Computación y Sistemas Vol. 1 No. 3 pp. 136-144
© 1998, CIC - IPN. *ISSN 1405-5546* Impreso en México

Experiments with Domain Knowledge in Unsupervised Learning: Using and Revising Theories

J. Béjar & U. Cortés
Departament de Llenguatges i Sistemes Informàtics (IA)
Universitat Politècnica de Catalunya
Jordi Girona Salgado 1-3 Barcelona 08034. Spain.
phone: +34 3 4017016 - Fax: +34 3 4017014
{bejar,ia}@lsi.upc.es

Article received on November, 1997; accepted on January 20, 1998.

Abstract

Using domain knowledge in unsupervised learning has shown to be a useful strategy when the set of examples of a given domain has not an evident structure or presents some level of noise. This background knowledge can be expressed as a set of classification rules and introduced as a semantic bias during the learning process.

In this work we present some experiments on the use of partial domain knowledge with the tool **LINNEO$^+$**, *a conceptual clustering algorithm. The domain knowledge (or domain theory) is used to select a set of examples that will be used to start the learning process, this knowledge has neither to be complete nor consistent. This bias will increase the quality of the final groups and reduce the effect of the order of the examples. Some measures of stability of classification are used.*

The improvement of the concepts can be used to enhance and correct the domain knowledge. A set of heuristics to revise the original domain theory has been experimented, yielding to some interesting results.

Keywords: Knowledge Acquisition, Domain Theory, Ill–Structured Domains, Clustering Methods.

1 Introduction

The use of unsupervised learning to discover useful concepts in sets of non classified examples allow to ease the labour of rule construction for knowledge bases. Tools that helps to this labour and that increase the quality of the knowledge obtained are very desirable specially if the domain has problems of ambiguity, lack of a clear structure, use of qualitative and quantitative knowledge or lack of a broad consensus between experts.

In this work we present the methodology used by **LINNEO$^+$** [Béjar, 1995; Béjar *et al.*, 1997; Sánchez *et al.*, 1997], that has been extended to use domain knowledge in order to *semantically bias* a conceptual clustering algorithm [Michalski & Steep, 1983; Fisher, 1987]. This knowledge helps to obtain more stable classifications and more meaningful concepts from unclassified observations.

It is shown, also, that little knowledge can produce considerable gain, despite of the ambiguity or the partial incorrectness of the knowledge. This ambiguity can be also solved using the improved classifications, performing specializations or generalizations that correct the problems on the domain knowledge.

This paper is organized as follows: section 2 is devoted to give some basic notions about **LINNEO$^+$**. In §3 we give a detailed description of the concept of domain theory and its use in the classification process, §4 exposes an example to illustrate the effects of the bias in the quality of the results. §5 describes the heuristics used to revise the domain theory using the improved classification, §6 shows an example of theory revision and compares it with the previous results. In §7 some conclusions are given.

2 LINNEO$^+$

LINNEO$^+$ is a knowledge acquisition tool oriented to ill-structured domains, domains with a weak structure, imprecise information and a membership function of the observations to the concepts. It uses an unsupervised learning strategy and incrementally accepts a *stream* of observations, trying to discover a classification scheme from the data. As a control strategy it retains only the *best* hypotheses that are consistent with the observations given a similarity criterion. Part of **LINNEO$^+$** could be considered as conceptual clustering method with two tasks critically important for its performance: **clustering,** which determines useful subsets of a dataset; and **characterization,** which determines a concept for each extensionally defined set discovered by clustering. The final characterization of the classes is build-up by **GAR** [Riaño, 1994] as a rule set. This task needs from an expert to accept (or reject) the resulting clusters. Other modules try to better exploit observational knowledge from the dataset, or take advantage of the experts knowledge if available.

The expert has to define a set of observations that he thinks as sufficient to model the domain and also defines a set of attributes relevant to the classification goal intended. The expert is allowed to represent attributes by means of two predefined types: **Quantitative**, **Qualitative**. For each observation, a vector is defined whose length (n) is just the number of attributes. Classically, this vector is the value vector for each property of the object, then the usual representation of an object is:

$$O_i = ((Attribute_k, Value_k)^+)$$

In **LINNEO$^+$** this representation has evolved to a more expressive one:

$$O_i = ((Attribute_k, Value_k, Status_k)^+)$$

where *Status* could adopt one among the following values [*Missing*, *Nought*, *Illegal*, *Acceptable*]. The *status* denotes implicit information about the value of an attribute, for a given object, the idea is to exploit this additional information. Also we maintain the representation (*Attribute.Value*) because it is more compact. So, when the status of an attribute is *Acceptable*, that means that for a given $Attribute_k$ its $Value_k$ belongs to the range, otherwise the *Value* is just its *Status*.

Missing has the traditional interpretation. Two alternative strategies are used in order to substitute the missing value. The first consists on assigning a value for that *Attribute* using the mean of the values or the value with greater frequency in its column, it is an *a priori* approach. The second, tries to induce the values after a classification excluding the objects with missing values, it is an *a posteriori* approach. The values of the classes that are more similar to the objects with the missing values are used. **LINNEO$^+$** uses the second approach. When *Nought* appears as status, it means that for this object the value of that attribute is *irrelevant*, and a special treatment is given while the classification process is running.

When the status is *Illegal*, the interpretation is more complex, because it means that this object can not have an *Acceptable* value (nor other status) for this $Attribute_k$, because there exists an structural or causal dependence with some $Attribute_w$, which absence or presence forbids $Attribute_k$ to have a meaningful value. This information can be viewed as a part of the domain theory and it is treated after the classification step [Béjar, 1995].

Once the expert has selected the attributes and the observations sample, the classification process starts. This process induces a tentative conceptual structure for the domain assuming that all the available information is present in the dataset. In general, any inductive classification process will group *objects* into *classes* using some criterion of similarity. We decided to use the classical concept of distance.

The numerical values are normalized to the interval $[0, 1]$ in order to avoid the influence of the different scales. The distance that will be used, in the example, for determining the similarity between two objects, O_i and O_j, is the generalized Hamming distance:

$$d(O_i, O_j) = \sum_{k=1}^{n} (diff(O_{ik}, O_{jk})) \quad (1)$$

where $diff(O_{ik}, O_{jk})$ is, for a *qualitative* attribute, 1 if O_{ik} and O_{jk} are different modalities and 0 otherwise, and for a *quantitative* attribute, it is the absolute value of their difference.

This similarity measure is very simple to compute and helps to find easily a preliminary structure. Each attribute has the same contribution to the similarity (in the interval $[0, 1]$), so, this allows to mix the two kind of attributes and to study the influence that have in the description of the data.

The center of a $class_i$ is obtained by calculating the mean value for each quantitative attribute of every object. For qualitative attributes, the center includes each one of its modalities with its corresponding occurrence frequency. Note that the center of a class is considered as the *prototype* of the objects contained in the class. The distance between an object and a class prototype

can be taken as the inverse of the degree of membership of the object to the class.

The aggregation algorithm builds clusters of similar objects given a initial parameter that we call *radius* that selects the level of generality of the induced concepts. A scheme of the algorithm is:

1) Use the first object of the dataset to generate the first class.

2) For each one of the remaining objects in the dataset, the *best* class among the current ones is selected. The best class for an object is the one in the previous set of classes with minimum distance to the object. Two things can happen at this moment:

   a) Distance to the best class is less than the current classification radius. In such case, the object is included in the class and the center, of class is recalculated. While recalculating the center some objects may escape from the center and locate themselves farther than the radius. If this happens, these objects are eliminated from the class and marked as not classified in the first step.

   b) Distance to the best class is greater than the current classification radius or there is not a best class. In such case a new class is created. The vector of the object currently under consideration becomes the initial center of the class.

3) The objects marked as not classified in the first step are now reclassified but this time without modifying the centers to avoid endless recalculation.

The result, once it has been confirmed by the expert, is a list of classes. Two aspects are taken into account when representing a class: its extensional description and its intensional description. The first is given by the enumeration of all elements contained in the class. The second one is a vector containing n attributes representing the class center.

The incrementality of the algorithm has the effect that the results depend on the order of presentation of the observations [Fisher *et al.*, 1992]. Some syntactical heuristics has been developed to reduce this effect called *not-yet* heuristics [Roure, 1994; Béjar, 1995].

This methodology has been successfully applied to some real domains as mental illnesses [Rojo, 1993], marine sponge classification [Béjar, 1995] and fault diagnose in wastewater treatment plants [Sànchez *et al.*, 1997]

3 Using a Domain Theory

In this section we introduce the concept of domain theory (DT) that expresses what the expert can explain about the domain that he is defining. We will describe the syntax used by the expert in order to define a domain theory and the role that this knowledge plays in the classification process.

3.1 Defining a Domain Theory

In unsupervised learning, and specially in ill-structured domains, the description of the observations is not enough usually to build up a set of concepts. The noise of the observations, the existence of irrelevant descriptors, or the non homogeneity of the sampling of observations can deviate the learning process from a meaningful result. It is desirable, thus, a *guide* from a higher level of knowledge to assure the success of the acquisition.

In our methodology, we allow the expert to define as Domain Theory (DT) as a group of constraints guiding the inductive process. Therefore, the DT semantically *biases* the set of possible classes. This DT acts just as a guide; it does not need to be complete. It could be very interesting for the experts to play with several definitions of DT as they could model several levels of *expertise* or to obtain different classifications using different points of view or bias. **LINNEO$^+$** with no DT available acts just as an apprentice with a syntactic heuristic to group objects by their similarity. The expert is allowed to express his DT in terms of rules that determine the definition of a part as the definition of classes he already knows to exist. A rule is composed by a class name (an identifier) and some constraints, a set of conditions that elements must fulfill in order to belong to the class. **LINNEO$^+$** accepts the following syntax to express rules:

$$
\begin{aligned}
&L\_val = Value | (Value^+)\\
&Op = = \mid \texttt{neq}(\neq) \mid > \mid < \mid >= \mid <= \mid \texttt{range}\\
&Clause = (Op\ L\_val\ attribute_i) \mid (\texttt{rel}\ lisp\_exp)\\
&CompClause = Clause \mid (\texttt{or}\ Clause^+)\\
&Rule = (CompClause^+ \implies C_j)
\end{aligned}
$$

"(`rel` *exp*)" stands for a relational expression between attributes expressed in `LISP` syntax, it is not expected that this expression will be complex; *lval* could be a non-null list of modalities in the case of qualitative attributes, and a single value in the case of quantitative attributes; $Attribute_i$ is the target attribute and, C_j is a dummy identifier for the set of objects that satisfies this rule. Each clause in a rule represent a conjunction and the `or` operator allows to express disjunctions. This syntax

could be easily adapted to a given domain, if needed.

3.2 Biasing with a Domain Theory

If the expert is able to build a DT, it is possible to use this knowledge to *bias* the classification using the constraints as a guide to preprocessing the dataset. Even in *ill-domains* the expert knows that to ignore some attributes for certain classes can be useful, because those attributes are not relevant in the prediction of class memberships. In the same way, the expert, knows that there are other attributes, or their conjunction, that could be used with a certain degree of confidence to try to predict class membership. The idea is to create a partition of the dataset using the rules defined by the expert in meaningful parts, the objects with some knowledge about its relation (those described by the rules). Those objects that not fulfill any of the rules are treated as without Domain Theory.

The treatment of the dataset is done previously to the classification as follows. All the objects that satisfies a rule (R_i) are grouped together ($\mathcal{S}_{R_i}$). Sometimes the expert gives more than one rule to constraint a set (or *class*). Sometimes, when rules are too general, two or more rules select the same object, in this case a *special* set is created and the rules pointing to that object are attached. All the objects that do not accomplish any rule are grouped in a *residual* set. After this process is carried out to a maximum, **LINNEO$^+$** generates $r + 2$ sets of objects, where r is the number of sets that the expert has constrained.

In each one of these sets, except for the *special* and the *residual*, **LINNEO$^+$** starts a classification process and eventually, creates at least a class for each one. Then a new process begins with the centers of these classes as seeds of the new classification and the rest of objects. In this process new classes can be formed corresponding to classes not described by rules.

The *bias* is obtained by the reordering and previous grouping of the observations in a meaningful scheme, rather than by the random order of the unbiased process. This yields to a more meaningful set of classes, more in the idea that the expert has of his domain structure. This avoids also the instability induced by the ordering of the observations.

4 Experiments with a Domain Theory

In order to test the effect of a domain theory in the process of classification, we have written a small set of rules for the *Soya bean* domain (See Table 1) to bias the resulting classes. These rules have been built by hand, inspecting the prototypes of the classes of a unbiased classification, and extracting the attributes more distinctive. This set of rules is neither complete nor consistent, because we just want to show that only a small piece of domain knowledge is enough to improve the stability, and therefore the quality, of a classification. These rules select 130 observations from a total of 307.

The experiment was carried out by comparing two sets of 20 random ordered classification using **LINNEO$^+$** of the *Soya bean* dataset [Michalski & Chilausky, 1980] obtained from the UCI Repository of Machine Learning Databases and Domain Theories [Murphy & Aha, 94]. The first without using the domain theory, and the second using our set of rules as domain theory.

In order to compare the resulting classifications we have developed an algorithm that provides a measure of the differences between two classifications [Béjar, 95; Béjar *et al.*, 1997; Faith & Belbin, 1986]. This measure, that we call *structural coincidence*, is used to provide a value for the stability of each set of classifications, as the mean of the difference of each pair of classifications in the set. Among these differences, it is taken into account the coincidence of objects in the same group and the number of classes of each classification.

Another measure of stability that is used in the comparison is based on the coincidence of the pairs of associations of observations between two partitions described in [Faith & Belbin, 1986], this measure decreases with the similarity.

The stability of a classification of the Soya Bean dataset without the DT is 77.6% for the first measure and -1013.4 for the second. The stability using the DT increases to 91% for the first measure and -4285.6 for the second. A cross comparison between the two sets of classification yields a value of 79.9% for the structural coincidence. This value has been calculated comparing each class resulting from each method with all the others and then getting the average. The interpretation of this value is that the classifications using the domain theory are similar to those created without using a *bias* but much more stable.

Another result is that in the set of classification without domain theory the number of classes is inside the

((= (diseased) fruit-pods) (= (colored) fruit-spots) (= (norm) seed) -> frog-eye-leaf-spot) (1)	((= (lt-normal) plant-stand) (= (severe) severity) (= (brown dk-brown-blk) canker-lesion) -> phytophthora-and-rhizoctonia-root-rot)(2)
((= (norm) fruit-pods) (= (tan) canker-lesion) (= (lt-norm norm) precip) -> charcoal-and-brown-stem-rot)(3)	((= (dk-brown-blk) canker-lesion) (= (abnorm) seed) (= (gt-norm) precip) -> anthracnose)(4)
((= (abnorm) seed) (= (tan) canker-lesion) -> purple-seed-stain)(5)	((= (few-present) fruit-pods) -> cyst-nematode(6)
((= (norm) leaves) (= (gt-norm) temp) -> diaporthe-pod-and-stem-blight)(7)	((= (above-sec-nde absent) stem-cankers) (= (brown) canker-lesion) (= (norm) fruit-pods) -> diaporthe-stem-canker-and-brown-spot)(8)
((= (lower-surf) leaf-mild) -> downy-mildew)(9)	((= (upper-surf) leaf-mild) -> powdery-mildew)(10)
((= (no) lodging) (= (w-s-marg no-w-s-marg) leafspots-marg) (= (90-100%) germination) -> brown-stem-rot-and-herbicide-injury) (11)	

Table 1: A Soya Bean Domain Theory

Dataset	Without DT	with DT
Marine Sponges	73.3%	80.9%
Mental Illnesses	74.5%	88.5%
Wastewater	63.7%	69.5%

Table 2: *Structural coincidence* in other datasets

interval 15 to 21, however using the rules the number of classes is inside the interval 15 to 19, both with a mean of 18 classes.

Applying this technique to other datasets yields similar results as can be seen on Table 2 [Béjar, 1995].

In the light of these results, we can say that the use of domain knowledge in unsupervised learning reduces the problem of obtaining meaningless groupings and also it reduces the instability induced by an improper input order.

5 Domain Theory revision

Due that the domain theory that the expert gives for the *biasing* process could be inconsistent or incomplete, it is worth to try to improve it in some automatic way. Some systems in machine learning try to improve incomplete or incorrect domain theories using labeled examples in order to fix the errors [Ourston & Mooney, 1993]. Our system is unsupervised, so we have to trust the classes formed in the classification process and the source of detected errors can only be the use of the DT previous to the classification.

We have been experimenting with some heuristics for theory revision. These heuristics are very conservative and do not change a lot of conditions in the rule, they only try to discover the minimum set of changes that improves the selectivity of the rule and that are consistent with the formed groups. The heuristics can only revise the clauses applied to one attribute with the operators $=$, $\neq$, $>$, $<$ and **range**, and not with **rel** expressions (see 3.1).

This revision has two parts. When a dataset is classified using the domain theory, two classes of rules may appear if we observe the consistency of the resulting partitions. There is a set of rules that selects a definite set of objects that no other rule selects, we call this set *non collision rules*. There is another set of rules whose sets of objects intersect among them. These are *ambiguous* rules and the multiple selected objects cannot be assigned to a definite set. So, the revision can be done separately for each set of rules. First, to improve the non collision rules trying to generalize them or by deleting superfluous conditions. Second, to correct the ambiguous rules, trying to specialize them in order that no object is selected by more than one rule. A more extended description of this process can be found in [Béjar, 1995]

5.1 Revision of Non Collision Rules

These rules can be treated separately, because all of them have their own set of examples, classified in one or more groups. The objective of this process is to fit the rules with the groups but not that of selecting objects from other groups.

This improvement has two phases. Firstly, the phase of specializing. Some rules can have an extension so broad, or excessive disjunctive conditions, that can prevent a later generalization. So, some of these conditions can be restricted or dropped in order to be consistent with the values of the objects in the classes selected by the rules. This can be done for example by eliminating modalities that do not appear in the values of a class from an equal (=) clause, or to restrict the < and > clauses to the upper and lower bound of the attribute in the prototype of the class, respectively.

The second phase is generalizing. Not all the objects from a class are selected by the rules that had generated it. It is desirable that the rules cover the maximum number of objects of this class. A way to achieve this is to generalize the conditions extending its ranges or dropping conjunctions, only if these changes are consistent with the rest of classes of the dataset. This generalization can be done for example by introducing more modalities in a equal (=) condition, modalities that appear in the class and have not been used by the expert or to change the clause to a **range** clause with the bounds of the attribute in the prototype. Also, it is possible to test the effect of eliminate each one of the conditions of the rule.

This process can broad the extent of the rules consistently with the classes formed. The corrected rules can help the expert to refine his knowledge.

5.2 Revision of Ambiguous Rules

This set corresponds to rules too general, or to classes where the expert cannot differentiate accurately. To treat these rules it is necessary to calculate what groups of rules are in conflict and what objects are the conflictive.

As information to correct those rules, it is taken into account the classes that group the conflictive objects and the rule that has formed this group. It is a logical assumption that of assigning the conflictive objects to the rule that has formed the class the objects belong to.

The objective is to specialize each rule that has some conflicts using as constraints the observations that it has not to select. The way to do this is to specialize the rule constraining its conditions or adding new conditions that exclude the conflicting observations.

The selection of the conditions is done by selecting the attributes of the classes (of the rule) that have values not present in the non desired observations. With these attributes, it is possible to construct new clauses in order to specialize the rule. If the attribute is quantitative, a clause that selects only the values between the range present in the classes can be constructed. If the attribute is qualitative, a clause that test that the modalities are only those present in the classes, can be constructed.

The specialization process is done by selecting some of the candidate clauses. Each clause is tested with the clauses from the rule. The clauses selected are those that reduce the most the selection of non desirable observations and maintain the selection of correct observations.

This is a very conservative heuristic, because prefers to maintain the selection of some non desirable observation if an excessive specialization excludes some of the correctly selected observations.

After the specialization process, a generalization can be done testing if some of the original conditions of the rule are now unnecessary due to the new added conditions.

Future studies will include the creation of multiple rules starting with the original rule as a common prefix, or the over specialization of a rule and *a posteriori* generalization that allows to regain observations.

This heuristics can be used iteratively, reclassifying the observations with the corrected rules in order to make use of the improved domain theory and to try to correct the rules within the new classification process. The process converges when no rule is corrected.

6 Evaluating the Revised Domain Theory

The same dataset has been used in order to evaluate the heuristics for correcting rules, but and artificially ambiguous domain theory has been constructed (See Table 3). There are, in this case, 12 rules (some of them grouping more than one category) that select 178 objects (33 in the special class) from a total of 307.

Concretely, the rule number 9 has the following collisions: rule 1 (6 observations), rule 2 (2 observations), rule 5 (2 observations), rule 6 (1 observation), rule 10 (5 observations), rule 11 (3 observations), rule 12 (4 observations); the rule number 8 has the following collisions: rule 3 (8 observations), rule 10 (2 observations). The total number of conflicting objects is 33.

After the correcting process, the number of collisions has been reduce to 7 objects, specializing 3 rules as can be seen in Table 4.

((= (diseased) fruit-pods) (= (colored) fruit-spots) (= (pot-severe) severity) -> frog-eye-leaf-spot) (1)	((= (lt-normal) plant-stand) (= (severe) severity) (= (none) int-discolor) -> phytophthora-and-rhizoctonia-root-rot)(2)
((= (norm) fruit-pods) (= (tan) canker-lesion) (= (lt-norm norm) precip) -> charcoal-and-brown-stem-rot)(3)	((= (dk-brown-blk) canker-lesion) (= (abnorm) seed) (= (gt-norm) precip) -> anthracnose)(4)
((= (abnorm) seed) (= (tan) canker-lesion) -> purple-seed-stain)(5)	((= (norm) leaves) (= (gt-norm) temp) -> diaporthe-pod-and-stem-blight)(6)
((= (few-present) fruit-pods) -> cyst-nematode(7))	((= (no) lodging) (= (tan) canker-lesion) -> herbicide-injury)(8)
((= (lt-80%) germination) (= (norm) plant-growth) -> bacterial-pustule)(9)	((= (above-sec-nde absent) stem-cankers) (= (brown) canker-lesion) (= (norm) fruit-pods) -> diaporthe-stem-canker-and-brown-spot)(10)
((= (lower-surf) leaf-mild) -> downy-mildew)(11)	((= (upper-surf) leaf-mild) -> powdery-mildew)(12)

Table 3: An Ambiguous Soya Bean Domain Theory

((= (no) lodging) (= (w-s-marg no-w-s-marg) leafspots-marg) (= (90-100%) germination) -> herbicide-injury)
((= (lt-normal) plant-stand) (= (severe) severity) (= (brown dk-brown-blk) canker-lesion) -> phytophthora-and-rhizoctonia-root-rot)
((= (lt-80%) germination) (= (norm) plant-growth) (= (absent) leaf-mild) (= (absent brown-w/blk-specks) fruit-spots) (= (norm) seed-size) -> bacterial-pustule)

Table 4: The Corrected Rules

Rule	Before	After	Rule	Before	After
Num 1	18	24	Num 2	17	17
Num 3	10	17	Num 4	31	31
Num 5	8	8	Num 6	5	6
Num 7	6	6	Num 8	2	2
Num 9	41	41	Num 10	11	14
Num 11	7	10	Num 12	6	10

Table 5: Number of Objects Selected

The number of objects selected by each rule after and before the correction can be seen in Table 5.

The structural coincidence with the ambiguous rules is 88.9% With the corrected rules it has been increased slightly. The value for the structural coincidence has been increased to 89.6%. It is not expected a great increase of stability because the number of selected objects by the domain theory has not been increased, but the gain of stability is maintained.

Applying this technique to other datasets with expert build domain theories yields to similar results, a slightly increase of stability is obtained, but the selectivity of the rules is increased [Béjar, 1995].

7 Conclusions

It has been shown that the use of domain knowledge as semantic bias in a unsupervised learning algorithm increases the quality of the result although. The domain knowledge does not need to be perfect, and can have some ambiguities or inconsistencies, an increase of stability of the results could still be achieved.

This knowledge aids also to cope with the ordering effect that suffer all incremental algorithms, biasing the process towards a meaningful result, obtaining a better set of concepts.

The fix and revision of the domain knowledge can also be done, obtaining a benefit from the better classification. Ambiguities can be detected and corrected observing the nature of the obtained groups and generalizing and specializing the knowledge in order to fit the description of the concepts.

7.1 Future Work

More extensive studies about heuristics to correct domain theories have to be done. Less conservative methods have to be evaluated in order to obtain a better set of new rules.

Some studies about how to generate new rules starting with the original rules specializing in order to fit them to

each of the different classes that can generate have also to be done. It is possible to generate one rule for each class with the original rule as a prefix. This can build a hierarchy of rules that can ease the use of this methodology in order to build knowledge bases for Knowledge-Based systems.

Information about relevance of attributes and causal structure can aid also to guide the process of selecting the adequate attributes to specialize rules.

Acknowledgments

This work is funded by Spanish Research Council (CICyT) TIC96-0878.

References

Béjar, J. *Adquisición de Conocimiento en Dominios Poco Estructurados.* Ph.D. thesis, Departament de Llenguatges i Sistemes Informàtics. Facultat d'Informática de Barcelona. Universitat Politécnica de Catalunya, 1995.

Béjar, J., U. Cortés, and R. Sangüesa. "Experiments with Domain Knowledge in Knowledge Discovery." In em Proceedings of the 1^{st} International Conference on the Practical Application of Knowledge Discovery and Data Mining, London, 1997

Faith, D. and L. Belbin. "Comparison of Classifications Using Measures Intermediate between Metric Dissimilarity and Consensus Similarity." *Journal of Classification*, 3:257–280, 1986.

Fisher, D. "Knowledge Acquisition Via Incremental Conceptual Clustering." *Machine Learning*, 2:139–172, 1987.

Fisher, D., L. Xu, and N. Zard. "Ordering Effects in Clustering. In *Proceedings of the Ninth International Workshop on Machine Learning*, pages 163–168, 1992.

Jain, A.K. and R. C. Dubes. *Algorithms for Clustering Data.* Prentice Hall, 1989.

Michalski, R. and R. E. Steep. "Learning from Observation: Conceptual Clustering," Vol. 2 of *Machine Learning an A.I. Perspective*, chapter 11, pages 331–363. Ed. Tioga, Palo Alto, California, 1983.

Michalski, R.S. and R. L. Chilausky. "Learning by Being Told and Learning from Examples: An Experimental Comparison of the Two Methods of Knowledge Acquisition in the Context of Developing an Expert System for Soybean Disease Diagnosis." *International Journal of Policy Analysis and Information Systems*, 4(2), 1980.

Murphy, P.M. and D. W. Aha. *UCI Repository of Machine Learning Databases.* Technical report, Irvine, CA: University of California, Department of Information and Computer Science, 1994.

Ourston, D. and R. J. Mooney. "Theory Refinement Combining Analytical and Empirical Methods." *Artificial Intelligence*, 1993.

Riaño, D. *Automatic Knowledge Generation from Data in Classification Domains.* Master Thesis, Facultat D'Informàtica de Barcelona. Unversistat Politécnica de Catalunya, 1994.

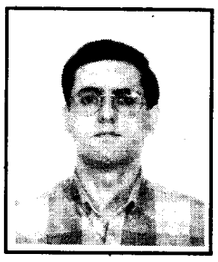
Rojo, E. *Aplicación del Software LINNEO a la Clasificación de Transtornos Mentales.* Ph.D. Thesis, Divisió de Ciencies de la Salut. Facultat de Medicina. Universitat de Barcelona, 1993.

Roure, J. *Study of Methods and Heuristics to Improve the Fuzzy Classifications of LINNEO$^+$.* Master Thesis, Facultat d'Informática de Barcelona Universitat Politécnica de Catalunya, 1994.

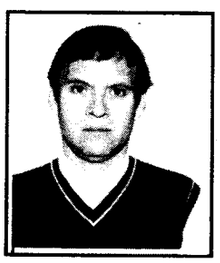
Sànchez, M., U. Cortés, J. Béjar, J. de Gracia, J. Lafuente, and M. Poch. "Concept Formation in WWWTP by Means of Classification Techniques: A Compared Study." *Applied Intelligence* 7, pages 147-165, 1997.

Serra, P., M. Sànchez, J. Lafuente, U. Cortés, and M. Poch. "DEPUR: A Knowledge-Based Tool for Waste Water Treatment Plants." *Engineering Applications of Artificial Intelligence*, 7(1):23–30, 1994.

Serra, P., M. Sànchez, J. Lafuente, U. Cortés, and M. Poch. "ISCWAP: A Knowledge-Based System for Supervising Activated Sludge Processes." *Computers and Chemical Engineering*, 1996. (Accepted).



Javier Béjar *is an Associate professor at the Languages and Informatic Systems Department of the Politechnic University of Catalunia, Spain.*



Ulises Cortés *recived a B.Sc. in Industrial and Systems Engineering from the Instituto Tecnológico de Estudios Superiores de Monterrey (ITESM) in 1982, and a Ph. D. from the Universitat Politécnica de Catalunya (UPC) in 1984. He is an Associate Professor in the Software Department of the UPC since 1988. Head of the Artificial Intelligence Ph. D. program and Vice-dean of the International Affairs of the Faculty of Computer Science of Barcelona (FIB). He is member of the AEPIA (Spanish Association for Artificial Intelligence). He is pioneer member of ACIA (Catalan Association of Artificial Intelligence). He is member of SMIA (Mexican Society of Artificial Intelligence). He has an extensive list of publications, conference papers, and tutorial and supervisory contributions. His main research topics are machine learning, knowledge acquisition an LISP-like languages.*