value; this value is then passed along to the second subsystem—the threshold unit—which produces the final binary output *(benign* or *malignant)*. The first subsystem (the fuzzy system) is the one evolved in our approach. The threshold subsystem simply outputs *malignant* if the appraisal value is above a fixed threshold value, and outputs *benign* otherwise. The threshold value is assigned by the user through knowledge of the problem at hand.

To gain an intuitive understanding of how a classification value is computed, let us sketch a simple example. Referring to the system of Fig. 4, assume that the following values are presented as inputs (these represent case # 145 of the WBCD database):

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
Value	4	3	1	1	2	1	4	8	1

The membership value of each variable is then computed in accordance with the (evolved) database of Fig. 7:

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
μ_{Low}	0.8	1	1	1	1	1	1	0.4	1
μ_{High}	0.2	0	0	0	0	0	0	0.6	0

This completes the fuzzification phase (the 'fuzzifier' unit of Fig. 2). Having computed these membership values, the inference engine (Fig. 2) can now go on to compute the so-called truth value of each rule. This truth value is computed by applying the fuzzy 'and' operator (Section 1.2) to combine the antecedent clauses (the membership values) in a fuzzy manner; this results in the output truth value, namely, a continuous value which represents the rule's degree of activation. Thus, a rule is not merely either activated or not, but in fact is activated to a certain degree represented by a value between 0 and 1. In our example, the rule activation values are as follows (remember that we 'chucked out' rule 2, since it was found to never fire):

	Rule 1	Rule 3	Default
Truth value	0.4	0.4	0.25

The inference engine (Fig. 2) now goes on to apply the aggregation operator (Section 1.2), combining the continuous rule activation values to produce a fuzzy output with a certain truth value (the point marked 'fuzzy output' in Fig. 2). The defuzzifier then kicks in (Fig. 2), producing the final continuous value of the fuzzy inference system; this latter value is the appraisal value that is passed on to the threshold unit (Fig. 4). In our example the appraisal value is 2.48.

In general, the appraisal value computed by our evolved fuzzy systems is in the range [2,4]. We chose to place the threshold value at 3, with inferior values classified as benign, and superior values classified as malignant. Thus, in our example, the appraisal value of 2.48 is classified as benign—which is correct.

This case in the WBCD database produces an appraisal value (2.48) which is among the closest to the threshold value. Most other cases result in an appraisal value that lies close to one of the extremes (i.e. close to either 2 or 4). Thus, in a sense, we can say that we are somewhat less confident where this case is concerned, with respect to most other entries in the WBCD database; specifically, the appraisal value can accompany the final output of the diagnostic system, serving as a confidence measure. This demonstrates a prime advantage of fuzzy systems, namely, the ability to output not only a (binary) classification, but also a measure representing the system's confidence in its output. (The three-rule system of Fig. 7 computes intermediate appraisal values (between, say, 2.4 and 3.6) for 39 cases; these might thus be considered the cases for which we are somewhat less confident about the output.)

5.4. Further experiments

In this section we describe two further experiments carried out; these are aimed at searching for systems with yet better performance than obtained hitherto—though possibly at the expense of some other aspect of the resulting system.

As noted in Section 4.1, fuzzy systems offer a major advantage in terms of (possible) linguistic integrity, i.e. interpretability by humans. With this goal in mind, the experiments described previously were constrained: we limited both the number of rules per system, as well as the number of variables per rule. This latter constraint was incorporated by favoring systems with few variables-per-rule via the F_v coefficient: lower F_v, meaning fewer variables-per-rule, entails higher overall fitness.

Can higher-performance systems be obtained by eliminating the F_v factor (albeit at the cost of reduced interpretability due to more complicated rules)? This was the aim of our first of the two experiments described herein. We eliminated not only the F_v measure but also the F_e factor, the resulting fitness function thus containing solely F_c. Our intent was to provide selection pressure for but one goal: overall classification performance. With this aim in mind we were also more 'generous' with the number of rules per system: whereas previously this was set to a (fixed) value between one and five, herein we set this value to be between three and seven.

We performed a total of 31 evolutionary runs, the results of which are summarized in Table 5. We note that our previous best system (Fig. 7) obtained 97.8%

Table 5
Results of evolutionary runs in which the variables-per-rule constraint has been removed[a]

Rules-per-system	Best system (%)	Average (%)
3	97.66 (5.3)	97.49 (5.4)
4	98.24 (5.8)	97.63 (5.4)
5	97.95 (6)	97.63 (5.6)
6	98.10 (6.2)	97.77 (5.4)
7	97.95 (5)	97.88 (5.2)
Total	98.24 (5.8)	97.68 (5.4)

[a] Results are divided into five classes, in accordance with the number of rules-per-system, going from three-rule systems to seven-rule ones. We performed 5–7 runs per class, totaling 31 runs in all; shown above are the resulting best systems as well as the average per class. Results include the overall classification performance and the average number of variables-per-rule in parentheses.

overall classification performance, while Table 5 shows an evolved system with a 98.24% classification rate. This latter system is thus able to correctly classify three additional cases. This small improvement in performance carries, however, a price: the slightly better system comprises four rules with an average of 5.8 variables per rule, whereas the previous one (Fig. 7) contains but three rules with an average of 4.7 variables per rule; we have thus traded off interpretability for performance. The judgment of whether this is worthwhile or not is entirely dependent on the human user. It is noteworthy that this choice (interpretability versus performance) can be easily implemented in our approach.

As explained in Section 4.1, the active rules diagnose benignity, with the default diagnosis being malignancy; this means that the if conditions have *benign* as a consequent, whereas the else condition has *malignant* as a consequent. Our second experiment sought to find out what would happen if this were reversed, i.e. could

Table 6
Results of evolutionary runs in which the default diagnosis is *benign* (rather than *malignant*)[a]

Rules-per-system	Best system (%)	Average
1	94.73 (2)	94.44 (2)
2	96.93 (1.5)	96.34 (1.8)
3	97.36 (2)	96.57 (1.7)
4	97.07 (1.8)	97.00 (2)
5	97.80 (2.8)	96.52 (2.2)
Total	97.80 (2.8)	97.68 (1.9)

[a] Results are divided into five classes, in accordance with the number of rules-per-system, going from one-rule systems to five-rule ones. We performed 4–6 runs per class, totaling 27 runs in all; shown above are the resulting best systems as well as the average per class. Results include the overall classification performance and the average number of variables-per-rule in parentheses. Note that we slightly modified the genomic representation (Section 4.2), such that if consequents are *malignant*, and the else consequent is *benign*.

Database

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

Rule base

Rule 1 **if** (v_2 **is** *High*) **and** (v_7 **is** *High*) **then** (*output* **is** *malignant*)

Rule 2 **if** (v_2 **is** *High*) **and** (v_3 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_8 **is** *High*) **and** (v_9 **is** *Low*) **then** (*output* **is** *malignant*)

Rule 3 **if** (v_3 **is** *High*) **and** (v_6 **is** *High*) **then** (*output* **is** *malignant*)

Rule 4 **if** (v_3 **is** *Low*) **and** (v_5 **is** *High*) **and** (v_8 **is** *High*) **then** (*output* **is** *malignant*)

Rule 5 **if** (v_1 **is** *High*) **and** (v_6 **is** *High*) **then** (*output* **is** *malignant*)

Default **else** (*output* **is** *benign*)

Fig. 10. The best evolved, fuzzy diagnostic system with active rules encoding malignant cases. It exhibits an overall classification rate of 97.80%, and an average of 2.8 variables per rule.

better systems be evolved with benignity as the default diagnosis? Table 6 delineates the results of 27 evolutionary runs. While we did not improve upon the results of the malignancy-default systems of Section 5.1, we did note a tendency toward a smaller number of variables-per-rule. The highest-performance system found in this experiment is fully specified in Fig. 10. It comprises five rules with an average of 2.8 variables-per-rule, exhibiting the same overall performance (97.8%) as the three-rule, 4.7 average-variables-per-rule system of Fig. 7. This nicely illustrates the tradeoff between these two parameters: number of rules per system and average number of variables per rule.

6. Concluding remarks

In this paper we applied a combined fuzzy-genetic approach 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), with the possibility of attributing a confidence measure to the output diagnosis; second, the resulting systems involve a few simple rules, and are therefore interpretable.

Our experience to date suggests that the fuzzy-genetic approach is highly effective where such medical diagnosis problems are concerned. We are currently pursuing two avenues of research: (1) application of the fuzzy-genetic approach to more complex diagnosis problems; and (2) improving and expanding upon the methodology presented herein (e.g. by making use of recent advances in evolutionary computation). Our underlying goal is to provide an approach for automatically

Table 7
Summary of top 45 evolutionary runs (of 120) described in Section 5.1, divided according to the three experimental categories discussed in the text (i.e. the three classes which differ in the training-set to test-set ratio)[a]

	100%/0%			75%/25%					50%/50%				
Rules	Fitness	Performance	Variables	Fitness	Training	Test	Performance	Variables	Fitness	Training	Test	Performance	Variables
1	0.9548	96.78	3	0.9553	97.46	95.91	97.07	4	0.9637	97.66	95.60	96.63	3
	0.9548	96.78	3	0.9597	97.27	95.32	96.78	3	0.9607	97.37	95.01	96.19	3
	0.9533	96.63	3	0.9557	96.88	96.49	96.78	3	0.9607	97.37	95.01	96.19	3
2	0.9576	97.36	3.5	0.9598	97.27	97.66	97.36	3	0.9603	97.95	95.89	96.93	4
	0.9593	97.22	3	0.9548	97.07	96.49	96.93	3.5	0.9579	97.08	96.77	96.93	3
	0.9578	97.07	3	0.9648	97.46	94.74	96.78	2.5	0.9636	97.95	94.43	96.19	3.5
3	0.9548	97.8	4.67	0.9577	97.27	97.66	97.36	3.33	0.9659	97.66	95.89	96.78	2.67
	0.9554	97.66	4.33	0.9557	97.27	97.08	97.22	3.67	0.9546	97.37	95.89	96.63	4
	0.9594	97.22	3	0.9577	97.27	95.91	96.93	3.33	0.9626	97.95	95.01	96.49	3.76
4	0.9543	97.8	4.75	0.952	97.27	98.25	97.51	4.25	0.9755	99.12	95.6	97.36	3.5
	0.9529	97.51	4.5	0.9563	97.07	96.49	96.93	3.25	0.971	98	95.89	97.36	3.75
	0.9594	97.22	3	0.9591	97.66	94.15	96.78	3.75	0.965	98.25	95.31	96.78	3.75
5	0.9599	97.51	3.4	0.9587	97.66	97.08	97.51	3.8	0.9586	97.66	96.77	97.22	3.8
	0.9483	97.36	5	0.9575	97.66	97.08	97.51	4	0.9756	98.83	94.72	96.78	3
	0.9584	97.36	3.4	0.9561	97.27	96.49	97.07	3.6	0.9623	98.25	95.01	96.63	4.2

[a] For each of the 45 evolved systems, the table lists its fitness value, its performance, and its average number of variables-per-rule. As explained in Section 5.1, the performance value denotes the percentage of cases correctly classified. Three such performance values are shown, (1) performance over the training set; (2) performance over the test set; and (3) overall performance, considering the entire database.

producing high-performance, interpretable systems for real-world diagnosis problems.

Acknowledgements

We thank Andrés Pérez-Uribe for helpful remarks.

References

[1] Alander JT. An indexed bibliography of genetic algorithms with fuzzy logic. In: Pedrycz W, editor. Fuzzy Evolutionary Computation. Dordrecht: Kluwer, 1997:299–318.
[2] Bellazzi R, Ironi L, Guglielmann R, Stefanelli M. Qualitative models and fuzzy systems: an integrated approach for learning from data. Artif Intell Med 1998;14(1–2):5–28.
[3] Bennett KP, Mangasarian OL. Neural network training via linear programming. In: Pardalos PM, editor. Advances in Optimization and Parallel Computing. Elsevier, 1992:56–7.
[4] Cordon O, Herrera F, Lozano M. On the combination of fuzzy logic and evolutionary computation: a short review and bibliography. In: Pedrycz W, editor. Fuzzy Evolutionary Computation. Kluwer, 1997:33–56.
[5] Espinosa JJ, Vandewalle J. Constructing fuzzy models with linguistic integrity. IEEE Transactions on Fuzzy Systems 1999, submitted for publication.
[6] Fogel DB. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. Piscataway, NJ: IEEE Press, 1995.
[7] Fogel DB, Wasson III EC, Boughton EM, Porto VW. Evolving artificial neural networks for screening features from mammograms. Artif Intell Med 1998;14(3):317.
[8] Fukuda T, Shimojima K. Fusion of fuzzy, NN, GA to the intelligent robotics. Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics, Vol. 3. IEEE, 1995:2892–2897.
[9] Heider H, Drabe T. Fuzzy system design with a cascaded genetic algorithm. Proceedings of 1997 IEEE International Conference on Evolutionary Computation. IEEE and IEEE Neural Network Council and Evolutionary Programming Society, 1997:585–588.
[10] Herrera F, Lozano M, Verdegay JL. Generating fuzzy rules from examples using genetic algorithms. In: Bouchon-Meunier B, Yager RR, Zadeh LA, editors. Fuzzy Logic and Soft Computing. World Scientific, 1995:11–20.
[11] Jang J-S R, Sun C-T. Neuro-fuzzy modeling and control. Proceedings of the IEEE. 1995 83 (3):378–406.
[12] Karr CL. Genetic algorithms for fuzzy controllers. AI Expert 1991;6(2):26–33.
[13] Karr CL, Freeman LM, Meredith DL. Improved fuzzy process control of spacecraft terminal rendezvous using a genetic algorithm. In: Rodriguez G, editor. Proceedings of Intelligent Control and Adaptive Systems Conference, Vol. 1196. SPIE, 1990:274–288.
[14] Kovalerchuk B, Triantaphyllou E, Ruiz JF, Clayton J. Fuzzy logic in computer-aided breast cancer diagnosis: analysis of lobulation. Artif Intell Med 1997;11(1):75–85.
[15] Koza JR. Genetic Programming. Cambridge, MA: MIT Press, 1992.
[16] Lee K-M, Kwak D-H, Lee-Kwang H. Fuzzy inference neural network for fuzzy model tuning. IEEE Trans Syst Man Cybern 1996;26(4):637–45.
[17] Lee MA, Takagi H. Integrating design stages of fuzzy systems using genetic algorithms. 1993 IEEE International Conference on Fuzzy Systems. IEEE, 1993:612–617.
[18] Lin C-T, Lee CSG. Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems. IEEE Trans Fuzzy Syst 1994;2(1):46–63.

[19] Mangasarian OL, Setiono R, Goldberg W-H. Pattern recognition via linear programming: Theory and application to medical diagnosis. In: Coleman TF, Li Y, editors. Large-Scale Numerical Optimization. SIAM, 1990:22–31.
[20] Mangasarian OL, Street WN, Wolberg WH. Breast cancer diagnosis and prognosis via linear programming. Mathematical Programming Technical Report 94-10, University of Wisconsin, 1994.
[21] Mendel JM. Fuzzy logic systems for engineering: a tutorial. Proceedings of the IEEE, 83(3):345-377, 1995.
[22] Merz CJ, Murphy PM. UCI repository of machine learning databases. http://www.ics.uci.edu/ ~ mlearn/MLRepository.html, 1996.
[23] Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs, 3rd edition. Heidelberg: Springer-Verlag, 1996.
[24] Nawa NE, Hashiyama T, Furuhashi T, Uchikawa Y. A study on fuzzy rules discovery using pseudo-bacterial genetic algorithm with adaptive operator. Proceedings of 1997 IEEE International Conference on Evolutionary Computation. IEEE and IEEE Neural Network Council and Evolutionary Programming Society, 1997.
[25] Peña-Reyes CA, Sipper M. Evolving fuzzy rules for breast cancer diagnosis. Proceedings of 1998 International Symposium on Nonlinear Theory and Applications (NOLTA'98), Vol. 2. Lausanne: Presses Polytechniques et Universitaires Romandes, 1998:369–372.
[26] Pedrycz W, Valente de Oliveira J. Optimization of fuzzy models. IEEE Trans Syst Man Cybern 1996;26(4):627–36.
[27] Setiono R. Extracting rules from pruned neural networks for breast cancer diagnosis. Artificial Intelligence in Medicine 1996:37–51.
[28] Setiono R, Liu H. Symbolic representation of neural networks. IEEE Computer 1996;29(3):71–7.
[29] Sierra B, Larranaga P. Predicting survival in malignant skin melanoma using Bayesian networks automatically induced by genetic algorithms. An empirical comparison between different approaches. Artif Intell Med 1998;14(1–2):215–36.
[30] Taha I, Ghosh J. Evaluation and ordering of rules extracted from feedforward networks. Proceedings of the IEEE International Conference on Neural Networks. 1997:221–226.
[31] Vuorimaa P. Fuzzy self-organizing map. Fuzzy Sets Syst 1994;66:223–31.
[32] Yager RR, Filev DP. Essentials of Fuzzy Modeling and Control. Wiley, 1994.
[33] Yager RR, Zadeh LA. Fuzzy Sets, Neural Networks, and Soft Computing. New York: Van Nostrand Reinhold, 1994.