Pre-publication draft of a paper which appeared in the proceedings of
PRICAI 1992 pp 555-561

Function finding in classification learning

Simon P. Yip, Dept. of Comp.Sci., Swinburne University of Technology, Hawthorn, 3122, Australia
E-mail: simon@saturn.cs.swin.oz.au
Geoffrey I. Webb, Dept. of Comp. Sci., Deakin University, Geelong 3217, Australia
E-mail: webb@cm.deakin.oz.au

Abstract :

The paper describes a method for extending domain models in classification learning by deriving new attributes from existing attributes. The process starts by finding functional regularities within each class. Such regularities are then treated as additional attributes in the subsequent classification learning process. The research revealed that these techniques can reduce the number of clauses required to describe each class, enable functional regularities between attributes to be incorporated in the classification procedures and, depending on the nature of data, significantly increase the coverage of class descriptions and improve the accuracy of classifying novel instances when compared to classification learning alone.

1. Introduction :

Attribute-value classification learning algorithms, such as AQ (Michalski, 1980, 1986) or ID3 (Quinlan,1986), aim to derive classification procedures capable of defining one class of instances as different from other classes. The condition parts of the classification rules are based on the range of values of each attribute. Previous algorithms have not in general supported the derivation of conditions based on relationships between attributes. Discovery learning algorithms, such as BACON (Langley & Zytkow, 1989) and ABACUS (Falkehainer & Michalski, 1986), aim at discovering functional regularities in empirical data. They do not attempt to include classification procedures.

This paper reports an attempt to use discovery learning to discover relations between attributes, and to use these to extend the domain model by creating new attributes in subsequent classification learning, with the objective of inducing better classification procedures: procedures that incorporate functional regularities between attributes. This idea was first hinted in one of Michalski 's (1984) constructive generalization rules which attempts to generate new attributes as arithmetic functions of original attributes. This paper provides a systematic examination and elaboration of this idea.

Let us use a simple example to illustrate the idea. Consider the following data:

Speed	Speed_limit	Class
71	70	speeding
80	60	speeding
110	100	speeding
80	110	not_speeding
60	70	not_speeding
100	100	not_speeding

Most attribute-value classification learning methods would be unable to discover the underlying relationship between the attributes : "Speed" and "Speed_limit". The following results obtained by a classification learning algorithm are typical :

IF ((Speed=110) AND (Speed_limit=100)) OR
((71<=Speed<=80) AND (60<=Speed_limit<=70))
THEN class = speeding
IF ((Speed=60) AND (Speed_limit=70)) OR
((80<=Speed<=100) AND (100<=Speed_limit<=110))
THEN class = not_speeding

As we can see, existing classification methods induce rules based on the values of individual attributes. A more general result for the above task would be the rules :

IF (Speed – Speed_limit) > 0 THEN class = speeding
IF (Speed – Speed_limit) <= 0 THEN class = not_speeding

If we can induce from the above data the relation: ((Speed–Speed_limit) > 0) and then transform it into an additional attribute in classification learning, we can arrive at the above rules. The research reported herein seeks to achieve this goal. To this end, before performing traditional classification learning, we induce functional relations between attributes and then use these relations to arrive at better, more concise rules.

In the following sections, we discuss the theoretical perspective of function finding in classification learning and summarize the research background of both classification learning and discovery learning. Two algorithms, one from each research area, will be discussed in more detail. The selected algorithm in the inductive classification learning area is DLG (Disjunctive Least Generalization, Webb, 1991a) and that in discovery learning is BACON (Langley & Zytkow, 1989; Langley, Zytkow, Simon and Bradshaw 1984). The paper then discusses and illustrates with examples, the objectives, rationale and procedure of employing discovery learning within a classification learning system.

2. Theoretical perspective of function finding in classification learning:

Function finding in classification learning can be viewed as one type of constructive induction, which refers to the use of background knowledge or a domain theory to derive additional information about an example. Obviously, if the class description is outside the description space that is defined by the domain model which is stated in terms of available attributes or features, then it can only be learnt by extending that space. Indeed, it is possible that the relevant attributes or best features that could be used in the class description may not be explicit or included in the examples (Elio & Watanabe, 1991). Constructive induction was first formalised by Michalski (1984), who identified several constructive generalization rules. The one relevant to this paper is the *detecting descriptor interdependence* rule.

2.1 The detecting descriptor interdependence rule :

To paraphrase Michalski (1984), suppose we are given attribute descriptions characterizing a class of objects. Such descriptions specify only attribute values of objects, they do not characterize the objects' structure. If a system observes that the values of attribute X increase as the values of attribute Y increase, a two-attribute descriptor: M(x,y) can be created, signifying that x and y have a monotonic relationship. The idea can be extended in two directions. The first is to create M-descriptors dependent on some condition (COND) that must be satisfied by the events under consideration : M(x,y)-COND. For example, descriptor M(length,weight)-blue states that the length and weight have a monotonic relationship for blue objects. The second direction of extension is to relax the requirement for the relationship. For example, the correlation coefficient between x and y can be evaluated and if the coefficient value is within a range, a descriptor R(x,y) is created. Similar to the M- descriptors, R-descriptors can be extended to R-COND descriptors. The M- or R-descriptors can also be used to generate new descriptors. For example, using a BACON like algorithm, if the values of x increases with values of y, a new descriptor: $z=x/y$ may be generated.

In finding functions to enhance class descriptions, we perceive two main objectives : a. increase class specificity b. increase generality and thus the boundary of the class set. A good class description must be general enough to cover new positive instances and specific enough to reject new negative instances. The learning task is to find the best combination of general and specific descriptions (Elio & Watanabe, 1991).

2.2 Function finding in classification learning within a finite set:

Consider the following data:

Suppose we have a set S of all values of x and y attributes :

$S = \{ x{:}N;\ y{:}N \}$ $N = \{1,2...9,10\}$

There are 100 elements in S. Assuming each element can be classified as either positive or negative, suppose we have the following training set :

x	y	class
2	4	positive
4	8	positive
5	10	positive
5	4	negative
6	3	negative

Without function finding, the class description may be :

IF $((2<=x<=4)\ \&\ (4<=y<=8))$ V
$((x=5)\ \&\ (y=10))$ THEN class = positive
IF $((5<=x<=6)\ \&\ (3<=y<=4))$ THEN class = negative
Or
Positive = $\{x{:}N;y{:}N \mid ((2<=x<=4)\ \&\ (4<=y<=8))$ V
$((x=5)\ \&\ (y=10))\}$
Negative = $\{x{:}N;y{:}N \mid ((5<=x<=6)\ \&\ (3<=y<=4))\}$

With such a description, the positive set consists of 16 elements :{(2,4),(2,5),(2,6)...} and the negative set 4 elements :{(5,3),(5,4),(6,3),(6,4)}. The remaining 80 elements are uncovered by the description. Suppose we discover the regularity within the positive class : $y = x * 2$. Using such a regularity as a Boolean attribute, the class description can be:

Positive = $\{x{:}N;y{:}N \mid y = x * 2\}$;
Negative = $\{x{:}N;y{:}N \mid y \neq x * 2\}$

With such a description, the positive set consists of 5 elements :{(1,2),(2,4),(3,6),(4,8),(5,10)} and the negative set 95 elements. Thus, the specificity of the description is increased and the number of uncovered cases reduced to zero.

Let us now look at a different training set :

x	y	class
2	4	positive
4	10	positive
5	7	positive
5	4	negative
6	3	negative

Without function finding, the class description may be :

Positive = $\{x{:}N;y{:}N \mid ((2<=x<=4)\ \&\ (4<=y<=10))$ V
$((x=5)\ \&\ (y=7))\}$
Negative = $\{x{:}N;y{:}N \mid ((5<=x<=6)\ \&\ (3<=y<=4))\}$

Suppose we discover, from the training set, the regularity :

Positive : $2<= (y – x) <=6$; Negative: $-3<= (y – x) <=-1$

This regularity is more general than that of the previous training set. If we transform this regularity as an additional attribute, the data becomes:

x	y	y–x	class
2	4	2	positive
4	10	6	positive
5	7	2	positive
5	4	-1	negative
6	3	-3	negative

With this derived training set, the class description can be :

Positive = $\{x{:}N;y{:}N \mid 2<=(y – x)<=6\}$;
Negative = $\{x{:}N;y{:}N \mid -3<=(y – x)<= -1\}$

With such a description, the positive set has 36 elements, the negative set 24 and the uncovered set 40.

Suppose, we can generalize the regularity further and the class description becomes :

Positive = $\{x{:}N;y{:}N \mid y > x\}$;
Negative = $\{x{:}N;y{:}N \mid y < x\}$

With such a description, the positive and negative class both have 45 elements each and the number of uncovered cases is decreased to 10.

This example shows that incorporating regularity can reduce the number of uncovered cases and reduce the number of conjunctive clauses in class descriptions. It also illustrates the impact of incorporating regularities with different degree of specificity.

3. Classification Learning:

An important objective of inductive learning from a set of positive and negative examples is to determine a general description capable of discriminating positive from the negative examples. Such induced descriptions can be used as classification procedures in knowledge-based systems to classify novel instances. AQ (Michalski,1980,1986) is typical of an induction algorithm that utilizes class descriptions.

3.1 Disjunctive Least Generalization algorithm (DLG) :

The inductive rule learning algorithm used in this research is disjunctive least generalization (DLG) (Webb, 1991a). It is a variant of the AQ algorithm (Michalski,1980,1986) that can process continuous attributes and does not use arbitrary parameters to constrain its search. It is an efficient data driven learning algorithm that can generate disjunctive class descriptions. It differs from other members of the AQ family in the manner in which it develops disjuncts. These are developed by least generalization (Plotkin, 1970,1971). The DLG algorithm can be expressed as follows :

```
Input : POS ( a training set of instances belonging to the  class
                 of interest)
        NEG  ( a training set of instances NOT belonging to
                 the class of interest)
Output : R ( a disjunction of  non-disjunctive descriptions for
               the class )
Initialize R to False;
While POS is not empty
Begin
   Initialize C to False;
   For X set to each successive instance in POS
      Set L to the least generalization of C that covers X;
      If L does not cover any instances in NEG set C to L;
   Remove from POS all instances covered by C;
   Set R to R v C;
End .
```

For example, with DLG and given the following training instances :

X	C	class
3	red	positive
5	yellow	positive
1	brown	positive
1	red	negative
4	brown	negative
6	blue	negative

the class descriptions for each class (expressed in rules) that might be induced are :

IF $((3<=X<=5)$ & $(C\in\{red,yellow\}))$ V
 $((X=1)$ & $(C\in\{brown\}))$ THEN class= positive
IF $((X=1)$&$(C\in\{red\}))$ V
 $((4<=X<=6)$&$(C\in\{brown, blue\}))$THEN class= negative

Thus, given the above induced rules and a novel object with attribute values of, for example, X=4, and C= red, one may classify that object as of class "positive". In this example, there are only two classes. For data with more than two classes, the class under focus is positive and other classes are regarded as negative.

4. Function finding from observations :

Another type of inductive learning is learning from observation. In this case, the program or the observer searches for regularities and general rules explaining all or at least most observations.

4.1 BACON

BACON (Langley, Zytkow, Simon and Bradshaw, 1984) is a sequence of discovery systems (version 1 to 6) . For discovering simple laws, BACON's most basic operation involves discovering a functional relation between two numeric terms. To discover laws relating two numeric variables, BACON employs three simple heuristics :
INCREASING : if the values of X increase as the values of Y increase, then define the ratio X/Y and examine its values.
DECREASING : if the values of X increase as the values of Y decrease, then define the product XY and examine its values.
CONSTANT : if the values of X are nearly constant for a number of values, then hypothesize that X always has this value.

For example, consider the following empirical data :

X	Y	Z
2	2	30
4.5	3	20
8	4	15
12.5	5	12

The BACON system, on examining the data and applying its "INCREASING" heuristic will find that the values of X increase with the values of Y. It will then define the ratio X/Y. It further realizes that the values of X/Y increase with the values of Y and so define the ratio (X/Y)/Y or X/Y^2. On applying the "CONSTANT" heuristics, it will realize that the values of X/Y^2 always equal to 0.5; thus it will conclude that $X/Y^2 = 0.5$. Similarly, on applying the "DECREASING" and then the "CONSTANT" heuristics, it will conclude that Y*Z = 60. Later versions of BACON can discover more complex regularities.

5. BACONC (BACON within Classification):

In the research, we applied BACON to example data within different classes, to see if we can discover regularities descriptive of that class. The BACON algorithm implemented to achieve such end, which we refer to as BACONC, was modified from the original BACON.

As mentioned above, a good class description must be general enough to cover new positive instances and specific enough to reject new negative instances. Thus, using function finding to enhance class descriptions, we attempt to find two types of functions: one to increase the specificity of the class descriptions, the other, the generality. To find specific functions, BACONC incorporates BACON algorithms to discover specific functions within each class. To find class description generality, BACONC searches from a pre-defined function space for functions with a generalized range (e.g. $2<=(y^2*x)<= 9$), capable of discriminating the positive class from the negative. If a function with generalized range derived from the positive class covers no negative instance, that function is accepted. The algorithm is described in the next section.

5.1 Discriminant function finding algorithm (DFFA):

DFFA, which finds functions with generalized range, discriminating the positive class from the negative, can be expressed as follows:

```
Input:   POS (a set of instances  belonging to the class of
                 interest)
         NEG (a set of instances NOT belonging to the class of
                 interest)
Output:  a function from the function space with generalized
         range or (Function not found)
Initialize a boolean variable FOUND to False;
Initialize all functions in the function space to unchecked;
While not FOUND and not all functions checked
Begin
   Select the next unchecked function from the function space;
   Derive a generalized range for the function from the
         POS instances;
   If the function does not cover any instance of NEG,
         set FOUND = True;
End.
```

Suppose we have a pre-defined function space consisting of the following functions, which involve two numeric terms (X and Y) and two constants (K_1 and K_2) :

$K_1 <= (Y+X) <= K_2$; $K_1 <= (Y–X) <= K_2$; $K_1 <= (Y*X) <= K_2$;
$K_1 <= (Y/X) <= K_2$; $K_1 <= (Y*X^2) <= K_2$; $K_1 <= (Y^2*X) <= K_2$;
$K_1 <= (Y/X^2) <= K_2$; $K_1 <= (X/Y^2) <= K_2$

Consider the following data:

X	Y	Class
4	5	positive
6	3	positive
5	3	positive
1	8	negative
10	7	negative
6	10	negative

The algorithm looks at the first function (Y+X) and derives from the positive instances : 8<=(Y+X)<=9. Since it covers a negative instance, it is abandoned. The next function in the function space is then examined. The function derived from the positive instances is: -3<=(Y–X)<=1. Since it covers a negative instance, it is also abandoned. The third function derived : 15<=(Y*X)<=20 covers all positive instances and no negative instance; so it is accepted as the function capable of discriminating the positive class from the negative.

5.2 BACONC algorithm:

Incorporating the ideas introduced so far, BACONC can be expressed as follows :

```
Input: POS ( a training set of instances belonging to the  class
             of interest)
       NEG ( a training set of instances NOT belonging to the
             class of interest)
Output  :  S  ( a set of functions descriptive of  the class)
Initialize S to empty;
For each non-redundant non-numeric condition
   For each non-redundant numeric attribute combination
      Find a function (if any) with BACON that covers all the
            instances of POS  and not any instance of NEG;
      If the above function is not found then find a function
         (if any) with a generalized range with DFFA
         (as defined above) that covers all the instances of
         POS  and not any instance of NEG;
      Include such function so found in S .
```

6.Using function finding in classification learning:

In this section, we discuss incorporating function finding in classification learning. First, by applying discovery learning to instances belonging to a certain class, functions that are relevant to that class may be found. Such functions can be treated as additional attributes in the subsequent step of classification learning. To complete the induction process, following DLG induction, we performed "Conservative Conjunct Deletion" and "Range generalization".

Conservative conjunct deletion is achieved by two scans through the clauses of each rule in opposite directions. For each clause, delete it and then see if the rule covers any cases in other classes. If it does, restore it. Start each scan from the full rule. Finally, delete only those conjuncts that were deleted in both scans.

Range generalization (Webb, 1991b) is a step to further generalize the constant range of functions discovered by DFFA. Consider the rules:

IF (5<=(y–x)<=10) & (6<=w<=10) THEN class = positive.
IF (-20<=(y–x)<=-5) & (-1<=w<=3) THEN class = negative

Suppose we want to range extend 5<=(y–x)<=10. Range generalization first deletes this clause and then examines all negative instances mis-classified by this description. Suppose, it mis-classifies three negative instances with values of (y – x) equal to -20, -10 and -5. Range generalization finds, among the above values, the maximum value that is below the lower bound of the function to be extended, and the minimum value that is above the upper bound. The values are -5 and none respectively. Thus, for the lower bound of the function, we know that we can extend it to somewhere between 5 and -5 (in this research, we take the mid-point). As to the upper bound, we can generalize that direction to infinity. Using this process, and depending on the examples in the training set, the above rule may be generalized to :

IF ((y–x)>0) & (6<=w<=10) THEN class = positive
IF ((y–x)<=0) & (-1<=w<=3) THEN class = negative

Since this function range generalization step is more radical, its execution can be made dependent on application domain and certain user-defined criteria.

The resulting algorithm is called **DLG$_{ff}$** (function finding in disjunctive least generalization) :

```
Input :  POS ( a training set of instances belonging to the  class
               of interest)
         NEG  ( a training set of instances NOT belonging to the
               class of interest)
Output  :  R  ( a disjunction of  non-disjunctive  descriptions
               for the class )
functions <-- BACONC(POS, NEG);
Extend the descriptions of cases in POS & NEG to
```

```
    include each  function as additional attribute;
rules <-- DLG(POS, NEG);
Generalize rules using Conservative Conjunct Deletion;
Generalize rules using function range generalization
                        (if applicable).
```

7. Evaluation :

The DLG$_{ff}$ algorithm can be illustrated by the following examples :

Study 1 : Let us examine another example on "Speed Limit" using DLG$_{ff}$:

Siren	Speed	Speed_limit	Class
No	71	70	speeding
No	80	60	speeding
No	110	100	speeding
No	80	110	not_speeding
No	60	70	not_speeding
No	100	100	not_speeding
Yes	110	100	not_speeding
Yes	90	110	not_speeding

When DLG alone is applied to this data, the result is five separate disjunctive clauses created for the class description :

IF ((Siren ∈ {No}) & (Speed=110) & (Speed_limit=100)) V
((71<=Speed<=80) & (60<=Speed_limit<=70))
THEN class = speeding
IF (Siren ∈ {Yes}) V
((80<=Speed<=100) & (100<=Speed_limit<=110)) V
((Speed= 60) & (Speed_limit=70))
THEN class = not_speeding

With DLG$_{ff}$ we first apply BACONC on each class, the regularities discovered are:

For class = speeding & Siren ∈ {No} :
1<=(Speed–Speed_limit)<=20
For class = not_speeding & Siren ∈ {No} :
-30<=(Speed–Speed_limit)<=0

If we transform the above functions into additional attributes and apply DLG on the data, the class descriptions induced becomes :

IF (71<=Speed<=110) & (60<=Speed_limit<=100) &
(Siren ∈ {No}) & (1<= (Speed–Speed_limit)<=20)
THEN class = speeding
IF ((60<=Speed<=100) & (70<=Speed_limit<=110) &
(Siren ∈ {No}) & (-30<=(Speed–Speed_limit)<= 0)) V
((90<=Speed<=110) & (100<=Speed_limit<=110) &
(Siren ∈ {Yes}) & (-20<=(Speed–Speed_limit)<=10))
THEN class = not_speeding

The Conservative conjunct deletion step improves the descriptions to :

IF (Siren ∈ {No}) & (1<=(Speed–Speed_limit)<=20)
THEN class = speeding
IF (-30 <=(Speed–Speed_limit)<=0) V (Siren ∈ {Yes})
THEN class = not_speeding

Range generalization further refines the rules to :

IF (Siren ∈ {No}) & ((Speed–Speed_limit)>0)
THEN class = speeding.
IF ((Speed–Speed_limit)<=0) V (Siren ∈ {Yes})
THEN class = not_speeding.

DLG$_{ff}$ greatly simplifies the rules by extending the domain model to include the new variable : (Speed–Speed_limit). The five disjuncts produced by DLG are reduced to three. The result is also much more general in that it is able to correctly classify any new case even if it involves a Speed and Speed_limit that did not appear in the training set.

Study 2:

Langley,Simon,Bradshaw and Zytkow(1987) discussed the use of BACON to discover intrinsic properties of materials. For example, consider the following data (from Langley, Simon, Bradshaw and Zytkow ,1987):

weight(W)	volume(V)	composition
55.923	5.326	silver
74.708	7.115	silver
99.561	9.482	silver
121.841	6.313	gold
91.135	4.722	gold
170.168	8.817	gold
57.182	5.016	lead
39.820	3.493	lead
77.828	6.827	lead

BACON would notice that :

composition	weight/volume(W/V)
silver	10.5
gold	19.3
lead	11.4

BACON would then postulate an intrinsic property of the material called d (d=weight/volume), and associate it with the nominal values of composition. Let us insert into the above data set some noise data. Suppose the noise data are :

weight(W)	volume(V)	composition
58.0	3.3	gold
99.0	8.0	lead

BACON's method of determining deviations from constancy is to incorporate a maximum percentage deviation (P) around the mean (M), and require all observations of a numeric term to fall in the interval [M(1-P), M(1+P)] before that numeric term can be qualified as constant (Langley, Simon, Bradshaw and Zykow, 1987). Setting percentage deviation (P) to 0.05 and applying DLG$_{ff}$ on the above data, an additional attribute of W/V is added and the following classification rules are induced :

IF (5.33<=V<=9.48) & (55.92<=W<=99.56) &
(W/V=10.50) THEN class = silver
IF (3.01<=V<=8.82) & (58.0<= W <=170.17)&
(17.58<=W/V <=19.30) THEN class = gold
IF (3.49<=V<=9.0) & (39.82<=W=99.0) &
(11.0<=W/V=12.38) THEN class = lead

The Conservative Conjunct Deletion step will further refine the rules to :

IF (W/V = 10.50) THEN class = silver
IF (17.58 <= W/V <= 19.30) THEN class = gold
IF (11.0 <= W/V <= 12.38) THEN class = lead

When compared to BACON, DLG$_{ff}$ can also postulate intrinsic properties when it added the new attribute : W/V. In addition, by assuming a range for the values associated with the intrinsic properties, it can provide robustness against noise data.

With BACON alone, increasing the value of percentage deviation (P) may provide similar robustness to noise data, but that will be at the expense of discovering specific functions.

Study 3:

In this example, we show that for certain types of data, combining discovery learning and classification learning has advantages over the classification method alone, both in reducing the number of disjunctive clauses in the class descriptions and improving the classification of novel instances accuracy. We can illustrate by using hypothetical data generated with a formula: Y=a*X, and arbitrary assign a= 2.0 for class=positive and a= 3.0 for class=negative. We can generate the following test data for x=0.5,1.00...to 50

X	Y	class
0.50	1.00	positive
.	.	.
.	.	.
50.00	100.00	positive
0.50	1.50	negative
.	.	.
.	.	.
50.00	150.00	negative

Using half of the 100 cases (selected randomly) in the data as training cases and the remaining half as novel instances for testing the accuracy of the classification procedure, we found that with DLG_{ff}, the combined technique, there are only 2 disjunctive clauses in the class descriptions and the accuracy is 100%. Whereas when using classification learning alone, there are 18 disjunctive clauses, and the accuracy is 82%. In this example, we can also observe that by setting a= 2.0 for "positive" and a= 3.0 for "negative"; the range of Y values of positive instances is 1.0 to 100.0 and that of negative is 1.5 to 150.0. In other words, some Y values of positive instances overlap with that of the negative. In fact, 82% of the values of Y across both classes fall within one common range. This example suggests that for cases where some hidden relationship between attributes underlies the classification and there is overlap in the attribute values across different classes, DLG_{ff} has significant advantages.

8. Further research :

8.1 Using function finding and long term memory to handle uncovered cases:

It is not uncommon that training examples are initially classified according some easily observable attributes. Since these observable attributes can achieve the classification objective, numeric relations between attributes found by BACONC may become redundant and be deleted during the conservative conjunct deletion step in DLG_{ff}. If these redundant attributes are stored but suppressed in long term memory instead of being deleted, they may be retrieved later to help classify uncovered novel cases as well as to learn incrementally. A few systems, such as Zhou's (1990) cumulative learning and Michalski, Mozetic, Hong and Lavrac's (1986) AQ15, had been reported to use computer secondary memory as perfect long term memory. Information stored in long term memory can be retrieved later to help incremental learning and noise handling. In DLG_{ff}, before the conservative conjunct deletion step, all class descriptions can be stored in long term memory. Conjunctive clauses made redundant by the conservative conjunct deletion step can be flagged and suppressed. In classifying novel instances, if the instance is uncovered by the class description, clauses suppressed in long term memory may be retrieved to assist the classification. To illustrate, suppose we have the following examples:

Colour	X	Y	Class
blue	2	4	positive
blue	3	6	positive
red	5	10	positive
brown	2	8	negative
brown	9	1	negative

Regularity discovered by BACONC of DLG_{ff} is:
for class=positive, Y=2*X
Initial class descriptions by DLG_{ff} is:
IF(colour∈{blue, red})&(2<=X<=5)&(4<=Y<=10)&(Y=2*X)
THEN class = positive
IF(colour ∈{brown})&(2<=X<=9)&(1<=Y<=8)&(Y≠2*X)
THEN class =negative

After conservative conjunct deletion, class description becomes :
IF colour ∈ {blue, red } THEN class = positive
IF colour ∈ {brown} THEN class = negative
If, for example, we have a novel case : colour = orange, X = 8, Y = 16. With the current class description, the case is uncovered. But with the initial class description retrieved from long term memory , then based on the clause : Y=2*X, the novel case can be classified as positive.

8.2 Incremental learning in function incorporated classification learning:

Incremental learning refers to modifying current concepts to accommodate new learning events. An incremental method should be able to specialize a concept so that it no longer covers a negative event and generalize a concept so that it covers a new positive event. If new events cannot be accommodated by modifying the current concepts or when there is noise detected, then, all events and concepts are re-examined. In this paper, we shall discuss incremental learning of function derived attributes.

By noting the number (N), mean and range of attributes of examples of each class learned, function derived attributes can be generalized or specialized to learn incrementally. Generalization of functions in incremental learning can take two forms :
(1) Generalizing a specific function : To accommodate new positive examples, specific functions (e.g. Y=2*X) may be generalized to a function with a constant interval (e.g. 1.9<=Y/X<=2.0), as long as the interval does not overlap with that of the negative class. (2) Extending a function range : To accommodate new positive examples, the constant range of a function may be extended (e.g. from 1.9<=Y/X<= 2.0 to 1.9<=Y/X<=2.2), if it does not overlap with that of the negative class.

Specializing a function in incremental learning can also take two forms:

(1) Reducing a function range: Suppose we have an initial class description of :
For class= positive : 2<=(y–x)<=20;
For class= negative: -30<=(y–x)<=-2;
Suppose, with range generalization, the class description was generalized to :
For class= positive : 0<(y–x) ;
For class= negative: (y–x)<0;
To accommodate a new negative example of say, y = 4, x = 3; the class description may be modified to:
For class= positive:1<(y–x);
For class = negative: (y–x)<= 1.

(2) Specializing a function can also occur when new positive examples shift the mean of the function constant term and thus the interval required for constancy.
Consider the following class description:
IF (0.92<=Y/X^2<=1.0)
THEN class = positive (N=4 , mean of Y/X^2=0.98)
IF (3.2<=Y/X^2<=5.3)
THEN class = negative (N=10, mean of Y/X^2=4.3)
With new positive examples of , say, : X= 6, Y=34 and X=8, Y=60, the values of N and Y/X^2 are updated to : N=6 , mean of Y/X^2 = 0.96698.
With mean changed to 0.96698, the required range for constancy (with percentage deviation(P) =0.05) is 0.91863 to 1.01533. The class description can then be specialized to :
IF (Y/X^2=0.96698)
THEN class = positive (N=6, range of Y/X^2 = 0.92 to 1.0)
IF (Y/X^2≠0.96698)
THEN class = negative (N=10, range of Y/X^2 =3.2 to 5.3)

It is obvious that, with incremental learning, functions are more likely to be generalized than specialized. Should the rules fail to accommodate new examples in incremental learning, all examples and rules can be retrieved from secondary memory for re-learning .

9. Conclusion :

In this paper, we have presented an algorithm (DLG$_{ff}$) based on combining the merits of classification and discovery learning. Our objective is to incorporate function-finding features in classification learning. Existing classification learning algorithms, such as DLG, derive classification procedures based on values of single attributes. Intuitively, incorporating functions between attributes should enhance class descriptions and thus improve classification procedures derived by data-driven methods, by reducing the number of disjunctive sets in the descriptions, classifying cases which would otherwise be uncovered and improving the accuracy of classifying novel instances.

Evaluation of DLG$_{ff}$ showed that the objectives are met for data of different classes with overlapping ranges in their attribute values. Given such data, the individual attributes cannot be used effectively as the basis to derive classification procedures. Functions characteristic of each class, if any, will then be crucial in deriving classification procedures. Thus, a combination of discovery and classification learning can improve upon existing classification learning methods. An important contribution of BACON.4 is its ability to postulate intrinsic properties characteristic of different types or class of objects. When we regard the nominal values of the independent variable as representing different classes of objects and apply the function finding-classification procedure, the combined algorithm proved to be an extension of the BACON effort, in that it renders robustness by providing a range for the values of the intrinsic properties. In the absence of complete data, the presence of noise data and the need for approximation of non-linear functions and/or functions which are too complex to be discovered by BACON, the combined algorithm, which allows a range for a function constant term, can improve upon existing function finding systems. Further research can focus on the incremental and cumulative learning aspect of function finding and real world applications.

References :

Elio, R. & Watanabe, L. (1991) An incremental deductive strategy for controlling constructive induction in learning from examples . *Machine Learning*. 7,7-44.

Falkenhainer, B.C. & Michalski, R. (1986) Integrating Quantitative and Qualitative discovery: The ABACUS system. *Machine Learning* 1,367-401.

Langley, P., Bradshaw, G.L. & Simon, H.A. (1983) Rediscovering chemistry with the BACON system. In Michalski, R.S. , Carbonell, J.G. and Mitchell,T.M. (eds.) *Machine learning: An artificial intelligence approach,* Tioga: Palo Alto,CA. 307-330 .

Langley, P., Zytkow, J.M., Simon H.A. & Bradshaw, G.L. (1984) The search for regularity: four aspects of scientific discovery. In Michalski, R.S. , Carbonell, J.G. and Mitchell,T.M. (eds.) *Machine learning: An artificial intelligence approach,* Springer-Verlag: Berlin. 425- 469.

Langley, P., Simon, H.A., Bradshaw, G.L & Zytkow, J.M. (1987) *Scientific Discovery: Computational exploration of the creative process*. MIT press :Cambridge.

Langley, P, Zytkow, J.M. (1989) Data driven approaches to empirical discovery. *Artificial Intelligence*, 40, 283 - 312.

Michalski, R.S.(1980) Pattern recognition as rule-guided inductive inference. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 2, 349-361.

Michalski, R.S.(1984) A theory and methodology of inductive learning. In R.S. Michalski, J.G. Carbonell & T.M. Mitchell (eds.), *Machine Learning: An artificial intelligence approach*, Springer-Verlag, Berlin, 83-129

Michalski, R.S. Mozetic, I., Hong, J & Lavrac, N. (1986) The multiple purpose incremental learning system AQ15 and its testing and application to three medical domains. *Proceedings* *of the fifth National Conference on Artificial Intelligence* (1041 - 1045) Philadelphia,P.A. Morgan Kaufman.

Plotkin, G.D. (1970) A note on inductive generalization. In B. Meltzer & D. Mitchie (eds.) *Machine Intelligence* ,5, Edinburgh University Press, Edinburgh, 153-163.

Plotkin, G.D. (1971) A further note on inductive generalization. In B.Meltzer & D. Mitchie (eds.) *Machine intelligence* 6, Edinburgh University press, Edinburgh, 101-124.

Quinlan, R.(1986) Induction of decision trees. *Machine learning* 1:81-106.

Webb, G. (1991a) Learning disjunctive characteristic descriptions by least generalization, Technical report 2/91, Deakin University,Geelong 3217

Webb, G. (1991b) Einstein:An interactive inductive knowledge acquisition tool. *Proceedings of the 1991 Knowledge Acquisition for Knowledge-based System Workshop*,Banff.

Zhou, H.H.(1990) CSM: A computational model of cumulative learning *Machine learning*, 5