FF99: A Novel Fuzzy First-Order Logic Learning System

Kwong-Sak Leung Irwin King Ming-Fun Tse
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T. Hong Kong
{ksleung, king, mftse}@cse.cuhk.edu.hk

Abstract

This paper describes a novel learning system, named FF99 that learns fuzzy first-order logic concepts from various kinds of data. FF99 builds on the ideas from both fuzzy set theory and first-order logic. Object relationships are described using fuzzy relations based on which FF99 generates classification rules expressed in a restricted form of fuzzy first-order logic. This new system has been applied successfully to several tasks taken from the machine learning literature. We demonstrate its usefulness in the applications of data mining through several experiments.

1. Introduction

Conventional inductive learning methods cannot handle fuzzy information and noisy data. Several algorithms were developed in recent years to learn concept descriptions to solve this problem. They use a fuzzy propositional attribute-value language for describing entities and classification rules. The simplicity of this formalism allows such systems to deal with large volumes of data; however, it becomes inefficient when faced with complex objects and concepts. On the other hand, learning algorithms using first-order logic accept descriptions of complex, structured entities. The fuzzy first-order logic combines the advantages of fuzzy logic and first-order logic, so it is very useful in representing knowledge in real life applications, in which the data are fuzzy, noisy and complex in nature. However, there is no learning algorithm designed using fuzzy first-order yet.

This paper describes a novel learning system, code named FF99, which learns fuzzy first-order logic concepts from data. The proposed learning algorithm is based on the FOIL algorithm, which is notable in learning non-fuzzy first order logical concepts [1]. FF99 builds on the ideas from both fuzzy set theory and first-order logic. Object relationships are described using fuzzy relations based on which FF99 generates classification rules expressed in a restricted form of fuzzy first-order logic. Like FOIL, FF99 performs a greedy search on the space of possible literals and selects a literal to add to the current clause in each iteration, a new clause is developed when the extension of the current clause cannot further reduce the cost. The whole algorithm terminates when the extension of the set of clauses cannot reduce the cost anymore.

The key to the success of the learning algorithm relies on a heuristic for assessing the suitability of a literal to be added to the current clause. A novel inductive bias, based on the continuous information measure of an error distribution, is covered in the paper. In the literal selection process, an error distribution is estimated quantitatively according to the difference of degrees of membership between the target and the learned fuzzy concepts. We have discovered that the shape of the error distribution reflects the usefulness (fitness) of a literal; in the way that a 'useful' literal is likely to produce an error distribution with distinct peaks, while a 'useless' literal produces a randomly shaped error distribution. Such qualitative observations are converted into a quantitative measurement by applying Shannon's continuous information theory. More specifically, FF99 measures the continuous entropy, which reflects the randomness of a continuous function, of the error distribution estimated for each potential literal to be added; the literal with the smallest continuous entropy will be selected.

We have tested FF99 using several databases from the UCI Machine Learning Repository. For instances, in the **iris** experiment, FF99 obtains an average accuracy of 97%. It also achieves classification accuracy of 88% in the Credit Approval (**crx**) experiment, which is higher than that obtained from C4.5. FF99 also performs adequately in the **kinship** database relational databases as it can correctly induce structural concepts.

The following section reviews fuzzy first-order logic, which is suitable for representing knowledge in real life applications. After going though the knowledge representation, the learning (data mining) algorithm is described in Section 3. The next section discusses the core part of the algorithm, which is the search heuristics, in detail. Section 5 presents the results obtained by FF99 on the **iris**, **crx** and **kinship** domain. The final two sections summarize our work and propose future research directions.

2. Knowledge representation

In this section, we discuss the fuzzy first-order logic. Specially, the notation of fuzzy Horn-clause will be given, as it is the knowledge representation used in FF99.

2.1 Fuzzy first-order logic

The classical Boolean logic is weak in the accurate modeling of real life problems, as it cannot handle imprecision or uncertainty information. Early in 1920, Lukasiewicz developed the three-valued propositional calculus [3] and worked on many-valued

0-7803-5731-0/99$10.00 ©1999 IEEE

logic. Consequently, many mathematics theories were developed for the tolerance of imprecision and uncertainty in the past few decades [3].

The basic idea of many-valued logic has been explored to some extent by a number of mathematicians since this century, but the real breakthrough was made by Zadeh [4]. The fuzzy set theory provides a systematic way to deal with imprecision which arises when the boundaries of a class of objects are not sharply defined. Among the very common examples of such classes are natural language concepts, namely linguistic variables, such as `Young women` and `Small cars`. Membership in such classes, which were suggestively called as fuzzy sets, is matter of degree rather than an all or nothing proposition. It measures the confidence that the member belongs to the normalized fuzzy set as a real number ranging from 0.0 (absolutely false) to 1.0 (absolutely true). For example, the fuzzy set `Strong` can be defined in Table 2.1.

Age	0	20	40	60	80
Membership	0.1	0.9	0.8	0.3	0.1

Table 2.1 Fuzzy set defining the fuzzy term `Strong`

In the example, `Age` is called the universe of discourse of the fuzzy set `Strong`. In general, a fuzzy set is assumed to imbed in a non-fuzzy universe of discourse, which may be any collection objects, concepts, or mathematical constructs. For example, a universe of discourse, U, may be the set of all real numbers, the set of all students in a class, etc. A fuzzy set A in U, or equivalently, a fuzzy subset of U, is characterized by a membership function $\mu_A : U \rightarrow [0,1]$ which associates with each element u of U a number $\mu_A(u)$ in the interval [0,1]. Furthermore, if the universe of discourse is a continuous quantity, one can have a continuous membership function. This is illustrated in Fig. 2.1.

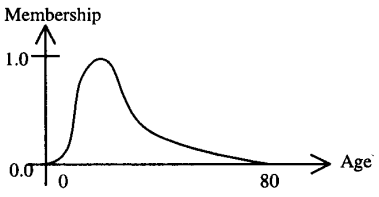


Fig. 2.1 Fuzzy set `Strong`

Fuzzy relations: Following the previous example, the fuzzy set `Strong` is defined over a one-dimensional universe `Age`. Now, if we want to redefine the fuzzy concept `Strong` over `Age` and another universe, say, `Weight`. We need a multiple-dimensional fuzzy set rather than the original single-dimensional fuzzy set. For the management of multiple-dimensional fuzzy concepts, Zadeh defined the notation of fuzzy relations. If U is the Cartesian product of n universe of discourse $U_1,\ldots,U_n$, then an n-ary fuzzy relation, R, is a fuzzy subset of U. R may be expressed as the union of its constituent fuzzy singletons $\mu_R(u_1,u_2,\ldots,u_n)/(u_1,u_2,\ldots,u_n)$, i.e.

$$R = \int_{U_1 \times U_2 \times \ldots \times U_n} \mu_R(u_1,u_2,\ldots,u_n)/(u_1,u_2,\ldots,u_n) \quad u_i \in U_i, i = 1,\ldots,n$$

where μ_R is the membership function of R.

Fuzzy Predicate: A fuzzy relation is closely related to a fuzzy predicate. In a fuzzy predicate (first-order predicate) $P(X_1,\ldots,X_n)$, P is a predicate symbol, each X_i is an individual variable which is usually defined as independent. If some values $c_1,\ldots,c_n$ are assigned to each of the individual variables, the result of the first-order predicate is a proposition. When the truth value of a predicate is in the set {0,1}, it is said to be binary first-order predicate, but when the truth value of a predicate is in the closed interval [0,1], it will be said to be fuzzy first-order predicate. The truth-value of the proposition is evaluated as the degree of membership of the responding fuzzy relation. Thus, a fuzzy first-order predicate can be considered as the membership function of a fuzzy relation over individual variables' universe of discourse. Each fuzzy predicate represents a concept, and it is refereed as a **literal** in a fuzzy Horn-clause.

Fuzzy Horn-clause: A fuzzy Horn-clause is a fuzzy logical expression in the form of:

$$H{:-}G_1,G_2,\ldots,G_n$$

H, G_1, G_2,..., G_n are fuzzy predicates. H is called the head of the clause and $(G_1, G_2,\ldots, G_n)$ is the body of the clause. It means:

IF G_1 AND G_2 AND ... AND G_n satisfies THEN H

In the inference process, G_1, G_2,..., G_n are the terms to 'prove' and they are known as the goals. Furthermore, $:\text{-}G_1$, G_2,..., G_n is called the goal clause.

2.2 Knowledge representation in `FF99`

Before we start to give the knowledge representation of the proposing data mining system, the adherence of knowledge-based method is explained. We claim that other non-knowledge-based learning methods, such as Neural Network, do not learn by acquiring sentences in a symbolic language. So, it is difficult to interpret and maintain the result of data mining. In contrast, the proposing system aims at using a powerful knowledge representation of data, so that we can extract more valuable information in data mining applications.

There are two kinds of information we have to consider. The first one is inexact information including fuzziness and certainty, both are natural in human knowledge. The second one is the relational information or first-order information, which is much more expressive than zero-order propositions especially when it is employed for the generalization of knowledge. We found that the fuzzy first-order logic is the most suitable candidate to represent knowledge in `FF99`.

Being more specify, given the examples of a set of fuzzy relations $R_1,R_2,\ldots,R_n$ and the examples of the target concept (a fuzzy predicate) C. The duty of the `FF99` is to give a **generalization** from the training examples to define the target concept in the form:

$$\begin{aligned} C' &: -L_{i_1}, L_{i_2}, \cdots, L_{i_a} \\ C' &: -L_{j_1}, L_{j_2}, \cdots, L_{j_b} \\ &\vdots \end{aligned} \qquad (2.1)$$

Where L_x is a literal defined over the fuzzy relation R_x with a particular set of variables. We call C' be the **fuzzy logical definition** of C. Following a similar notation in the `FOIL` induction algorithm, each $C' : -L_{i_1}, L_{i_2}, \cdots, L_{i_a}$ is called a **clause** (fuzzy Horn-clause). In the context of `FOIL`, each clause **covers** part of the positive examples and the whole set of the clauses is

assumed to cover at least 85% of all positive examples. We assume that the target concept could be expressed in some logical combinations of literals. The assumption is sound as it naturally models the logical combination of human logic.

3. Learning algorithm

This section introduces a learning algorithm called FF99 that is based on the modification of Quinlan's FOIL algorithm [1], which is successful in learning first-order non-fuzzy concepts. Table 3.1 compares FF99 with the well-known FOIL algorithm.

	FOIL	FF99
Knowledge representation	Predicates and Horn-clauses	Fuzzy predicates and fuzzy Horn-clauses
Training examples	Partitioned into positive set and negative set	No partition
Learning algorithm	Greedy search	Greedy search
Search heuristics	Generalization and specification by the guidance of discrete entropy	Literal evaluation by quantitative analysis and distribution analysis
Speed	Very fast	Fast, but the distribution analysis needs much processing
Accuracy	High	Very high, as the result is fuzzy and has no sharp decision boundaries

Table 3.1 Comparing FOIL and FF99

In order to induce concepts in fuzzy environment, FF99 abandons the generalization and specialization procedures in classical inductive learning methods, but uses a new searching heuristics instead. With this modification, FF99 is able to manipulate fuzzy and relational information naturally.

3.1 Algorithm overview

The task addressed by this system is to find, for one **target concept** C at a time, a **fuzzy logical definition** C' in terms of clauses of literals. Each literal is a fuzzy predicate. Here is a general form of the **fuzzy logical definition** as given in Eq. 2.1. FF99 performs a greedy search on the space of literals. Either the best literal is appended to the right hand side (RHS) of the current clause, or a new null clause is created as the current clause until the termination criteria is met. The pseudo code of the algorithm is given as follows:

```
Initialize the fuzzy logical definition C':-NULL
Loop
    Backup current C'and the current COST
    Select a subset of literals by pruning criteria
    For each literal L_i in the subset
        Append L_i to RHS of the last clause in C'
        Estimate E(t) and f(e) from C' and C
        Compute MSE and CE
        Calculate and save the COST
        Restore C'
    End For
    If the minimum COST < current COST
        Append L_i to RHS of last clause, update C'
    Else If the last clause is not a new clause
        Open a new null clause in C'
    Else
        Terminate the whole learning loop
    End If
End Loop
```

As described in the pseudo code, the whole algorithm aims at constructing a fuzzy logical definition that has a minimum *COST*, which is the heuristics to guide the selection of literal. The *COST* is estimated though two processes: the ***quantitative analysis*** and the ***distribution analysis***. They are described below.

3.2 Quantitative analysis

FF99 uses the greedy search approach to assemble clauses from literals, since a similar exhaustive search is far too expensive for the stepwise construction of clauses. By adapting the pruning techniques from the FOIL algorithm, FF99 becomes a very efficient system. The heuristics for assessing the usefulness of a literal is vitally important and should be analyzed carefully.

The error function $E(t)$: In the process of assembling clauses, FF99 chooses a literal at each step and updates the fuzzy logical definition C'. All literals (except those are pruned) are tested to build C; and an error function $E(t)$ is generated for each literal being tested, where t is a tuple (i.e. a training case) in the target concept. The definition of $E(t)$ is:

$$E(t) \equiv \mu_{C'}(t) - \mu_C(t), \qquad (3.1)$$

It compares the membership of each tuple in the target concept and the learned logical definition. So, it is a function in the space of the tuples $\{t\}$ and it is bound by [-1,1].

It is unusual to compare two fuzzy concepts by this tuple-by-tuple membership subtraction method. Examples of classical fuzzy comparison methods are the Similarity and Affinity measures described in [5]. These methods are effective in measuring the correlation and the similarity of two fuzzy sets. However, we are not searching for the literal that gives the most "similar" C' comparing with C. Instead, a literal is considered as the "best" in the sense that it is most likely to construct the best final fuzzy logical definition. The formulation of the error function facilities the measure of the errors tuple by tuple quantitatively. We can carry out a distribution analysis by generating an error distribution for the error function (see Section 3.3).

Mean Square Error: The error function E in Eq. 3.1 gives the difference of membership between the target concept and the learned logical definition. That is, for each possible variable binding in the target concept, E always returns a real value between [-1,1]. We want to characterize the error function by a numeric quantity, so that we could use this numeric quantity to assess the usefulness of a literal (a fuzzy predicate) as the next component of the right-hand side of a clause. A common approach is to calculate the mean square value of the error function (*MSE*):

$$MSE = \sum_T (E(t))^2 / |T| \quad , \qquad (3.2)$$

where T is the set of tuples and $|T|$ is the number of the set of tuples of the target concept. *MSE* is real and bounded by the limits $0 \le MSE \le 1$. Normally, we try to minimize the *MSE* to as close to zero as possible.

FF99 uses *MSE* because it is (1) simple, (2) widely accepted, (3) sign insensitive, and (4) the squaring operation suppresses the

medium errors. In such cases, the distribution analysis of the error distribution should contribute more heavily instead of depends on the *MSE* quantitative measure solely.

3.3 Distribution analysis

FF99 assesses the usefulness of a literal as the next component of the right-hand side of a clause by estimating the *MSE* and analyzing the **distribution of errors** in the error function. We have observed that:

> "A useful literal is likely to produce a nodal error distribution that has a tall peak around the zero error position and a few clear peaks in the positive or negative error range. While an improper literal is likely to produce a randomly shaped error distribution that is peaky in the entire range of errors"

It is quoted as the **nodal characteristics** of literals. It involves the concepts of an error distribution and an estimation of the Continuous Entropy (*CE*). We will discuss the distribution analysis in detail in Section 4.

3.4 Heuristics as a combination of *MSE* and *CE*

Both the *MSE* and the *CE* measurements have their strengths and weaknesses. The *MSE* measurement is simple and direct to estimate the total amount of errors; however, it has deficiency in analyzing the shape and nodal characteristics of the error distribution. The *CE* measurement remedies the problem and shoulders the load of distribution analysis. Unfortunately, it is "blind" to the position of the peaks in the error function Intuitively, we could combine these two measurements into a single *COST* heuristics so that they can compensate each other. In each step of the assembling of the fuzzy logical definition, FF99 searches for the literal that has the least *COST*.

We have evaluated several models (functions of *MSE* and *CE*) in detail and we have chosen the best one, which is detailed as follows.

$$COST = -\log(MSE^{CE}) \tag{3.3}$$

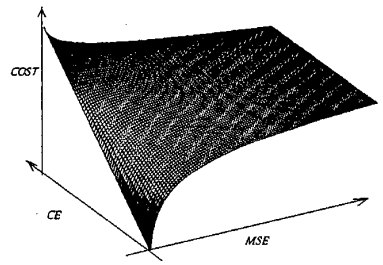


Fig. 3.1 *COST* as a function of *MSE* and *CE*

The *COST* surface in Fig 3.1 provides the heuristics to search for literals in several experiments and the results are satisfactory.

4. Distribution analysis

As mentioned in Section 3, the search heuristics (*COST*) in FF99 involves the quantitative analysis (*MSE*) and the distribution analysis (*CE*). This section concentrates on the distribution analysis. It first introduces the concept of error function. Then, it describes the nodal characteristics of literals and finally the calculation of the continuous entropy.

4.1 The error distribution *f(e)*

As defined in Eq. 3.1, the error function *E(t)* returns a set of errors $\{e\}$ on the space of tuple $\{t\}$. The frequency distribution of this set of errors $\{e\}$ is called the **error distribution** *f(e)*. In practice, the "actual" error distribution cannot be determined unless an infinite number of training samples is given. So, we have to **estimate** the error distribution from the error function. A survey of existing density estimation methods is given by [6]. Among these methods, the kernel estimator, which is discussed in next subsection, is used in FF99 to construct *f(e)* from *E(t)* because it is simple, easy to implement and it performs well even the sample size is small.

4.2 Kernel estimator

A kernel estimator is defined by:

$$f(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - X_i}{h})$$

where h is the window width, n is the sample size and K is a kernel function which satisfies several conditions [7]. According to [7], FF99 chooses the biweight kernel function $K(x)$ and the window width h as follows:

$$K(x) = \begin{cases} 15(1-x^2)/16 & ,|x|<1 \\ 0 & ,otherwise \end{cases}$$

$$h = 3.729 \times \sigma \times n^{-1/3} / 3$$

where σ is the standard deviation

Finally, by using the biweight kernel estimator with the window width defined above, we can estimate the error distribution *f(e)* from the error function *E(t)*. These two functions provide information for the quantitative analysis and distribution analysis of literals.

4.3 Nodal characteristics of *f(e)*

In subsection 3.3, we mentioned the nodal characteristics of literals that a useful literal is likely to produce a nodal error distribution with a clear "zero error peak". This subsection will discuss these nodal characteristics.

The zero error peak: As defined, the error function *E(t)* gives the difference of the membership between each tuple of the target concept and the learned logical definition. Suppose now for all t, $E(t) \approx 0$, the error function *f(e)* will give a zero error peak as shown in Fig. 4.1 because it is very **probable** that for a certain tuple, the error (the difference of membership between the fuzzy logical relation C' and the target concept C) is near zero.

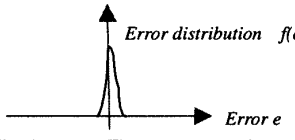


Fig. 4.1 The zero error peak

Negative error peaks: Suppose that the target concept C should be described as the fuzzy Horn-clauses:

$$C \text{:-} L_1, L_2$$

In this case, the membership of each tuple in the target concept

should be approximately equal to the minimal of the corresponding membership of L_1 and L_2:

$$\mu_C(t) \approx \min(\mu_{L_1}(t), \mu_{L_2}(t))$$

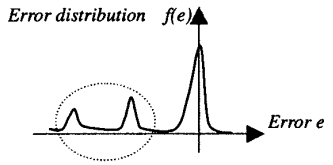
Now, suppose we are learning the target concept C and we are in the moment to search for the 2nd literal to add in the right-hand side of the clause:

$$C: -L_1, ? \quad \Leftrightarrow \quad C': -L_1$$

By the definition of the error function in Eq. 3.1, we have:

$$E(t) = \mu_{C'}(t) - \mu_C(t) \approx \mu_{L_1}(t) - \min(\mu_{L_1}(t), \mu_{L_2}(t))$$

For the covered tuples, $E(t) \approx 0$ and it contributes to the zero error peak; while $E(t)<0$ for other tuples. Moreover, we argue that the uncovered tuples are belonging to a few groups in the concept represented by L_2. So that, $E(t)$ is likely to have some **typical negative values**; and each typical negative error value corresponds to a negative error peak. Fig. 4.2 visualize $f(e)$, it shows the situation that the uncovered literals could be grouped into two classes in the concept represented by L_2.



Fig, 4.2 Two negative error peaks

Positive error peaks: Now, suppose that the target concept C should be described as the set of fuzzy Horn-clauses:

$$\begin{cases} C: -L_1 \\ C: -L_2 \end{cases}$$

In this case, the membership of each tuple in the target concept should be approximately equal to the maximal of the corresponding membership of L_1 and L_2:

$$\mu_C(t) \approx \max(\mu_{L_1}(t), \mu_{L_2}(t))$$

In a similar way, we are now constructing the 2nd clause of C':
By the definition of the error function in Eq. 3.1, we have:

$$E(t) = \mu_{C'}(t) - \mu_C(t) \approx \mu_{L_1}(t) - \max(\mu_{L_1}(t), \mu_{L_2}(t))$$

This time, $E(t)>0$ for the uncovered tuples. By the same argument, $E(t)$ is likely to have some **typical positive values**; and each typical positive error value corresponds to a positive error peak. Fig. 4.3 show two positive error peaks.

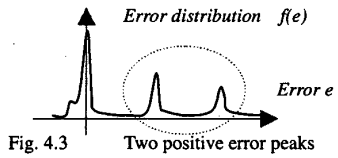


Fig. 4.3 Two positive error peaks

4.4 Applying information theory

Shannon introduced the information and the concepts of entropy [8]. It is defined as a measure of "uncertainty" or "randomness" of a random phenomenon. The information theory is famous in the field of inductive learning, e.g. ID3 [2] and FOIL use the discrete entropy to guide the construction of decision trees and logical definitions.

However, all knowledge in FF99 is modeled as fuzzy predicate, of which of the membership function is continuous and not discrete in nature. Intuitively, the discrete entropy cannot be applied in FF99; nevertheless, the continuous entropy shows to be very effective in characterizing the shape of the error distribution $f(e)$. As discussed, CE measures the "randomness" of a continuous function, which is the error distribution $f(e)$ in FF99. According to the nodal characteristic of $f(e)$, a "suitable" literal produces a nodal error distribution that $f(e)$ has a tall peak around the zero error position and a few clear peaks in the positive or negative error range; in contrast, a "unrelated" literal produces a random-shaped error distribution that $f(e)$ is not nodal and is peaky in the whole range of errors. Interesting enough, a nodal $f(e)$ is less random than a peaky $f(e)$ in shape; such observations inspired us to apply the information theory as measure the randomness of $f(e)$. These observations are shown in Fig. 4.4. Furthermore, experiment results show that CE is a very effective heuristics in evaluating literals, even it is used solely to be the *COST* of a literal.

However, the speed of FF99 is not acceptable in learning multiple order information as it spends about 30 minutes to obtain the above solution. Because FF99 is currently implemented by Matlab scripts and the pruning techniques are not fully used yet. These pruning techniques mainly involve the selection of variables (e.g. X_1, X_2, X_3 in the definition of brother) that are adopted from the FOIL algorithm.

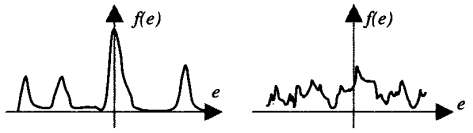


Fig. 4.4 (a) A "suitable" literal gives a nodal $f(e)$ and a nodal $f(e)$ gives a small CE. (b) A "unrelated" literal gives a peaky $f(e)$ and a peaky $f(e)$ gives a large CE

Bounds of CE: In order to use the CE as a part of the heuristics, we must know the upper and lower bounds of it. An impulse distribution gives the minimum $CE = -\infty$, and a uniform distribution (a white spectrum) gives the maximum $CE=\log_2|R|$, R is the range of the spectrum. These two distributions are shown in Fig. 4.5.

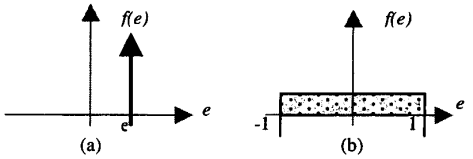


Fig. 4.5 (a) Impulse $f(e)$ gives the smallest CE. (b) White spectrum gives the largest CE

Theoretically, the range of errors should be limited by [-1,1]. However, the error function $f(e)$ is produced by a density estimation process in FF99; so that the practical range of errors is R=[-2,2] and $|R|$=4. Finally, we get the limits of CE as:

$$-\infty \le CE \le \log_2 4 = 2$$

5. Results

This section presents results obtained by FF99 on a variety of learning tasks reported in the literature in order to show that it is a powerful and general learning mechanism. The **iris** experiment tests the classification accuracy of FF99. The **crx** experiment shows that FF99 is useful in real life data mining application and

FF99 learns relational concept in the **kinship** experiment.

5.1 Classifying iris plants

The **iris** plants database contains 3 classes of 50 instances each, where each class refers to a type of iris plant. The class `setosa` is linearly separable while `versicolor` and `virginica` are not linearly separable from each other. Each training instance is described by four numeric attributes: Sepal Length (SL), Sepal Width (SW), Petal Length (PL) and Petal Width (PW). Since the attributes are numeric values, we have to preprocess them by fuzzification techniques. The percentiles information of data are used to estimate the parameters of the membership function.

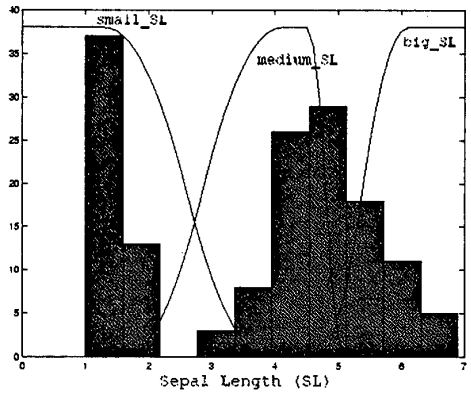


Fig. 5.1 Fuzzification by incorporation of percentiles information

Three individual sets of experiment are conducted by randomly selecting 90 instances (60%) for training and 60 instances (40%) for testing. One fuzzy logical definition is generated for each class; the memberships of these three logical definitions are compared and the one with the greatest membership is considered as the classification result.

Iris class	Hit rate (%)	Exp. set I	Exp. set II	Exp. set III
Setosa	training	100.0	100.0	100.0
	testing	100.0	100.0	100.0
Versicolor	training	94.4	95.6	94.4
	testing	96.7	95.0	95.0
Virginica	training	91.1	93.3	96.7
	testing	100.0	96.7	91.7
Classify by comparing memberships	training	95.6	98.9	95.6
	testing	100.0	95.0	96.7
Average	training	96.7		
	testing	97.2		
Overall		96.9		

Table 5.1 Results of FF99 on **iris** domain

The following are the typical results. Note that `virginica` is defined by two clauses, i.e. a logical OR of two fuzzy predicates.

```
setosa :- small_SL, small_PL
versicolor :- medium_PL, medium_PW
virginica :- big_PW
virginica :- big_PL
```

The classification results obtained from FF99 are excellent, as they are comprehensive and understandable by human beings. This property is particularly important in the applications of data mining, when we want to discover and analyze the underlying information from raw data. Furthermore, the classification accuracy is very high when compared with other learning algorithms. Table 5.2 compares the accuracy of our learning algorithm with that of [9].

Algorithm	FF99	FAQR	GVS	IVSM
Accuracy (%)	96.9	97	96	95.78
Algorithm	NTgrowth	Dasara	C4.5	
Accuracy (%)	94.87	94.67	93.87	

Table 5.2 Results of the **iris** experiments from different algorithms

5.2 Learning credit card approval criteria

The domain for this case study, named **crx**, concerns approval of credit facilities using a **real dataset** provided by a bank. The 690 cases are split 44.5% to 55.5% which represent credit card "approved" and "rejected" respectively. The 15 attributes include 6 with numeric values and 9 discrete-valued (nominal) attributes. The preprocessing stage transforms each binary attribute into a crisp concept. Meanwhile, each non-binary nominal attribute is **linearlized** into several crisp concepts. Each crisp concept means a possible attribute value. And each numeric attribute is **fuzzified** into 3 fuzzy concepts: `large`, `medium`, and `small`.

Five individual sets of experiment are conducted by randomly selecting 582 cases (90%) for training and 65 cases (10%) for testing. We use 0.5 as the cutting membership of the learned logical definition to determine whether the credit card is "approved" or "rejected". Three of the five experiments give the same resulting logical definition, while the other two experiments give another one result.

Exp. set	Learned logical definition	Error % Training	Testing	Overall
1	`Approve :- C9` `Approve :- C4big`	13.4	12.3	13.3

2	`Approve :- C9`
	`Approve :- C4big`
	`Approve :- C6x, C15small`
		13.3	12.3	13.2

3	`Approve :- C9`
	`Approve :- C4big`
	`Approve :- C6x, C15small`
		13.6	9.2	13.2

4	`Approve :- C9` `Approve :- C4big`	13.8	9.2	13.3
5	`Approve :- C9` `Approve :- C4big`	13.8	9.2	13.3
	Average	13.6	10.4	13.3

Table 5.3 Results of FF99 on the **crx** domain

We observed that: (1) Although there are two groups of the learned logical definition, the orders of assembling the logical definition in all experiments are the same. That is, `C9` is found to be the first clause and `C4big` is the second clause in all experiment sets. It shows that the heuristics of evaluating literals is consistent and effective. (2) Despite the learned logical definitions are very brief, it has high predicative power. (3) The target concept `Approve` is crisp (non-fuzzy); nevertheless, it gives the certainty (the membership) of credit card approval. In general, the certainty is more informative and useful than the class value only.

The **crx** domain was used in Quinlan's C4.5 experiments [10]. The results are summarized as:

Knowledge representation	Average complexity	Average error rate (%)
Decision tree	9.7 nodes	15.8
Production rule	7.8 rules	16.5
Composition rule	11.0 rules	14.4

Table 5.3 Results of C4.5 on the **crx** domain

Obviously, FF99 performs better than C4.5 in the **crx** experiments as it has smaller complexity (2-3 rules) and lower error rate (13.3%). The advantage of FF99 comes from the knowledge representation, which can describe fuzzy concepts and define soft boundaries between concepts. The cutting membership and the parameters of fuzzification could be tuned in order to get even better results. The successful results show that FF99 is a powerful and general data mining system.

5.3 Learning kinship relation

The **kinship** relational database consists of 24 unique people in two families. The training data is provided in form of non-fuzzy predicate, e.g. `father(Tom, Alex)` indicates that "Tom is the father of Alex". In our experiment, we try to learn the logical definition of `brother` by supplying the training relations in `father`, `brother` and `son`. The correct logical definition should be:

$$\texttt{brother}(X_1, X_2) \texttt{:-} \texttt{father}(X_3, X_1), \texttt{son}(X_2, X_3)$$

FF99 successfully generated the above logical definition of the `brother` relation. We see that the knowledge representation of FF99 is also powerful in handling relational (2nd order) and other higher order information.

6. Discussion

6.1 Knowledge representation

The fuzzy first-order knowledge representation used in FF99 is good in handling both inexact and relational data. However, this knowledge representation also suffers two main problems:

1. It is not expressive enough, as it does not allow the logical NOT. Moreover, it can only model imprecise information but not uncertain information. In order to overcome this shorting, we adopt an extension in which a "clause" may contain negated literals on its right-hand side. Furthermore, we could attach certainty factors to each literal and each clause.

2. The validity and the effectiveness of this knowledge representation strongly rely on the "quality" of the fuzzy relations. There is no crisp sense of "correct" or "wrong" in the membership function of a fuzzy relation and the judgement is normally subjective. This issue is concerned in the fuzzification of numeric data. From our experiments, we found that the percentile information is generally effective in guiding the fuzzification process. However, the preprocessing (i.e. fuzzification or lineralization of raw data into form of predicates) is very difficult to be totally automated.

On the other hand, this knowledge representation gives some comprehensive and understandable results. Given this, FF99 has high potential in the real applications of data mining. For example, it may be used in mining customer preferences of supermarket, extracting fuzzy concept from questionnaire, etc.

6.2 Learning algorithm

FF99 suffers from the greedy search paradigm in the clause assembly process since it does not explore any alternatives once a clause is built, i.e. the "best" choice may not be the "correct" choice at each step. However, a similar exhaustive search is far too time-consuming for the stepwise construction of clauses. In FOIL, Quinlan suggested the alternative of beam search, which retains the best N partial structures at each step, and it increases search effort by a factor of N compared with that of greedy search.

The greedy algorithm also depends heavily on the design of the search heuristics. In FF99, the search heuristics is terminated by the quantitative analysis and the distribution analysis. Although we have no formal mathematical proof for the effectiveness of the heuristics, especially the part of distribution analysis, the experiment results are satisfactory and they show that the heuristics provides a good guidance in the construction of fuzzy logical definition. However, we are now seeking more real applications to test the heuristics. We are also trying to formulate the heuristics mathematically.

7. Conclusion

In conclusion, FF99 induces fuzzy concepts that are comprehensive to human beings. Also, it provides a satisfactory way to handle fuzzy relational data. The learning algorithm is robust and efficient. Finally, the experiment results have shown the novel learning system is suitable for data mining from both continuous and discrete information, and is also very tolerant to noisy data. The continuous entropy measure is an effective heuristics. Our three experiments have shown that FF99 is capable of generating correct classification in fuzzy concepts to variety of test data sets.

8. References

[1] Quinlan, J.R., *Learning Logical Definitions from Relations, Machine Learning*, 5, 239-266, 1990.

[2] Quinlan, J.R., *Induction of Decision Trees.* Machine Learning 1(1): 81-106.

[3] Hiroyuki Yasui, Yoshiaki Hamada, Masao Mukaidono, *Fuzzy Prolog based on Lukasiewicz implicatoin and bounded product*, IEEE Fuzzy Systems 1995.

[4] L.A.Zadeh, *Fuzzy Sets*, Information and Control 8 (1965) 338-353.

[5] K.S.Leung, Y.T.So, *Consistency Checking for Fuzzy Expert Systems*, Int. J. of Approx. Reasoning 1993, 9: 263-282

[6] B.W.Silverman, *Density Estimation for Statistics and Data Analysis*, Chapman and Hall.

[7] Terrell, *The maximal smoothing principle in Density Estimation*, Amer. Stat. Assoc. 85. P470-477, 1990.

[8] Shannon, C.E.. *A Mathematical Theory of Communication.* Bell System Tech. J., 27, 379-423, 623-656.

[9] C.J.Tsai, S.S.Tseng, C.H.Wang, C.T.Yang, M.F.Jiang, *A Fuzzy Inductive Learning Algorithm for Parallel Loop Scheduling*, IEEE SMC 1997.

[10] Quinlan J.R., *C4.5: Programs for machine learning*, Morgan Kaufmann, Oct 1992.