Integration of Machine Learning and Knowledge Acquisition with a Genetic Algorithm

Vladimir Estivill-Castro
School of Information Systems,
Queensland University of Technology,
Brisbane, 4000, Australia.

vlad@icis.qut.edu.au
phone: (61-7) 3864-1944
fax: (61-7) 3864-1969

February 25, 1997

Abstract

Inductive inference techniques are showing signs of maturity as applications with real-world data are emerging. Techniques that allow symbolic representation of the acquired knowledge allow for its validation, revision and understanding by human experts. EFOPREL v 1.1(Evolutionary First Order PREdicate Logic) is an inductive and supervised learning system of first order predicate logic rules using a genetic algorithm for searching the space of theories explaining the set of examples. EFOPREL contributes to computer assisted knowledge acquisition because it allows discovery of flexible and/or alternative rules from examples. Moreover, EFOPREL integrates sources of knowledge and establishes collaboration between the genetic searcher and the human expert.

Topics: Knowledge Acquisition, Inductive Inference, Machine Learning, Genetic Algorithms, Logic rules.

1 Introduction

Building an expert system involves *Knowledge Acquisition*; that is, the process of eliciting, analyzing, and interpreting the knowledge that human experts use when solving problems and transforming this knowledge into a suitable machine representation [27]. Experience has shown that this process of knowledge acquisition is both difficult and time consuming and it is often a major bottleneck in the production of expert systems [23].

Inductive inference or machine learning techniques are showing signs of maturity as applications with real-world data are emerging [35]. Programs that induce classifiers from a set of preclassified examples constitute one approach to avoiding the difficulties associated with acquiring knowledge from domain experts. Rather than engaging in the typically difficult task of compiling concise and correct knowledge from human experts and representing such knowledge as situation-action rules, machine learning programs attempt to induce knowledge by examining examples of solved cases. The situations are described in terms of a set of attributes, and the actions are represented in terms of a fixed set of classes. The first uses of inductive classifiers for knowledge acquisition were reported in restricted domains [22]; however, their success on a variety of industrial environments and complex domains [35] has stimulated research in this area [49].

Normally, the space of all possible hypotheses consistent with the data is too large to search exhaustively. A practical, learning algorithm, therefore, must employ heuristics to help it discover useful rules. Furthermore, a learning program must be fast and efficient because it typically has to deal with large training sets of examples. Because of its simplicity and efficiency, Quinlan's ID3 [38] algorithm for induction of rules represented as a decision tree has gained widespread popularity in the machine learning community. Much research has been devoted to improving ID3 to make it more applicable to real-world problems by finding ways to deal with noise [37], avoiding some of its inherent weaknesses [8], making it incremental (such as ID4 and ID5 [46]) and handling inconclusive data [42]. ID3 search of the space of decision trees is essentially a hill-climbing approach guided locally by an information theoretic measure. Other machine learning algorithms are also guided by similar heuristics and may get trapped in local optima. Since genetic algorithms are less sensitive to local optima, they have been used as an alternative to conduct the search (for example, the AQ [32] algorithm for learning logic rules has been adapted for genetic search [1]).

Properties of concepts can be inferred or explicitly described when knowledge is represented as rules. For the process of knowledge acquisition, knowledge representation by logical rules has the very important feature that knowledge is explicit for human understanding and thus, rules contribute to synthesizing knowledge verifiable by human experts. Many other arguments have been provided to use logic rules for knowledge representation [31, 41, 49]. However, although rules may be derived from ID3's decision trees [39], this allows for little flexibility. For example, different rules with near-by accuracy are hard to obtain. Alternative rules for describing a concept from a different set of attributes are natural in many applications [50]. Another aspect that we overcome here is the limitations of expressivity found on the rules when represented as decision trees [49].

EFOPREL v. 1.1(Evolutionary First Order PREdicate Logic) is a inductive and supervised learning system of first order predicate logic rules using a genetic algorithm for searching the space of theories explaining the set of examples. The system is similar in spirit to BEAGLE [17, 18] and GABIL [12, 14] and other genetic algorithms for inducing logic rules [1, 4, 15, 24]. However, the rules generated by EFOPREL are always first order predicate logic rules with logical connectives such as AND, OR and NOT. EFOPREL results in more intelligible rules because, as opposed to BEAGLE, EFOPREL does not mix data types. As opposed to GABIL and other systems [4], EFOPREL can handle both continuous and discrete attributes. EFOPREL also has a clear strategy to simplify rules. These features make EFOPREL efficient and productive since the induced rules are easier to understand. EFOPREL facilitates knowledge elicitation. EFOPREL genetic search allows a more robust and flexible search of the space of rules, as well as the generation of alternative rules for the same concept.

EFOPREL intends to be a tool to improve the process of knowledge elicitation and knowledge acquisition. Rather that being at the end of the chain in the process of building expert systems, EFOPREL's application environment is the early stages of construction. Therefore, EFOPREL's spirit differs from inductive learning systems in that the source of knowledge is not restricted to examples. Knowledge acquisition techniques usually assume that the only source of knowledge are human experts. However, on early stages of knowledge elicitation, a natural assumption is that both knowledge sources (training data sets and human experts) are available. EFOPREL integrates the participation of human experts in the inference from examples. EFOPREL allows a human user to discover and to confirm explicit rules from examples in a collaborative and interactive mode. Sometimes an improvement to the rules will be provided as a modification by the human expert to the rules discovered by EFO-

PREL. Other times, human experts may formulate rules out of their experience and will test them with EFOPREL or submit them to EFOPREL so they are evolved further. EFOPREL may work as a tutor for the novice or as a colleague to the more experienced human. This integrates the sources of knowledge and contributes to the elicitation, exchange and validation of knowledge. Although we believe that EFOPREL's spirit of integrating knowledge sources is one fundamental feature of the system, there are other aspects that contribute to our proposal of such a system. EFOPREL's design allows us to handle attributes in a continuous domain in fuzzy discrete intervals that are amendable to human understanding. This aspect not only incorporates natural elements of fuzzyness in human reasoning (qualitative and imprecision classification) into EFOPREL but differs from other approaches to partition continuous domains [9, 36] in favor of the production of logic rules that are more understandable by humans.

EFOPREL's genetic search uses a fitness function that is closely related to a measure of classification accuracy. Moreover, EFOPREL's genetic search incorporate aspects like classification cost and attribute testing cost in a natural way. The relevance of incorporation of miss-classification costs and tests costs in a set of rules has recently been emphasised [45].

This paper is organized as follows. Section 2 reviews rule-based building of classifiers from supervised data sets and presents the first order logic language used by EFOPREL to induce rules. Here we explain how EFOPREL handles continuous and discrete attributes. Section 3 describes the use of genetic algorithms in EFOPREL. In particular, Subsection 3.1 describes the encoding of rules. Subsection 3.2 explains fitness evaluation. Genetic operators are described in Subsection 3.3 while the mechanism for simplifying rules is discussed in Subsection 3.4. Section 4 compares EFOPREL to previous systems for machine learning including those based on genetic algorithms. Section 5 illustrates the collaborative knowledge elicitation integrated in EFOPREL. We present some final remarks in Section 6.

2 Rule-based classification

A sound basis for decision making is fundamental for the outcome to have a practical predictive value. The capacity to extrapolate from known cases or past experience in order to anticipate possible outcomes with a high level of accuracy is regarded as valuable expertise. Researchers in Statistics and Artificial Intelligence have produced a pool of techniques for learning classification procedures from simple classified objects [33]. *Supervised learning* con-

$$
\begin{array}{lcl}
\langle \textsf{Logic\_Rule} \rangle & ::= & \text{IF } \langle \textsf{Antecedent} \rangle \text{ THEN } \langle \textsf{Consequent} \rangle \\
\langle \textsf{Consequent} \rangle & ::= & \langle \textsf{Class\_Label} \rangle \\
\langle \textsf{Antecedent} \rangle & ::= & \langle \textsf{Boolean\_Expression} \rangle \\
\langle \textsf{Boolean\_Expression} \rangle & ::= & \langle \textsf{Predicate} \rangle \\
 & | & \langle \textsf{Boolean\_Expression} \rangle \wedge \langle \textsf{Boolean\_Expression} \rangle \\
 & | & \langle \textsf{Boolean\_Expression} \rangle \vee \langle \textsf{Boolean\_Expression} \rangle \\
 & | & \neg \langle \textsf{Boolean\_Expression} \rangle
\end{array}
$$

Figure 1: The syntax of logic rules discovered by EFOPREL.

centrates in extracting decision criteria from samples of solved cases that would be applicable to new data.

In this section we focus our attention on this basic learning task of building classifiers; however, our interest is that the extracted knowledge be represented as decision-making rules amendable to human understanding. These decision-making rules specify a way to relate any particular pattern of observations (coded as attribute vectors) to one of the specified classes. We will refer to the potential relevant observations of a particular problem as attributes. An example will consist of a vector of values for observations and its associated class. The task of the learning element is to generalize the sample of empirical observations and identify essential patterns that are not overly specific to the sample data and that have a high predicting capacity on unseen cases.

Classification procedures based on some representation of logic rules are usually more accessible to human inspection than other methods [49]. EFOPREL induces logic rules as `if-then` statements (if the antecedent holds, the object should be labeled with the class label in the consequent). The antecedent is a logic expression constructed from nested combinations of atomic predicates using the logical connectives $\wedge$ (AND), $\vee$ (OR) and $\neg$ (NOT). The conjunction $\wedge$ and disjunction $\vee$ are always treated as binary while the negation $\neg$ is unary.

Figure 1 provides the grammar for the logic rules used by EFOPREL.

EFOPREL's predicates are constructed from the attributes describing examples and possible new cases in two fundamental ways.

1. If the values of attribute A_i range in a small discrete domain $D = \{d_1, \ldots d_t\}$ (for

example the attribute sex may take values from the set $\{male, female\}$), then the grammar of EFOPREL includes $A_i\_d_j$ as a predicate symbol (for $j = 1, \ldots, t$). For example, the predicate symbols SEX_MALE and SEX_FEMALE will be included for the attribute sex discussed above. The meaning of this predicate symbols is naturally that $A_i\_d_j(x)$ evaluates to true if and only if x is an object with value d_j on attribute A_i. For example, SEX_FEMALE($Marie\_Currie$) and SEX_MALE($Albert\_Einstein$) are true while SEX_FEMALE($Albert\_Einstein$) and SEX_MALE($Marie\_Currie$) are false.

2. When the possible values of attribute A_i constitute a continuous domain D (for example the attribute height) or even a large discrete domain (as for example the attribute age), the grammar of EFOPREL includes predicates of the form

$$\text{VERY\_}^*\text{LOW\_}A_i,$$

$$\text{NORMAL\_}A_i$$

and

$$\text{VERY\_}^*\text{HIGH\_}A_i$$

(where the regular expression a^* denotes 0, 1 or any finite number of repetitions of the symbol a). The meaning of these predicates will constitute a fuzzy partition of the domain D. The intended meaning is that a predicate like

$$\text{VERY\_VERY\_LOW\_HEIGHT}(x)$$

is true for an interval of values of the attribute height that are smaller than those values in the interval of the domain on which

$$\text{VERY\_LOW\_HEIGHT}(x)$$

is true. For every value $d \in D$ of the continuous attribute A_i there is always one predicate that evaluates to true. The predicates are linearly ordered. However, the boundary between two consecutive intervals of the continuous domain is nondeterministically defined (more will be said on this later).

As presented in this section, EFOPREL expresses concepts in a logic language that has the expressivity of first order logic formula with only unary predicates and allowing just one variable in a formula. This is equivalent to propositional calculus; however, this is the

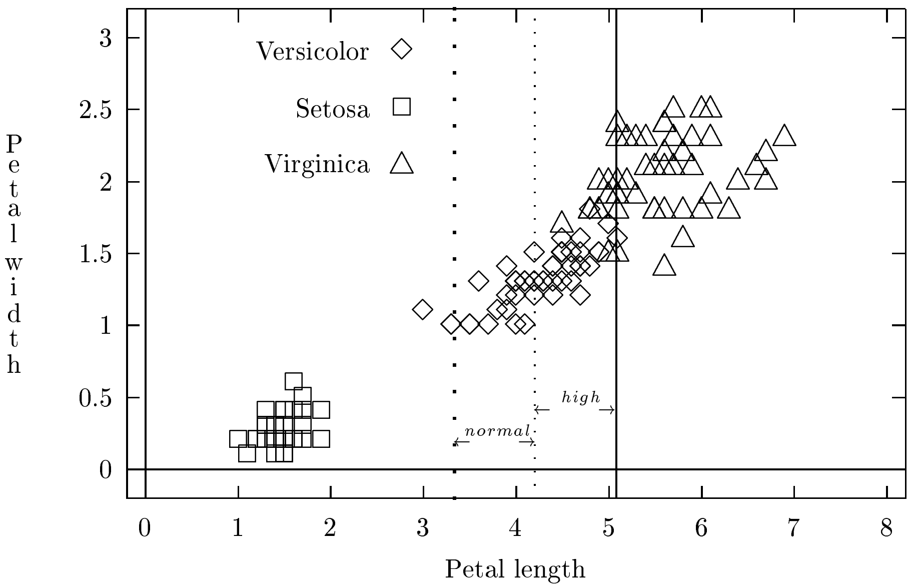

Figure 2: The Iris data set and a classification rule for versicolor.

common framework for most Machine Learning systems where examples are in attribute vector format. We will discuss extending the logic language of EFOPREL in later sections.

To illustrate the learning of rules with EFOPREL we will use the well-known Iris data set [26] originally used by Fisher [16] in the development of linear discriminants[1]. There are three classes of Iris: Setosa, Versicolor and Virginica. The length and width of both petal and sepal were measured on 50 flowers of each variety. Thus, the data set consists of 150 supervised attribute-vectors of four attributes. The original problem is to classify a new Iris flower into one of these three types on the basis of the four continuous features representing physical characteristics of the flowers. For example, EFOPREL v 1.1 has produced the rules

1. If HIGH_PETAL_LENGTH(x) *OR* NORMAL_PETAL_LENGTH(x) Then VERSICOLOR.

2. If HIGH_PETAL_WIDTH(x) *OR* NORMAL_PETAL_WIDTH(x) Then VERSICOLOR.

3. If NORMAL_PETAL_WIDTH(x) *OR* HIGH_PETAL_LENGTH(x) *OR* NORMAL_PETAL_LENGTH(x) Then VERSICOLOR.

[1]The data sets used here were retrived from the UCI Repository of Machine Learning Databases and Domain Theories managed by D.W. Aha at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

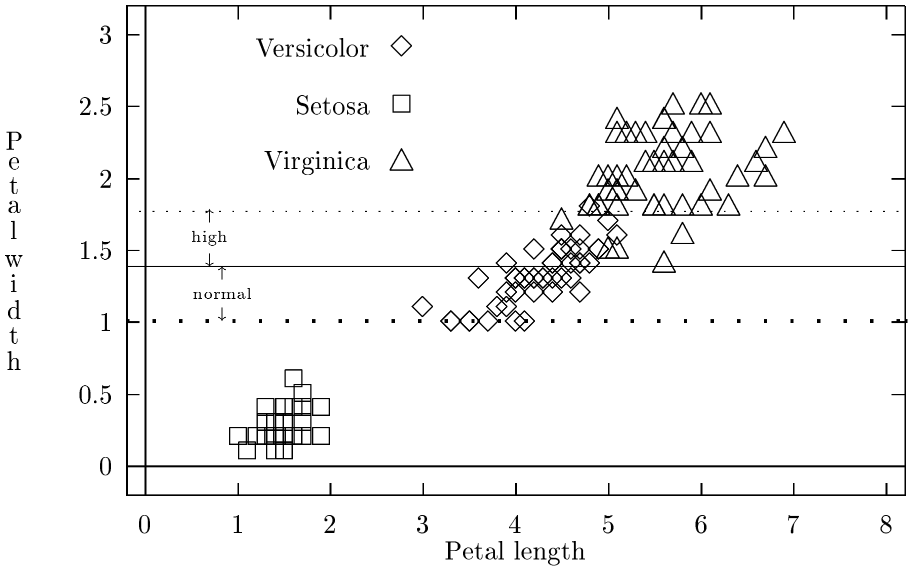

Figure 3: The Iris data set and an alternative classification rule for versicolor.

These rules can be visualized by plotting the data sets as functions of the two attributes: petal length and petal width. Figure 2 plots the data and the range for Rule 1 above. This rule corresponds to classifying as versicolor a flower with petal length $>$ 3.32cm and petal length $<$ 5.08cm. Figure 3 plots the data set and Rule 2 above, while Figure 4 plots the data set and Rule 3.

3 Genetic search

In EFOPREL, searching for good classifiers for a class is performed via a genetic algorithm because there are many rules for describing a class with many levels of accuracy.

A genetic algorithm is a probabilistic method inspired by an abstract model of our current understanding of the natural process of evolution. Because of this, the nomenclature imports and adjusts some terminology from biology and genetics. Through the process, an initial *population* of encoded individuals (or *chromosomes*) improves as parents are replaced by better offspring. Thus, three mechanisms are required to use a genetic algorithm. The first is the encoding to represent individuals for their manipulation by the computer program, the second is a mechanism of awarding merit to an individual. The third is a mechanism of

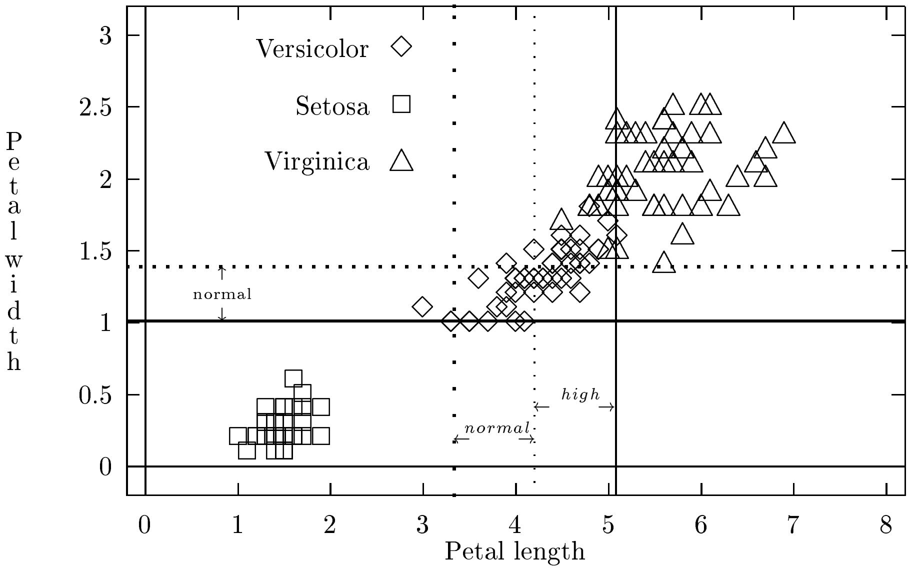

Figure 4: The Iris data set and a third alternative classification rule for versicolor.

deriving a new generation from the previous population. When using a genetic algorithm to maximize[2] a given function F, the value $F(c)$ is considered the *fitness* of the individual c. The encoding mechanism allows the representation of individuals (*phenotypes*) as chromosomes (*genotypes*).

For example, if we are maximizing the function $F_1(x) = 255^2 - x^2$ in the interval $[-255, 255]$, then the fitness for the value 6.5 is $F_1(6.5) = 64,982.75$. By representing integers in binary with a leading bit for the sign, the integers in the interval $[-255, 255]$ could be encoded as a binary string of length 9, such that $c = b_8 b_7 \ldots b_0$ (with $b_i \in \{1, 0\}$, for $i = 0, \ldots 8$) represents the integer $x = (-2b_8 + 1) \times \sum_{i=0}^{7} b_i 2^i$.

It is important to observe that the genetic algorithm will work in the space $\mathcal{S}$ of encodings (*chromosomes* or *genotypes*). In the example above, the space of encodings is the set $\mathcal{S}_1 = \{0,1\}^9$ of all binary strings of length 9 (but other encodings are possible, even decimal notation or floating point representation).

The algorithmic description of a *classical* or *traditional* genetic algorithm [11] in its generic format is presented in Figure 5. Later, we will elaborate on the details of crossover operators.

[2]In this paper we refer to maximizing a positive function, since minimizing F is equivalent to maximizing $-F$; and also maximizing $F'(x) = F(x)[F(x) > 0]$ (where $[F(x) > 0]$ is 1 if the predicate is true and 0 otherwise) is equivalent to maximizing $F(x)$.

GA $(\mathcal{S}, F, n, Prob_{cross}, Prob_{mutation}, stop)$;

Initialize population :

$generation \leftarrow 0$;

$P_0 \leftarrow \langle c_1, c_2, \ldots c_n \rangle$ where c_i is selected, independently and uniformly in $\mathcal{S}$.

Repeat

$Parents \leftarrow \langle c'_1, c'_2, \ldots, c'_n \rangle$ where c'_i is selected, independently and randomly from

$P_{generation} = \langle c_1, c_2, \ldots, c_n \rangle$ with probability distribution

$Prob[c'_i = c_{j_0}] = F(c_{j_0}) / \sum_{j=1}^{n} F(c_j)$.

For each letter α in each chromosome $c'_i \in Parents$

change α with probability $Prob_{mutation}$.

For $i := 1$ to $\lfloor n/2 \rfloor$

replace c'_{2i} and c'_{2i-1} with their crossover offspring

with probability $Prob_{cross}$.

$generation \leftarrow generation + 1$;

$P_{generation} \leftarrow Parents$

until *stop*

Figure 5: Algorithmic description of GA. The size of the population is n, the function F is the objective function and $Prob_{cross}$ is the probability of crossover while $Prob_{mutation}$ is the probability of mutation.

Selection is the process that extracts the set of *Parents* from which the new population is produced. Note that the probability distribution from which a parent is drawn is given by the ratios of fitness values of each chromosome to the total fitness value of the current population. Termination of execution is determined by a boolean criteria (*stop*); usually, when a large number of generations is reached or when all chromosomes are extremely alike.

3.1 The encoding of EFOPREL's rules

The encoding mechanism is fundamental in a genetic algorithm because it is the space of encodings (and not of individuals) a genetic algorithm is working with. In a sense, the encoding defines the view of the world for a genetic algorithm. For example, the binary encoding $c = b_8 b_7 \ldots b_0$ of the integer $x = (-2b_8 + 1) \times \sum_{i=0}^{7} b_i 2^i$ can not present fractions to the genetic algorithm or integers with magnitude larger that 255. Fixed length character strings have been found to present difficulties for some problems in artificial intelligence where the desired solution is hierarchical and where the size of the solution is unknown in advance [29, 28]. Thus, rather than using a mechanism to represent logic rules as fixed length strings [4, 12, 20], we take the approach of operating in structures that represent individuals by their parse tree [17, 29].

In EFOPREL, a genetic algorithm will explore the set of rules for a class by manipulating a population of encoded rules with a common consequent. Because the logic rules of EFOPREL are constructed from the grammar in Figure 1, a suitable representation for computer manipulation of a rule is the parse tree of the antecedent. For example, the rule

IF NORMAL_PETAL_WIDTH(x) OR HIGH_PETAL_WIDTH(x)
OR NORMAL_PETAL_LENGTH(x) THEN VERSICOLOR

may be represented as shown in Figure 6 (a).

In fact, EFOPREL v 1.1 represents antecedents as trees where the internal nodes are binary nodes with pointers to their children and a pointer to its parent (the parent pointer in the root is void). Internal nodes to represent $\neg$ (NOT) are unnecessary because, by De Morgan's Laws applied to logic expressions, we can place the necessary negations at the leaves (leaves encode literals). Binary nodes store whether they are a conjunction (using 1 for $\wedge$) or a disjunction (using 0 for $\vee$) in one bit. The rank of a node n is the number of nodes in the subtree rooted at n. The rank is recorded for efficient implementation of crossover

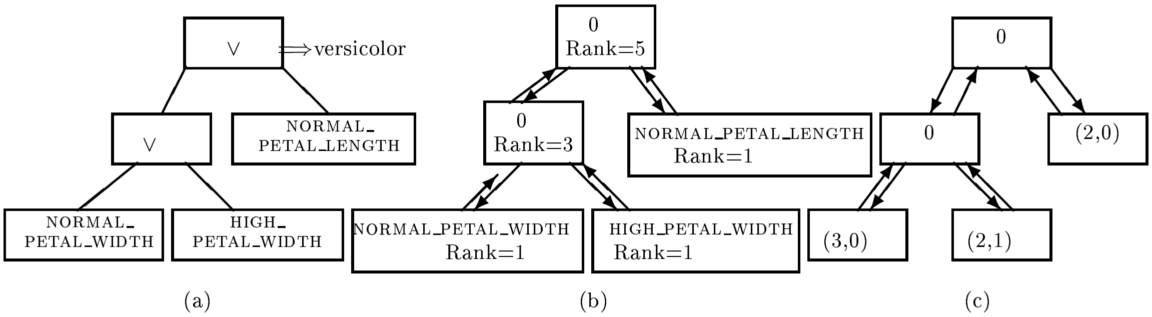

Figure 6: The process of encoding a rule into a tree for computer manipulation.

operators because the rank of n is one more than the number of links of the subtree rooted at n. Figure 6 (b) illustrates the encoding of internal nodes.

External nodes (the leaves of the parse tree) encode literals (predicates and their negations) as binary strings. For a discrete attribute A_i (categorical or ordered) with domain $D = \{d_1, \ldots, d_t\}$, we use $\lceil \log_2 t \rceil$ bits to encode the predicate $A_i\_d_j$ as the value of $j-1$ in binary. Thus, a leaf node for the predicate $A_i\_d_j$ stores the pair of integers $(i-1, j-1)$ as two binary strings.

For a continuous attribute A_i, the predicate VERY_^{j}LOW_A_i is represented as a pair of integers $(i-1, -(j+1))$ where each of the integers is coded in binary (note that in this case the second integer is always negative). The predicate NORMAL_A_i is represented as the pair of integers $(i-1, 0)$ and the predicate VERY_^{j}HIGH_A_i is represented by the pair $(i-1, j+1)$. In each pair, the two integer entries of the pair are encoded in binary. The user of EFOPREL defines a maximum value for j so that the number of bits sufficient for the encoding of j is known before a particular execution.

Evaluation of an antecedent on an object x is simple and can be defined recursively as follows. If the antecedent is a literal, then the boolean value of the literal is the value of the antecedent. If the node is a conjunction, its value is the logical AND of the values of its two subtrees. If the node is a disjunction, its value is the logical OR of the values of its two subtrees.

We now elaborate more on the evaluation of predicates for continuous attributes. In this case, the predicates correspond to a discretization of the domain of values dependent on two parameters $\rho \in (0,1]$ and $\tau \in [0, \rho/2)$ specified by the user of EFOPREL before the execution of a generic search. Let A_i be a continuous attribute and let μ_i and σ_i the observed mean and standard deviation in the values of attribute A_i among the sample (or training set). Then, the value x_{ik} of attribute A_i for the k-th example x_k is normalized by the transformation

$$z_{ik} \leftarrow \frac{x_{ik} - \mu_i}{\sigma_i}. \tag{1}$$

With this normalization, the variable z_{ik} has mean 0 and variance 1.

The domain of z is partitioned in intervals of $\rho\sigma_i$ units. The *deterministic interval predicate* for NORMAL_A_i is the question: $z \in [-\rho/2, \rho/2]$? Thus, NORMAL_A_i(x) is true if the i-th attribute of x, when normalized by transformation (1) results in a value in the interval $[-\rho/2, \rho/2]$. That is, the value of the i-th attribute of x_k is a normal value if it is not more than $\rho\sigma/2$ units from the observed mean in the examples.

Let us illustrate this with the Iris data set and the predicate NORMAL_PETAL_LENGTH(x). In the 150 examples, the observed mean of the attribute petal length is $\mu_{petal\_length} = 3.76$ and the standard deviation is $\sigma_{petal\_length} = 1.76$. When $\rho = 0.5$ the predicate NORMAL_PETAL_LENGTH(x) is asking if z_{petal_length} is in the interval $[-0.25, 0.25]$, where

$$z_{petal\_length} = \frac{x_{petal\_length} - \mu_{petal\_length}}{\sigma_{petal\_length}} = \frac{x_{petal\_length} - 3.76}{1.76}.$$

Now we write this in terms of a question for x_{petal_length}. Note that,

$$\frac{x_{petal\_length} - 3.76}{1.76} \in [-0.25, 0.25]$$

if and only if

$$-0.25(1.76) \leq x_{petal\_length} - 3.76 \leq 0.25(1.76),$$

if and only if $3.32 \leq x_{petal\_length} \leq 4.20$.

Similarly, the deterministic interval predicate for VERY_^{j}LOW_A_i is the question: $z \in [-\rho(j+1)-\rho/2, -\rho(j)-\rho/2]$? The deterministic interval predicate for VERY_^{j}HIGH_A_i is the question: $z \in [\rho(j)+\rho/2, \rho(j+1)+\rho/2]$? To illustrate this consider HIGH_PETAL_LENGTH(x). This corresponds to the case $j = 0$ and the predicate is asking the question

$$z_{petal\_length} \in [0.25, 0.75]$$

or equivalently $4.20 \leq x_{petal\_length} \leq 5.08$. Thus, the antecedent NORMAL_PETAL_LENGTH(x) *OR* HIGH_PETAL_LENGTH(x) is as illustrated in Figure 2 (a).

However, EFOPREL uses the parameter τ to add an amount of nondeterminism to the boundaries of the intervals in the discretization. If $z \in [a, b]$? is the question corresponding to an interval, then this question is really

$$z \geq a \text{ and } z \leq b \text{ ?}$$

EFOPREL replaces this question by

$$z \geq a + Y_l \text{ and } z \leq b + Y_h \text{ ?}$$

where Y_l and Y_h are independent random variables that are real valued and uniform in $[-\tau, \tau]$ and which are sampled every time the predicate is evaluated. Thus, near the boundary of the interval, the predicate may evaluate to true once and, when evaluated again on the same point near the boundary, it may evaluate to false. This makes the boundaries of continuous predicates distinct or blurred (as regulated by τ) and allows regulated tolerance for noise in data sets.

3.2 The fitness of a chromosome

The fitness of an encoded rule ϕ is a function of the error rate incurred by ϕ in classifying the sample data. The error rate for a rule is a measure of the agreement of the consequent of the rule and the class label in the inspection of each example. Thus, for each example in the training set, there are four possibilities; refer to Table 1.

The fitness of a rule is then evaluated as follows. For each correct evaluation of an example, (a match with the decision provided by the class label of the example) the rule accumulates positive points, while for each incorrect evaluation of an example, the rule receives negative points. The performance of the rule is a weighted average of the two types of positive points and of the two types of negative points.

Evaluation of a Rule in an Example			
Class (tutor)	*Antecedent*	*Explanation*	*Type of (mis)match*
`True`	`True`	Example classified correctly as member of the class	TT
`True`	`False`	Example classified incorrectly as outside the class	TF
`False`	`True`	Example classified incorrectly as member of the class	FT
`False`	`False`	Example classified correctly as outside the class	FF

Table 1: Agreement between a rule and the label of the class.

If the fitness function is a direct estimation of the error rate difficulties occur. For example, when we have a large training set but only a few examples of the class we are trying to find a rule for. In this case, the other examples are negative examples for the class. A rule that always answers FALSE will get a high performance just because it answers correctly on the large number of negative examples. This difficulty was observed in other inductive inference systems using genetic algorithms [4, 12].

EFOPREL solves this problem using weights that are in correlation with the proportion of positive examples in the training set. These weights are denoted ω_{TT}, ω_{TF}, ω_{FT} and ω_{FF}. Also these weights allow the user to adjust the search to take into account misclassification costs. The formula for evaluating the fitness F of a rule ϕ in a training set TR is

$$\begin{aligned} F(\phi) = \quad & \gamma^* \times & \{ & \omega_{TT} \times (\sharp \text{ of TT matches}) \\ & & & -\omega_{TF} \times (\sharp \text{ of TF mismatches}) \\ & & & -\omega_{FT} \times (\sharp \text{ of FT mismatches}) \} \\ & +\gamma^- \times & \{ & \omega_{FF} \times (\sharp \text{ of FF matches}) \} \\ & & & -\omega_{OR} \times Rank(\phi), \end{aligned}$$

where,

$$\gamma^* = \frac{\sharp \text{ of examples in } TR}{\sharp \text{ of positive examples in } TR},$$

$$\gamma^- = \frac{\sharp \text{ of examples in } TR}{\sharp \text{ of negative examples in } TR}$$

and ω_{OR} is a weight that benefits shorter rules.

3.3 The genetic operators

We now present the details of the genetic operations. The conclusion of each rule is transmitted to the offspring untouched, both during crossing and during mutation.

The crossing step selects probabilistically subtrees (subexpressions) from the parents and just exchanges these subtrees. The subtrees may not be proper subtrees. In particular, two parental logic expressions encoding two antecedents result in two descendants using parts of each parent. The crossover generates offsprings that are syntactically and semantically valid because entire subexpressions are swapped [18, 28]. Figure 7 illustrates this with an example. The logic expression for an antecedent is encoded as its parse tree and considered as a rooted trees with ordered branches. The $k-1$ links of a tree T (with rank k at the

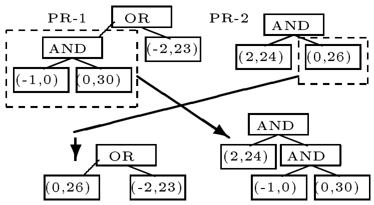

Figure 7: Crossing between rules.

root) are numbered in depth-first order. A cuting link is randomly and uniformly selected in each parent tree. The point of cut in the tree T is recursively determined as follows. A uniform random choice c in $\{1, \ldots, k\}$ is made. If $c = k$, then the expression rooted at T will be swapped. If $c < k - 1 - rank(\text{right\_child})$, the cut is the c-th cut point of T's left child. Otherwise, the cut is the $[c - rank(\text{left\_child})]$-th cut point of the right child.

The mutation operator is just a traversal of the tree (the genotype). With very low probability the components of the tree are modified. Every bit may be negated. Thus, a node with an $\vee$ (OR) can reverse to an $\wedge$ (AND). At the leaves, instead of evaluating the fifth attribute we may now evaluate the 21-st (because the fifth bit is reversed). The predicate VERY_HIGH (which is encoded as 2) may change to VERY_VERY_HIGH (encoded as 3) or VERY_LOW (encoded as -2), also by the reversal of only one bit. Mutation applied to a link may replace the root node of that link by one of the subexpressions.

3.4 Simplification of rules

The simplification of rules in EFOPREL is another aspect that deserves clarification. Simplification allows EFOPREL to obtain shorter and more understandable equivalent expression for a hypothesis consistent with the data. Simplification is required because the same logical concept has equivalent encodings. Simplification of constant expressions was addressed in BEAGLE [18] and its need was identified elsewhere [28]. EFOPREL addresses simplification effectively. For example, without simplification EFOPREL produced the following rule for the Iris data.

IF (NORMAL_PETAL_LENGTH(x) *OR* HIGH_PETAL_LENGTH(x))
OR ((LOW_SEPAL_LENGTH(x) *AND* VERY_HIGH_SEPAL_LENGTH(x))
OR NORMAL_PETAL_LENGTH(x))

THEN VERSICOLOR.

However, the subexpression LOW_SEPAL_LENGTH(x) AND VERY_HIGH_SEPAL_LENGTH(x) is always *false* because, for all x, the petal length can not have a very high value and a low value (these predicates correspond to disjoint intervals in the partition of the domain of this attribute). Thus, we have a subexpression *false* OR NORMAL_PETAL_LENGTH(x). This is equivalent to simply NORMAL_PETAL_LENGTH(x). Finally using the fact that $\phi \vee \psi \vee \phi$ is logically equivalent to $\phi \vee \psi$ the apparently complex rule is simply

IF NORMAL_PETAL_LENGTH(x) OR HIGH_PETAL_LENGTH(x)
THEN VERSICOLOR.

EFOPREL solves simplification because EFOPREL's antecedent can be simplified by the criteria of minimizing the number of literals. Thus, simplification is performed by standard methods used in logic circuit design (for example, *Incremental Consensus*[30]). However, we do believe that simplification is responsible for EFOPREL computational efficiency. Other approaches to learning based on genetic algorithms, but utilizing a parameterized representation in the form of parse trees have faced problems of exploring redundant and/or invalid solutions. These problems have recently received attention [25, 34] in an attempt to provide grammatical (syntactic or semantic) means to avoid such problems while assuring suitable computational complexity. EFOPREL's approach frees the user of specifying the target language with such detail (a task that may require considerable insight into the problem and good judgement). EFOPREL automatically simplifies chromosomes repeatedly after a few generations (typicly after four or five). In this way, EFOPREL achieves three goals. First, trees may have redundant branches (that serve as memory for the disruptive effects of genetic operators). Second, the search keeps a regulated focus on canonical genotypes. Third, the process to assure the first two objectives is computationally efficient.

4 A comparison

In this section we compare EFOPREL with previous approaches for inducing logic rules from a training set of classified examples.

Rule	Correct: In class & antecedent true	Correct: ¬ in class & antecedent false	Errors: ¬ in class & antecedent true	Errors: In class & antecedent false	Accuracy
IF ZONE_SW(x) THEN LANGUAGE_SPANISH(x)	8	165	8	13	89.1%
IF RELIGION_CATHOLIC(x) THEN LANGUAGE_SPANISH(x)	20	153	20	1	89.1%
IF RELIGION_CATHOLIC(x) AND BLUE_PRESENT(x) THEN LANGUAGE_SPANISH(x)	16	163	10	5	92.2%
IF RELIGION_CATHOLIC(x) AND BLACK_ABSENT(x) THEN LANGUAGE_SPANISH(x)	18	160	13	3	91.7%
IF RELIGION_CATHOLIC(x) AND LOW_NUMBER_OF_BARS(x) THEN LANGUAGE_SPANISH(x)	17	163	10	4	92.7%
IF RELIGION_CATHOLIC(x) AND NORMAL_POPULATION(x) THEN LANGUAGE_SPANISH(x)	19	156	17	2	90.2%
IF RELIGION_CATHOLIC(x) AND NORMAL_POPULATION(x) AND BLUE_PRESENT(x) THEN LANGUAGE_SPANISH(x)	16	166	7	5	93.8%
IF RELIGION_CATHOLIC(x) AND NORMAL_POPULATION(x) AND[BLACK_ABSENT(x) OR NORMAL_NUMBER_OF_CIRCLES(x)] THEN LANGUAGE_SPANISH(x)	17	165	8	4	93.8%

Table 2: Logic rules inferred by EFOPREL with population size 50 on 100 generations. Probability for crossover was set to 0.6 and for mutation the probability was set to 0.1. The weights of the fitness function were $\omega_{TT} = 5,000$, $\omega_{FF} = 4$, $\omega_{FT} = 5,000$, $\omega_{TF} = 5,000$ and $\omega_{OR} = 50$. Rules are shown in order of complexity.

4.1 Power of elicitation

EFOPREL is similar to BEAGLE [17]. Thus, our first comparison uses the data set of flags[3] used to illustrate BEAGLE [18]. The data set has the encoding of general and flag information for 194 countries. The target concept is those countries where the official language is Spanish. Three executions of EFOPREL obtained the rules in Table 2.

The attribute zone is the geographic quadrant, based on the Greenwich meridian (East/West) and the Equator (North/South) and has values $\{NE, SE, SW, NW\}$. Thus, the first rule says that if a country is in the south-west quadrant then its official language is Spanish. Other rules are based on the official (main) religion of the country and predicates to test if particular colors are present in the national flag. Perhaps further clarification is required for NORMAL_NUMBER_OF_CIRCLES which results of treating the number of circles in the flag as a continuous attribute. One first important aspect of EFOPREL's output is the large number of alternative rules produced that suggest attributes that are relevant for the classification. BEAGLE produces 4 rules (refer to Table 4). The second rule in BEAGLE is obtained by making the attributes population, religion and white unavailable for classification; thus alternative rules based on other features are generated. Elimination of the attributes population, religion and white in the training set and using EFOPREL resulted in rules shown in Table 3.

We find these results illustrative of the capabilities of EFOPREL in two aspects. We focus on the human understanding of rules first; that is the cognitive aspect. For example, consider the most accurate BEAGLE rule. What does it mean to verify that the population (in millions) is larger than the boolean value (0 or 1) of the presence of white in the flag and to subtract the encoding for religion? Note that the rule depends heavily on the encoding. Perhaps we could infer that `religion` is relevant for whether a country is Spanish speaking. However, this fact is most explicit in the set of EFOPREL's rules. With EFOPREL we clearly confirm that Catholic religion is relevant and highly indicative of Spanish speaking countries. Descriptive interpretation of BEAGLE's third rule is a challenge. Thus, although BEAGLE's rules are interesting relations between attributes with correlation to the goal concept, EFOPREL's rules allow better and easier human understanding and manipulation, with also a closer semantics to human usage.

The second aspect is classification accuracy. All of the rules generated by EFOPREL

[3]The data sets "Breast Cancer", "Flags", "Iris Plant" and "Trains" used here were retrived from the UCI Repository of Machine Learning Databases and Domain Theories managed by D.W. Aha at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Rule	Correct		Errors		Accuracy
	In class & antecedent true	¬ in class & antecedent false	¬ in class & antecedent true	In class & antecedent false	
IF ZONE_NE(x) THEN LANGUAGE_SPANISH(x)	3	169	4	18	88.6%
IF ZONE_NE(x) AND NORMAL_AREA(x) THEN LANGUAGE_SPANISH(x)	2	173	0	19	90.2%
IF ZONE_NE(x) AND GREEN_ABSENT(x) THEN LANGUAGE_SPANISH(x)	3	169	4	18	88.6&
IF LAND_MASS_SOUTH_AMERICA(x) OR[(ZONE_NW(x) AND TOP_LEFT_COLOR_BLUE(x)] THEN LANGUAGE_SPANISH(x)	8	165	8	13	89.1%

Table 3: Logic rules inferred by EFOPREL with population size 50 on 100 generations (some attributes have been eliminated from the training set). Rules are shown in order of complexity.

(Tables 2 and 3) are more accurate than the second best rule by BEAGLE and also some rules are more accurate that BEAGLE's most accurate rule.

A second comparison involves the well-known natural database for diagnosing recurrence of breast cancer[4] from Ljubljana Research [5] used in the discussion of AQ15 [32]. This data set has been used for evaluating the performance of several inductive learning algorithms [10, 44, 7], but in particular, for two inductive learners using genetic algorithms [4, 14]. The data set has 286 descriptions of cases and two classes. A case is defined by 9 attributes, a small number of examples have some attributes missing and it is known that a small amount of noise of unknown origin is present (some of the noise is manifested as cases with identical features but different classification).

EFOPREL induced rules from a random subset of 200 cases, and the accuracy of the inferred rule was tested on the unseen cases. The system was run on 10 different random training sets with 200 cases (and in many cases, it converged to the same rule). Some of the rules are presented in Table 5. Rules in Table 5 were generated with a population of 50 rules

[4]The data sets used here were retrived from the UCI Repository of Machine Learning Databases and Domain Theories managed by D.W. Aha at `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Rule	Correct: In class & antecedent true	Correct: ¬ in class & antecedent false	Errors: ¬ in class & antecedent true	Errors: In class & antecedent false	Accuracy
IF $((white \leq population) - religion)(x)$ THEN LANGUAGE_SPANISH(x)	20	160	13	1	92.7%
IF $(landmass \leq 2.5)(x)$ THEN LANGUAGE_SPANISH(x)	20	145	28	1	85.0%
IF $((zone \leq 2.6) < (((zone \geq 2.0) \geq gold) < area))(x)$ THEN LANGUAGE_SPANISH(x)	21	141	32	0	83.5%
IF $((quarters + stripes) > 2.0)(x)$ THEN LANGUAGE_SPANISH(x)	17	124	49	4	72.6%

Table 4: Logic rules inferred by BEAGLE with 35 generations. Rules are shown in order of accuracy.

and 100 generations. Exactly half of the previous population is replaced by new offspring and "windowing" [11] was used to adjust fitness values. Probability for crossover was set to 0.6 and for mutation the probability was set to 0.1. The weights of the fitness function were $\omega_{TT} = 5,000$, $\omega_{FF} = 4$, $\omega_{FT} = 5,000$, $\omega_{TF} = 5,000$ and $\omega_{OR} = 50$.

Again, we find the results illustrative of the power of EFOPREL for inductive inference in several aspects. On the aspect of accuracy, EFOPREL shows performance at the levels reported in the literature with systems based on decision tress (the ID3 family of systems), decision sets (the AQ family of systems), decision lists (the CN2 family of systems), Bayes classification, neural networks and other geneticly based systems [5, 32, 10, 44, 7, 4, 14]. EFOPREL's rules are able to classify accurately new examples, even in the presence of noise in the data set.

On the aspect of simplicity, the quest for rules as short as possible has been emphasised many times [5, 10, 49]. This is mainly for two reasons: shorter rules usually do not overfit the data; however, also shorter rules are easier to communicate, validate or apply by human experts.

EFOPREL's rules are so succinct that they are presented in Table 5. We refer to Clark and Niblett's approach [10] to evaluate the complexity of the knowledge structure induced by different algorithms (although other approaches are very similar [5, 32]). The complexity of a decision tree is the number of nodes (including leaves) in the tree. For the CN2 and AQ

Rule	Accuracy Train	Accuracy Test
IF DEGREE_MALIG_3(x) THEN RECURRENCE(x)	73%	70%
IF DEGREE_MALIG_3(x) OR INV_NODES_9TO11(x) THEN RECURRENCE(x)	73%	70%
IF DEGREE_MALIG_3(x) OR INV_NODES_3TO5(x) THEN RECURRENCE(x)	71%	71%
IF DEGREE_MALIG_3(x) OR INV_NODES_3TO5(x) OR INV_NODES_9TO11(x) THEN RECURRENCE(x)	71%	71%
IF DEGREE_MALIG_3(x) OR [INV_NODES_3TO5(x) AND INV_NODES_9TO11(x)] THEN RECURRENCE(x)	71%	71%
IF [DEGREE_MALIG_3(x) AND TUMOR_SIZE_30TO34(x)] OR [DEGREE_MALIG_3(x) AND NODE_CAPS_YES(x)] THEN RECURRENCE(x)	80%	71%
IF DEGREE_MALIG_3(x) AND [TUMOR_SIZE_30TO34(x) OR DEGREE_MALIG_2(x) OR NODE_CAPS_YES(x)] THEN RECURRENCE(x)	80%	71%

Table 5: Logic rules inferred by EFOPREL with population size 50 on 100 generations. Rules are shown in order of complexity.

family, the complexity is the number of selectors in the final rule list and rule set respectively.

A parallel measure of complexity for EFOPREL's rules are the number of nodes in the tree encoding the rule. Thus, the rules in Table 5 have complexity 1, 3, 3, 5, 5, 7 and 7, respectively. On the same data set, the representative of decision trees had complexities of 122 (for no pruning) and 44 (for pruning), while AQR had complexity 208 [10]. CN2 had complexities 28 at 90% threshold, 20 at 95% threshold and 4 at 99% threshold [10].

4.2 Efficiency

On the issue of efficiency we report the following. EFOPREL is coded in C. Three executions with the breast data (200 cases) with population size 50 and for 100 generations required 16 seconds, 11 seconds and 14 seconds, respectively on Pital (a PC-486 running LINUX in multiuser mode and X11). Newbool [4] requires a population of 1000 individuals for this data set and at least 20 minutes on a Sparcstation. For the trains [43] data set (10 cases), EFOPREL's 16 independent runs on Pital resulted on an average running time of 1.40 seconds per run (with a 99% confidence interval of ± 0.28), the longest running time was 2.80 seconds and the shortest 0.90 seconds. EFOPREL achieved 100% classification accuracy with 80 generations in each of the 16 runs. While for this data set, the parallel implementation of GA-SMART [20] used a multiprocessor of 16 transputers, 12 minutes and 170 generations to achieve equivalent results. We were not forced to limit generations or population size as in the case of GABIL [14]. Thus, with respect to other geneticly based inference tools, we are confident that EFOPREL is more efficient when inducing logic rules.

With respect to other paradigms for inductive inference we have used CN2 [10] as a benchmark. Table 6 provides run-time comparisons (in seconds) between CN2 [5] and EFOPREL for three computer platforms (we were unable to port CN2 to IRIX 6.0). For this direct comparison the training set was the full data-set of breast-cancer data. The results are averages over 10 runs with 95% confidence intervals. Moreover, EFOPREL is highly parallelizable and we are also currently developing a parallel C-version for the Silicon Graphics Power Challenge L. Thus, EFOPREL serial and parallel versions should scale up easily to more complex and larger data sets.

[5] Version distributed on `http://www.cs.utexas.edu/users/pclark/software.html#cn2`.

	Toshiba 486 LINUX	SUN IPXstation SUN/OS 5.4	SG Power Challenge L IRIX 6.0
CN2	7.03s ± 0.02	3.01s ± 0.01	
EFOPREL	14.87s ± 1.44	6.7s ± 1.09	4.68s± 0.52

Table 6: Running time comparisons between CN2 and EFOPREL.

4.3 The target language

In our description of far, EFOPREL expressivity may seem to be framed in propositional calculus. However, there are many situations where this can be conveniently and efficiently extended. For example, consider a binary predicate $P(x, y)$ where $x \in D_1$ and $y \in D_2$. We can construct a unary predicate P' for the target language of EFOPREL with the semantics that $P(x, y)$ if and only if $P'(z)$ with $z \in D_1 \times D_2$ and $z = (x, y)$. A similar approach can be used for predicates of larger arity. Another possibility occurs when one of D_1 or D_2 is finite. For example, if $D_1 = \{v_1, \ldots, v_n\}$ (thta is, $\|D_1\| = n$), we can represent the predicate P in the target language of EFOPREL with n predicates P_v_1, P_v_2, …, P_v_n. This last approach can be extended by using a template formula that serves as a dictionary of terms and using fixed length binary strings to encode which atomic and non-atomic formula compose the individual [20]. However, this extension introduces a large number of redundancies. Moreover, the size of the search space grows exponentially large with the size of the template with a serious impact on performance.

An illustration that binary predicates or relations involving quantifiers can be rephrased to a target language managed with EFOPREL is presented here with the trains data set [43]. The goal for this data set is to discriminate those trains going east from those going West. In order to present the terms of this data set we give here a brief description of three rules derived by EFOPREL (the first classifies exactly one instance incorrectly while the last two give 100% accuracy). The meaning of the first rule is a train going West does not contain a 4th car or if it does, then such car has no load. The second rule says a train going East has a triangle-shaped load next to a triangle-shaped load or next to a rectangle-shaped load. Finally, the third rule is a generalization of the first, which covers the case that a train has a 4th car with a load. In such case, the train going West must have a circle-shaped load in its fifth car and no triangle-shaped load next to a rectangle-shaped load.

IF NOT TRAIN_CONTAINS(t,CAR_4)
OR
NOT [∃ y:: CONTAINS_LOAD (CAR_4,y)]
THEN GOING_WEST

IF LOAD_SHAPE(x,TRIANGLE) AND LOAD_SHAPE(y,TRIANGLE) AND NOT NEST_TO(x,y)
AND
LOAD_SHAPE(z,TRIANGLE) AND LOAD_SHAPE(w,RECTANGLE) AND NOT NEST_TO(z,w)
THEN GOING_WEST

IF NOT TRAIN_CONTAINS(t,CAR_4) OR NOT [∃ y:: CONTAINS_LOAD (CAR_4,y)]
OR
[∃ x:: (CONTAINS_LOAD(CAR_5,x) AND LOAD_SHAPE(x,CIRCLE) AND
NOT (∃ z,w:: NEXT_TO(z,w) AND LOAD_SHAPE(z,TRIANGLE)
AND LOAD_SHAPE(w,RECTANGLE))]
THEN GOING_WEST

4.4 Other aspects

Other aspects that also favor EFOPREL are its capacity to handle continuous attributes as well as discrete attributes. It can work in batch or incremental mode. In batch mode, the entire training set is presented initially. Incremental mode allows the user to specify a size parameter to partition the training set in groups and incorporate groups of cases, one group at a time. A group is incorporated when classification accuracy on the cases from previous groups reaches a user defined threshold (or stabilizes). EFOPREL can start with a random population or randomly select a positive example to construct a rule and seed the initial population with rules constructed so far. As already mentioned, EFOPREL handles uneven proportions of representability of classes by examples. Thus, EFOPREL is a flexible tool for discoverying regularities as logic rules in supervised data sets.

EFOPREL does not require a domain theory to help its search (as in the case of SIA01 [1]) and does not limit the form of rules (as also is the case of SIA01 [1]). It allows flexible manipulation of the rules. The user may remove conditions, and restrictions or modify individuals in a population and explore their evolution further with EFOPREL. This means that EFOPREL does not require special genetic operators for adding conditions [1, 14] or making rules more general or more specific. EFOPREL provides two mechanisms to achieve the objectives these types of operators are usually put in place for. The first mechanism is implicit and consists of adjusting the weights of the fitness evaluation. Thus, if the user observes that populations are converging to rules that accurately classifying members of a class, but an undesirable number of negative cases are also classified as in the class, it may be desirable to specialize the antecedent, or equivalently adjust the weights. The other mechanism is explicit and involves defining a logic expression that will be constantly joined (with an AND for specialization or with an OR for generalization) to the individuals of every population.

Note that EFOPREL's search at this level is guided by the human user, rather than to be fully automated with genetic operators whose role must be adjusted adaptively [14]. This leads us to the following section.

5 The spirit of EFOPREL

The applicability of Machine Learning tools to the field of expert systems was raised explicitly by B.G. Buchanan [6]. Since then, researchers have found evidence suggesting contributions of symbolic learners to knowledge acquisition [2] as well as evidence that techniques in knowledge acquisition surpass inductive inference mechanisms of the ID-3 family [21]. EFOPREL's spirit follows the approach that knowledge elicitation and machine learning are different but complementary [47]. They are both useful to develop a knowledge base. EFOPREL follows the Machine-Expert Collaborative Knowledge Acquisition approach [47]. However, as opposed to the Knowledge Factory [48], EFOPREL spirit retains some roles for the knowledge engineer. EFOPREL intends to facilitate machine learning for knowledge acquisition in a flexible and collaborative effort. Thus, control is on the human user. Because EFOPREL is based on genetic search, it allows deletion of rules, modification of weights and production of alternative rules, as well as specialization and generalization of rules.

In order to facilitate communication between the user and the geneticly-based inductive learner, EFOPREL preserves the target language of the rules (as in MECK [47] and the Knowledge Factory [48]). EFOPREL offers a GUI interface that provides summarized information and meta-information about the accuracy of the rules, the types of miss-classification and how individual examples are classified. EFOPREL also provides access to the parameters of the genetic search for tunning or experimenting other potential search paths.

In what follows we illustrate this aspects of EFOPREL with the data set of flags. The first aspect is a facility for the user to construct, edit and test rules in the target language. The user can construct rules interactively. For illustration consider the construction of the first rule of Table 2. The user will only be required to indicate that is constructing a predicate for attribute `zone` (EFOPREL displays attribute names for the data set). After this, EFOPREL presents the possible options (in this case `NW`, `SE`, `SW` and `NE`). The user will select `SW` and the system will automaticly build a rule with this predicate as the antecedent:

$$\text{IF } \textsc{zone\_SW}(x) \text{ THEN } \textsc{Language\_Spanish}(x). \tag{2}$$

Similarly, to build the second rule in Table 2, the user indicates the attribute `religion`. EFOPREL will offer the alternatives (as defined by the description of the data set); namely, `Catholic`, `Other_Christian`, `Muslim`, `Buddhist`, `Hindu`, `Ethnic`, `Marxist`, `Others`. Selection of the first option builds the rule:

$$\text{IF } \textsc{religion\_Catholic}(x) \text{ THEN } \textsc{Language\_Spanish}(x). \tag{3}$$

Thus, the user can input knowledge in the form of rules. The rules are displayed by EFOPREL using several alternative formats. Besides their logic form shown earlier, EFOPREL can display the parse tree as well as a list of terms resulting from post-order traversal of the parse tree. This later format is useful to see how rules are evaluated non-recursively using a stack.

The rules constructed by the user can be inserted into an initial population of the genetic algorithm for developing variants of them. Moreover, the rules of an initial population of a second search may be resulting rules from a first search slightly edited or modified by the user.

For example, as mentioned earlier, the two rules above are the result of a genetic search on the flags data set. EFOPREL's meta-information of the classification produces output that is very similar to the first two lines of Table 2. Indication of the accuracy and types of miss-classification is given. The user may notice that while the first rule has 13 false negatives the second rule has only 1. This may suggest that a combination of the antecedent of the rules may lead to a more accurate rule. After building the rules (alternatively, from the output of a first genetic search), the user may indicate two rules to be combined by an AND. Thus, combining Rule (2) and Rule (3) above, combines the antecedents and results in the following rule:

$$\text{IF } \textsc{zone\_SW}(x) \text{ AND } \textsc{religion\_Catholic}(x) \text{ THEN } \textsc{Language\_Spanish}(x). \tag{4}$$

Rule	Correct		Errors		Accuracy
	In class & antecedent true	¬ in class & antecedent false	¬ in class & antecedent true	In class & antecedent false	
IF ZONE_SW(x) AND RELIGION_CATHOLIC(x) THEN LANGUAGE_SPANISH(x)	8	171	2	13	92%
IF ZONE_SW(x) OR RELIGION_CATHOLIC(x) THEN LANGUAGE_SPANISH(x)	20	147	26	1	86%

Table 7: Logic rules resulting from specialization and generalization of Rule (2) and Rule (3), respectively.

This is a user controlled specialization since the antecedent is more restrictive. Alternatively, the user may want to generalize the rules when combining them with a disjunction:

$$\text{IF zone\_SW}(x) \text{ OR religion\_Catholic}(x) \text{ THEN Language\_Spanish}(x). \quad (5)$$

The user may test the accuracy of this new rules, and EFOPREL offers a menu option for this. The results would be as in Table 7. The user may be satisfied with Rule (4), since this specialization has improved the accuracy to 92%. Note that Rule (4) is the collaborative effort of the genetic search in the data set and the user's suggestion to combine two rules. This is a collaborative knowledge acquisition effort. EFOPREL offers other interesting capabilities for enhancing this approach.

We now illustrate EFOPREL's capacity to report to the user what are the examples that are miss-classifyed. For Rule (4), this facility reports that the two countries that do not speak Spanish but are classified as so are Brazil and French Polynesia, while the 13 countries that do speak Spanish but are ruled out by Rule (4) are Colombia, Costa-Rica, Cuba, Dominican Republic, El Salvador, Guatemala, Honduras, Mexico, Nicaragua, Panama, Puerto-Rico, Spain and Venezuela. This information may prompt the user's domain knowledge to edit and refine the rule; for example, most of the Spanish speaking countries miss-classifyed are in Central America. However, the attribute `landmass` has only the values `N_America`, `S_America`, `Europe`, `Asia` and `Oceania`. One alternative is to extend the granularity of this attribute and relabel the attribute vector of each example accordingly. This illustrates that EFOPREL's spirit can lead to refinements in the language for knowledge representation. Never the less, EFOPREL offers also another alternative based on selective genetic search

Rule	Correct		Errors		Accuracy
	In class & antecedent true	¬ in class & antecedent false	¬ in class & antecedent true	In class & antecedent false	
IF RELIGION_CATHOLIC(x) AND [LANDMASS_S_AMERICA(x) OR LANDMASS_N_AMERICA(x)] THEN LANGUAGE_SPANISH(x)	19	170	3 Brazil, Haiti & French-Guyana	2 Cuba & Spain	97.4%

Table 8: Logic rule derived by genetic search from user suggested specialization of Rule (3).

for rule refinement.

The results of Table 7 suggest that specialization was a reasonable approach. Moreover, Rule (3) makes only one false negative error and classifies all Central American countries (which are officially Catholic) as Spanish speaking. This fact can be verifyed interactively by querying EFOPREL for the correctly and incorrectly classified examples for Rule (3). Thus, specialization of Rule (3) should be explored. EFOPREL allows this by allowing the user to specify a rule with an undetermined term P to be explored by genetic search. For example, the user may construct the following second order expression.

IF RELIGION_CATHOLIC(x) AND
P
THEN LANGUAGE_SPANISH(x).

After this, the user may ask EFOPREL to infer a logic term to be placed where the undetermined term P is. EFOPREL's genetic search engine has been designed to handle this. Only a population of logic terms is manipulated and when it comes to evaluation of their fitness, they are plugged into the place indicated by P. The genetic engine does not manipulate the determined part of the rule. In our example, EFOPREL's most dominant result for P above is

LANDMASS_S_AMERICA(x) OR LANDMASS_N_AMERICA(x).

The user can display the new formed rule and its classification properties as in Table 8. Note EFOPREL's ability to find what we suspected from our domain knowledge.

Rule	Correct		Errors		Accuracy
	In class & antecedent true	¬ in class & antecedent false	¬ in class & antecedent true	In class & antecedent false	
IF RELIGION_CATHOLIC(x) AND (ZONE_SW(x) OR LANDMASS_N_AMERICA(x) OR [NORMAL_AREA(x) AND HIGH_NUM_HORZ_STRIPES(x)]) THEN LANGUAGE_SPANISH(x)	19	170	3	2	97.4%
IF RELIGION_CATHOLIC(x) AND (ZONE_SW(x) OR LANDMASS_N_AMERICA(x) OR [LANDMASS_S_AMERICA(x) AND ZONE_NW(x)]) THEN LANGUAGE_SPANISH(x)	19	169	4	2	96.9%
IF RELIGION_CATHOLIC(x) AND (ZONE_SW(x) OR LANDMASS_N_AMERICA(x) OR LANDMASS_S_AMERICA(x) OR [NORMAL_AREA(x) AND ZONE_NW(x)]) THEN LANGUAGE_SPANISH(x)	20	169	4	1	97.4%

Table 9: Logic rule derived by genetic search from user suggested specialization of Rule (3).

Alternatively, we may want to use EFOPREL to find a balance between the previous specialization and generalization performed with Rules (2) and (3). Favoring specialization of Rule (3) the user can build the following expression.

IF RELIGION_CATHOLIC(x) AND
(ZONE_SW(x) OR P)
THEN LANGUAGE_SPANISH(x).

After this, EFOPREL is used to infer a logic term to be placed where the undetermined term P is. Results of this inference are shown in Table 9. In general, one or more undetermined terms can be placed in a second order formula.

This facility for collaborative knowledge acquisition is due to EFOPREL's use of genetic search. It offers an implicit second order dialogue between the user and Machine Learning, because P is ranging over the possible formula (that is, P has a value in $\langle$Boolean_Expression$\rangle$s, as defined in Figure 1).

The collaborative effort provides guidance to both parties. EFOPREL's inference suggest relevant attributes and initial rules. The user suggests patterns for new rules. In a sense, the user is shaping the fitness function adaptively for the benefit of the genetic-search. Most importantly, the user can make explicit its advise to the artificial learner and be naturally involved in directing its inference mechanism.

6 Discussion and conclusion

The knowledge-acquisition and knowledge-base construction phases in the development of expert systems are regarded as time consuming and expensive [23]. The machine learning community has identified this as the "Feigenbaum bottleneck" [5] and the development of EFOPREL has been motivated for this quest for methods for automatic knowledge acquisition. Methods based on inductive learning regard the data set as it main (if not its only) input; and thus, the data set is the source of knowledge. We have shown that EFOPREL offers an effective and efficient system for inductive learning.

Simultaneously, there has been a development of architectures and systems where the principal source of knowledge is the human expert and the computer tools are support tools to the process of knowledge elicitation. The most well-known example is KSS0 [40, 19]. Although the inductive learning algorithm ID3 is part of the tools of KSS0, the human experts are the main source of "knowledge", and the elicitation of knowledge has no influence on the behavior of the artificial learner. That is, EFOPREL offers a closer level of interaction. Human participation is extended from configuring the initial parameters for the artificial learner (a task appropriate for a Knowledge Engineer) to a collaborative effort during knowledge elicitation.

Because of the flexibility gained with the use of a genetic algorithm, and EFOPREL's design, EFOPREL offers to integrate the two sources of knowledge (the human experts and the data sets). It remains closer to the machine learning approach in that the main source of knowledge is a data set. However, EFOPREL allows the interaction with the user for shaping, generalizing or specializing rules as well as generation of alternative rules. Because of EFOPREL's complementary role with KSS0 [19], it could be easily integrated into KSS0's framework, and its representation and use will facilitate an effective interaction with the user. EFOPREL has been integrated into KEM [3] (a knowledge acquisition tool similar to KSS0 that integrates cognitive structuring techniques like Repertory Grids). In KEM, EFOPREL's initial population can be the output of an inductive learner of the decision-tree family.

EFOPREL's departure from the assumption that the data set is the only source of knowledge can also be extended to other approaches for inductive inference. For example, the decision-tree family of learners varies mostly by the numeric criteria to estimate the most informative attribute in order to split a node and further develop a tree. If this metric offers two equal values for two different attributes the algorithm proceeds arbitraryly. Perhaps it would be best to opt for another source of knowledge to decide upon these two attributes. A natural alternative is the human expert, whose domain knowledge can incorporate aspects like the costs of tests for attributes as well as aspects like the availability of the attribute (in real applications, some attributes may be missing with more frequency that other and a tree that uses available attributes should be preferred). Modifying the metric to take these aspects into account will perhaps complicate the parameters for use of the tool past acceptable limits. This is so, because the weight the metric must allocate to the new information is domain dependent and would probably require proper setting from the users based upon the their domain knowledge. For example, the optimal rate of application of specialization and generalization operators in GABIL [14] was found to be domain dependent (GABIL's early approach to this is to attempt to also learn suitable application rates).

For the families of decision sets and decision lists (AQ family and CN2 family, respectively), a metric also guides the search for specialization or generalization, but may reach a point where two alternatives have very similar values. In later stages of knowledge elicitation, it might be feasible to assume the availability of a domain theory for this type of decisions [1]. But, even in this case, how is such a domain theory constructed? Once again, rather than using decimal fractions of a complex numeric criteria, a natural alternative is be to query the domain expert.

EFOPREL offers the engine for knowledge discovery on a data set whose size or structure may prohibit the realization of this task by human beings. EFOPREL allows the provision of advice to guide its search with explicit and implicit mechanisms. EFOPREL's short first-order rules allows understanding and validation from human users. Providing such rules to novice users increases their performance. For example, in the breast-cancer data, human practitioners are reported to have an accuracy of 62% [5]. EFOPREL's simplest rule that requires inspection of just one attribute would be easy to teach to humans and would increase their accuracy (or force them to reflect on their beliefs). We expect to elaborate more on this aspect as EFOPREL is used in the development of some applications where a knowledge elicitation phase involves interaction with an expert.

References

[1] S. Augier, G. Venturini, and Y. Kodratoff. Learning first order logic rules with a genetic algorithm. In U.M. Fayyad and R. Uthurusamy, editors, *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, page 21, CA, USA, 1995. American Association for Artificial Intelligence, AAAI Press.

[2] A. Ben-David and J. Mandel. Classification accuracy: Machine learning vs. explicit knowledge acquisition. *Machine learning*, 18(1):109–114, 1995.

[3] E.I. Benitez-Guerrero. KEM: Herramienta colaborativa de extracción de conocimiento. Master's thesis, Universidad Veracruzana — LANIA, Xalapa, Veracruz, México, January 1997.

[4] P. Bonelli and A. Parodi. An efficient classifier system and its experimental comparison with two representative learning methods on three medical domains. In R.K. Belew and L.B. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 288–295, San Mateo, CA, 1991. University of California, San Diego, Morgan Kaufman.

[5] I. Bratko, I. Kononenko, N. Lavrac, I. Mozetic, and E. Roskar. Automatic synthesis of knowledge. In Y. Kodratoff and A. Hutchinson, editors, *Machine and human learning: advances in European research*, pages 25–33, London, UK, 1989. Eurotech, Kogan Page.

[6] B.G. Buchanan. Can machine learning offer anything to expert systems. *Machine Learning*, 4:251–254, 1989.

[7] G. Cestnik, I. Konenenko, and I. Bratko. Assistant-86: A knowledge-elicitation tool for sophisticated users. In I.Bratko and N.Lavrac, editors, *Progress in Machine Learning*, pages 31–45. Sigma Press, 1987.

[8] J. Cheng, U.M. Fayyad, K. Irani, and Z. Qian. Improved decision trees: A generalized version of ID3. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 100–107, San Mateo, Ca., 1988. Morgan Kaufmann.

[9] D.K.Y Chiu, A.K.C. Wong, and B. Cheung. Information discovery through hieratchical maximum entropy discretization and synthesis. In G. Piatetsky-Shapiro and W.J. Frawley, editors, *Knowledge Discovery in Databases*, pages 125–140, Menlo Park, CA, 1991. AAAI.

[10] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

[11] L. Davis. *Handbook of Genetic Algorithms.* Van Nostrand Reinhold, New York, 1991.

[12] K.A. DeJong and W.M. Spears. Learning concept classification rules using genetic algorithms. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 651–656, Sydney, Australia, 1991. Morgan Kaufmann.

[13] K.A. DeJong and W.M. Spears. On the state of evolutionary computation. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 618–623, 2929 Capus Drive, San Mateo CA 94403, 1993. University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers.

[14] K.A. DeJong, W.M. Spears, and Gordon D.F. Using genetic algorithms for concept learning. *Machine Learning*, 13(2-3):161, November 1993.

[15] S. Endo and A. Ohuchi. Disjunctive form concept learning system based on genetic algorithm. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics — Intelligent Systems for the 21st Century*, pages 4696–4701, NY, USA, 1995. IEEE, IEEE Press.

[16] R. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[17] R. Forsyth. BEAGLE — a Darwinian approach to pattern recognition. *Kybernetes*, 10(3):159–166, 1981.

[18] R. Forsyth. *Machine Learning, Principles and Techniques.* Chapman and Hall Computing, New York, 1989.

[19] B.R. Gaines and K.L.G. Shaw. Eliciting knowledge and trasferring it effectively tp a knowledge-based system. *IEEE Transactions on Knowledge and Data Engineering*, 5(1):4–14, February 1993.

[20] A. Giordana and C. Sale. Learning structured concepts using genetic algorithms. In D. Sleeman and P. Edwards, editors, *Proceedings of the Nineth International Workshop on Machine Learning*, pages 169–178, San Mateo, CA. USA, 1992. Morgan Kauffman.

[21] D.M. Grzymala-Busse and J.W. Grzymala-Busse. The usefulness of a machine learning approach to knowledge acquisition. *Computational intelligence = Intelligence informatique*, 11(2):268–279, 1995.

[22] A. Hart. The role of induction in knowledge elicitation. In A.L. Kidd, editor, *Knowledge Acquisition for Expert Systems — A Practical Handbook*, pages 165–189, New York, 1987. Plenum Press.

[23] F. Hayes-Roth and N. Jacobstein. The state of knowledge-based systems. *Communications of the ACM*, 37(3):27–39, 1994.

[24] C.Z. Janikow. A knowledge-intensive genetic algorithm for supervized learning. *Machine Learning*, 13:189–228, 1993.

[25] C.Z. Janikow. A methodology for processing problem constraints in genetic programming. *Computers & Mathematics with Applications*, 32(8):97–113, Oct. 1996.

[26] M.G. Kendall, A. Sruart, and J.K. Ord. *The advanced Theory of Statistics, Vol 3, Design and Anlysis of Time Series.* Griffin, London, fourth edition, 1983.

[27] A.L. Kidd. Knowledge acquisition — an introductory framework. In A.L. Kidd, editor, *Knowledge Acquisition for Expert Systems — A Practical handbook*, pages 109–136, New York, 1987. Plenum Press.

[28] J.R. Koza. Concept formation and decision tree induction using the genetic programming paradigm. In H.P. Schwefel and R. Maenner, editors, *Parallel Problem Solving From Nature*, pages 124–128, Berlin, Germany, 1991. Springer-Verlag.

[29] J.R. Koza. Evolution and co-evolution of computer programs to control independently-acting agents. In J.A. Meyer and S.W. Wilson, editors, *From Animals to Animats — Proceedings of the First International Conference on Simulation of Adaptive Behavior*, pages 366–375, Cambridge, MA, 1991. MIT Press.

[30] E.J. McCluskey. *Introduction to the Theory of Switching Circuits.* Mc Graw-Hill Book Company, NY. US, 1965.

[31] R.S. Michalski. Understanding the nature of learning. In R.S. Michalski, J. Carbonell, and T. Mitchell, editors, *Machine Learning An Artificial Intelligence Approach*, San Mateo, CA, 1986. Morgan Kaufmann. Volume 2.

[32] R.S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *Proceedings of the Fifth National Conference on Artificial Intelligenc*, pages 1041–1045, San Mateo, CA, 1986. Morgan Kaufmann.

[33] D. Michie, D.J. Spiegelhalter, and C.C. Taylor. Introduction. In D. Michie, D.J. Spiegelhalter, and C.C. Taylor, editors, *Machine Learning, Neural and Statistical Classification*, pages 1–5. Ellis Horwood Series in Artificial Intelligence, 1994.

[34] D.J. Montana. Strongly typed genetic programming. *Evolutionary Computation*, 3(2), 1995.

[35] S. Muggleton. *Inductive acquisition of expert knowledge.* Addison-Wesley/Turing Institute Press, Workingham, UK, 1990.

[36] G. Piatetsky-Shapiro. Discovery analysis, and presentation of strong rules. In G. Piatetsky-Shapiro and W.J. Frawley, editors, *Knowledge Discovery in Databases*, pages 229–248, Menlo Park, CA. USA, 1991. AAAI Press.

[37] J.R. Quinlan. The effect of noise on concept learning. In R.S. Michalski, J. Carbonell, and T. Mitchell, editors, *Machine Learning An Artificial Intelligence Approach*, San Mateo, CA, 1986. Morgan Kaufmann. Volume 2.

[38] J.R. Quinlan. Induction of decision trees. *Machine Learning Journal*, 1:81–106, 1986.

[39] J.R. Quinlan. Generating production rules from decision trees. In J. McDermott, editor, *Proceedings of the Tenth Internationa Joint Conference on Artificial Intelligence*, pages 304–307, Los Altos, CA, 1987. IJCAI-87, Morgan Kaufmann.

[40] M.L.G. Shaw and B.R. Gaines. An interactive knowledge-elicitation technique using personal construct technology. In A.L. Kidd, editor, *Knowledge Acquisition for Expert Systems — A Practical handbook*, pages 109–136, New York, 1987. Plenum Press.

[41] P Smyth and R.M. Goodman. Rule induction using information theory. In G. Piatetsky-Shapiro and W.J. Frawley, editors, *Knowledge Discovery in Databases*, pages 159–176, Menlo Park, CA. USA, 1991. AAAI Press.

[42] S. Spangler, U.S. Fayyad, and R. Uthurusamy. Induction of decision trees from inconclusive data. In A.M. Segre, editor, *Proceedings of the Sixth International Workshop on Machine Learning*, pages 146–150, San Mateo, CA, 1989. Morgan Kaufmann.

[43] R.E. Stepp and R.S. Michalski. Conceptual clustering: Inventing goal-oriented classifications of structured objects. In R.S. Michalski, J. Carbonell, and T. Mitchell, editors, *Machine Learning An Artificial Intelligence Approach*, pages 471–498, San Mateo, CA, 1986. Morgan Kaufmann. Volume 2.

[44] M. Tan and L. Eshelman. Using weighted networks to represent classification knowledge in noisy domains. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 121–134, An Arbor, MI, 1988.

[45] P. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, pages 369–409, 1995.

[46] P. Utgoff. ID5: An incremental ID3. In *Proceedings of the International Conference on Machine Learning*, pages 107–120, Ann Arbor, MI, USA, 1988.

[47] G.I. Webb. Control, capabilities and communication: Three key issues for machine - expert collaborative knowledge-acquisition. In *Proceedings of the 7th European Workshop on Knowledge Acquisition for Knowledge-based Systems*, Toulous, France, 1993. Springer-Verlag. Also Technical Report TR C93/04 School of Computing and Mathematics, Deakin University, Australia.

[48] G.I. Webb. Integrating machine learning with knowledge acquisition through direct interaction with domain experts. *Knowledge-based systems*, 9(4):253–266, 1996.

[49] S.M. Weiss and C.A. Kulikowski. *Computer Systems That Learn – Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems.* Morgan Kaufmann Publishers, Inc, San Mateo, CA., USA, 1991.

[50] W. Ziarko, R. Golan, and D. Edwards. An application of DATALOGIC/R knowledge discovery tool to identify strong predicate rules in stock market data. In G. Piatetsky-Shapiro, editor, *Proceedings of the AAAI 1993 Workshop on Knowledge Discovery in Databases*, pages 80–101. AAAI, AAAI Press, 1993.