MACHINE LEARNING FOR MAP INTERPRETATION: AN INTELLIGENT TOOL FOR ENVIRONMENTAL PLANNING

FLORIANA ESPOSITO, ANTONIETTA LANZA, DONATO MALERBA, and GIOVANNI SEMERARO
Dipartimento di Informatica, Università degli Studi di Bari, Bari, Italy

The design of a user interface integrating instruments for visual and textual representation and image interpretation is a relevant problem when developing an advisory system for environmental planning. Indeed, the user of the system needs a support to the interpretation of maps, that is, a tool that segments maps and automatically associates geometric regions on a map with those semantic labels useful for applying hints and advices suggested by the environmental planning system. In the article, we present the application of symbolic machine learning techniques to the interpretation of maps. Two inductive learning systems, namely, INDUBI/CSL and ATRE, have been used to complete the knowledge base of an expert system for environmental planning. The application described concerns the recognition of four environmental concepts that are relevant for environmental protection. The positive results obtained in two different experiments prove the strength of the adopted approach for the interpretation task.

A relevant problem when developing an advisory system for environmental planning is the design of a user interface integrating instruments for visual and textual representation. In this case, it is difficult to separate the "interface" and the "application," and the interface must be considered in terms of integrative perspectives (Kuutti & Bannon, 1993). From the user's point of view, the interface *is* the system, that is, the way the user's work is supported by the system. At the conceptual level the interface is that part of the system that allows the user to utilize the system meaningfully for some purpose in a definite situation. In this kind of system, where massive techniques of visualization and interpretation are required, the majority of the application belongs to the interface. It is necessary to develop a tool for image interpretation, aiming at a content-based retrieval of maps.

The authors are grateful to Dino Borri, Angela Barbanente, and Mauro Iacoviello of the Department of Town Planning, Polytechnic of Bari, for their help in the individuation and description of the environmental classes. We also wish to thank the IGM for providing the information and the vectorized map extracts, which made it possible to set up the preprocessing algorithms. Many thanks to Lynn Rudd, who revised the first draft of the paper.

Address correspondence to Floriana Esposito, Dipartimento di Informatica, Università degli Studi di Bari, Via E. Orabona 4, 70126 Bari, Italy. E-mail: esposito@lacam.uniba.it

The application should not only display maps but, for example, should enable the user to identify morphological elements characterizing a certain landscape, to individuate some human artifacts as sources of changes in the environment, and to concentrate on the most suitable areas for the application of the planning rules. Indeed, the use of the decision support facilities (notices, laws, methods, and models for urban and environmental planning) is through the semantic indexing of the maps stored in a digital video library. This means to automatically associate geometric regions on a map with those semantic labels useful for applying hints and advices suggested by the environmental planning system. So, the maps must be segmented, indexed, searched, manipulated, and presented according to their contents, that is, they must be interpreted. Any attached textual information, for example, from the thematic charts, must also be treated because it is useful for improving the query capabilities, for quickly filtering video and locating potential areas of interest.

We are aware of the application of a similar approach to the design of human-machine interfaces for map conversion systems, which aim at acquiring paper-based maps and transforming them into vector representations (Quek & Petro, 1993). In our application the maps, already digitized, must be processed and interpreted in order to be handled by an expert system for environmental planning (Esposito & Lanza, 1996).

Symbolic machine learning techniques have been widely applied to automatically acquire the knowledge bases of expert systems (Kodratoff et al., 1994). In the article, these techniques are used to generalize the classification rules for recognizing and selecting environmental morphologies from digital maps. The process is organized into acquisition, learning, and recognition phases. First, map-digitized data are converted into elementary units or cells whose dimension and shape can vary, depending on the specific classification aims; then spatial and structural features are extracted and expressed in a symbolic form. Before starting the learning process, the trainer has to supply a number of instances of the concepts he/she wants to recognize in the maps. The learning process is performed by a learning system able to induce dependent concepts expressed as definite clauses. The interpretation task is performed thanks to the inferred rules that allow the recognition of the morphological patterns. The performance of the whole process is evaluated in terms of both classification rate and semantic content of the acquired concepts.

The advantages of the proposed approach are manifold:

- The use of symbolic descriptions directly generated from maps by a preprocessing unit allows for the integration of the visual information with textual information from different sources.
- The recognition rules can be automatically learned from positive and negative examples and updated.

- The possibility of learning dependent concepts allows the realization of the shift of the language, which is useful in order to overcome problems like zooming and windowing when trying to interpret the same images at different scaling levels.
- The description language adopted for the rules is very comprehensible for experts, so they can control the significance of the acquired concepts for environmental planning tasks.
- The inferred rules are useful not only for classification and prediction tasks, but also for context-based information retrieval, since the concepts allow for the semantic indexing of the map element.

Preliminary results obtained by applying two different inductive learning systems developed at the University of Bari, namely INDUBI/CSL and ATRE, to the problem of map interpretation are discussed below. They confirm the appropriateness of learning techniques for image understanding in order to enhance the characteristics of man-machine interfaces.

INDUBI/CSL

INDUBI/CSL is an empirical learning system of the family of INDUBI systems (Esposito et al., 1994a). It has been developed in C language.

The representation language adopted by INDUBI/CSL is an evolution of the logic language VL_{21} (Michalski, 1980). Indeed, both examples and hypotheses are expressed as VL_{21} *generally* Horn clauses (Grant & Subrahmanian, 1995).

We assume the reader to be familiar with the notions of *substitution, positive* and *negative literal, fact,* and *Horn clause* (Lloyd, 1987). A clause $C = l_1 \vee l_2 \vee ... \vee l_n$ is considered as the set of its literals, that is, $C = \{l_1, l_2, ..., l_n\}$. Here *head(C)* and *body(C)* denote the set of positive literals in C and the set of negative literals in C, respectively. Furthermore, we will denote with *vars(C), consts(C),* and *terms(C)* the set of the variables, of the constants, and of the terms occurring in C, respectively.

Any two clauses will always be assumed to be variable disjoint. This does not limit the expressiveness of the adopted language, since any two nonvariable disjoint clauses can always be standardized apart.

Specifically, in INDUBI/CSL each example is represented by exactly one *ground-linked range-restricted definite* clause. The definitions of definite clause and range-restricted clause are given by De Raedt (1992). A definition of linked clause (Helft, 1987) follows.

Definition 1 (linked clause)

A definite clause is *linked* if all of its literals are linked. A literal is linked if at least one of its arguments is linked. An argument of a literal is linked if either the literal is the head of the clause or another argument in the same literal is linked.

An instance of a linked clause is $C = P(x) \leftarrow Q(x, y), Q(y, z)$. C is linked, since all its literals are linked. Conversely, the clause $D = P(x) \leftarrow Q(y, z)$ is not linked. Indeed, the literal $Q(y, z)$ is not linked in D.

In INDUBI/CSL each hypothesis is a set of VL_{21} linked range-restricted definite clauses with the same head. The main difference between VL_{21} clauses and first-order predicate logic (FOPL) clauses lies in the definition of an atomic formula. Indeed, in VL_{21} the equivalent to the FOPL atom is the notion of *selector,* which is written as

$$[L = R]$$

where L, the *referee,* is a function symbol with its arguments, that is, it is in the form $f(t_1, t_2, ..., t_n)$, and R, the *reference,* is a set of values in the referee's domain.

The semantics of a selector is the following: It is true if L takes one of the values in R. Function symbols of referees are called *descriptors.* They are n-ary typed functions ($n \geq 1$), mapping onto one of three different kinds of domains: *nominal, linear,* and *tree-structured.* A nominal, or *categorical,* domain is a domain in which no relation is imposed on its values. A linear domain is one in which a total order is imposed on its values. Finally, a tree-structured domain is a partially ordered set, whose elements can be represented as nodes of a tree.

In VL_{21}, monadic functions are called *attributes,* while n-adic functions, with $n > 1$, are called *relations.* Moreover, a *predicate* is considered to be a particular function whose nominal domain is {false, true}.

A set of hypotheses, one for each learned concept, forms a *logical theory,* which is constrained to be expressed as a *hierarchical program,* that is, as a logic program for which it is possible to find a *level mapping* (Lloyd, 1987) such that in every VL_{21} clause $[f(t_1, t_2, ..., t_n) = \lambda] \leftarrow L_1, L_2, ..., L_m$, the level of every descriptor occurring in the body of the clause is less than the level of the descriptor f.

An *example E* is *positive* for a hypothesis H if its head has the same descriptor and sign as the head of the clauses in H. An example E is *negative* for H if its head has the same descriptor but opposite sign (the head of a negative example is a negated VL_{21} selector).

Another characteristic of INDUBI/CSL is that it adopts the quasi-ordering induced upon the set of the VL_{21} definite clauses by the notion of θ_{OI}-*subsumption under object identity* or θ_{OI}-*subsumption.*

In the following, we point out the difference between the notions of θ-*subsumption* and θ-*subsumption ordering,* as given by different authors (Chang & Lee, 1973; Muggleton & Feng, 1992; Plotkin, 1970), and those of θ_{OI}-*subsumption* and θ_{OI}-*subsumption ordering,* respectively.

As known in the literature, θ-subsumption induces a quasi-ordering upon the set of clauses, henceforth called θ-subsumption ordering and denoted by $\leq_\theta$, that is, $\leq_\theta$ is reflexive and transitive, but not antisymmetric. Here, $\sim_\theta$ denotes the equivalence relation induced by $\leq_\theta$. It should be noted that it does not coincide with set equality. Indeed, two equivalent clauses under $\sim_\theta$ cannot only be alphabetic variants, but can also have a different number of literals, as for $\{P(x), P(f(y))\}$ and $\{Pf(z))\}$. Thus, as Plotkin (1970) pointed out, there is a reduced member of any equivalence class under $\sim_\theta$ and this member is unique up to an alphabetic variant. Unfortunately, learning systems cannot easily identify such a reduced member during the search for an effective definition of a concept. As a consequence, the adoption of θ-subsumption as a generalization model may cause nontermination of the learning process (Esposito et al., 1994b). These remarks led us to adopt a different generalization model in INDUBI/CSL, based on the notion of θ_{OI}-subsumption. In turn, the definition of θ_{OI}-subsumption (θ-subsumption under object identity) is based on the notion of object identity.

Definition 2 (object identity)

Within a clause, terms denoted with different symbols must be distinct. This notion is the basis for the definition of both an equational theory for clauses and a quasi-ordering upon them.

In FOPL the adoption of the object identity assumption can be viewed as a method for building an equational theory into the ordering, as well as into the inference rules of the calculus (resolution, factorization, and paramodulation) (Plotkin, 1972).

Such equational theory is very simple, since it consists of the following axiom schema, in addition to the set of the axioms of Clark's equality theory (CET) (Lloyd, 1987):

$$t \neq s \in body(C) \qquad \text{for each clause } C \text{ and for all pairs } t, s \text{ of distinct terms that occur in } C \tag{OI}$$

The (OI) axiom can be viewed as an extension of both Reiter's unique-names assumption (Reiter, 1980) and CET's axioms (7), (8), and (9) to the variables of the language.

Under the object identity assumption, the clause

$$C = P(x)\text{:-}Q(x, x), Q(y, a)$$

is an abbreviation for the following clause:

$$C_{OI} = P(x) \text{ :- } Q(x, x), Q(y, a) \parallel [x \neq y], [x \neq a], [y \neq a]$$

where P, Q denote predicate letters, x, y are variables, a is a constant, and the inequations attached to the clause can be seen as constraints on its terms. These constraints are generated in a systematic way by the (OI) axiom. In addition, they can be dealt with in the same way as the other literals in the clause. Therefore, under object identity, any clause C generates a new clause C_{OI} consisting of two components, called $core(C_{\mathrm{OI}})$ and $constraints(C_{\mathrm{OI}})$, where $core(C_{\mathrm{OI}}) = C$ and $constraints(C_{\mathrm{OI}})$ is the set of the inequalities generated by the (OI) axiom, that is,

$$constraints(C_{\mathrm{OI}}) = \{\ t \neq s \mid t,\ s \in terms(C),\ t,\ s \text{ distinct}\}$$

Formally, when the object identity assumption is made, any clause

$$C = head(C) \text{ :- } body(C)$$

takes the form

$$C_{\mathrm{OI}} = head(C) \text{ :- } body(C) \parallel I$$

where I is the set of inequations generated by the (OI) axiom.

Now we can introduce the ordering relation defined by the notion of θ-subsumption under object identity, namely, θ_{OI}-subsumption.

Definition 3 (θ_{OI}*-subsumption)*

A clause C θ-subsumes a clause D under object identity (C θ_{OI}-subsumes D) if and only if (iff) there is a substitution σ such that (s.t.) $C_{\mathrm{OI}}\sigma \subseteq D_{\mathrm{OI}}$.

Thus a clause C is more general than or equal to a clause D under θ_{OI}-subsumption (or D is more specific than or equal to C), and we write $D \leq_{\mathrm{OI}} C$ iff C θ_{OI}-subsumes D, that is,

$$D \leq_{\mathrm{OI}} C \text{ iff } \exists\ \sigma\text{: } C_{\mathrm{OI}}\sigma \subseteq D_{\mathrm{OI}}$$

In such a case, we say that C is a generalization of D and D is a specialization of C under θ_{OI}-subsumption. We write $D <_{\mathrm{OI}} C$ when $D \leq_{\mathrm{OI}} C$ and $not(C \leq_{\mathrm{OI}} D)$, and we say that C is more general than D (C is a proper generalization of D), or D is more specific than C (D is a proper specialization of C) under θ_{OI}-subsumption, or C properly θ_{OI}-subsumes D. We write $C \sim_{\mathrm{OI}} D$ when $C \leq_{\mathrm{OI}} D$ and $D \leq_{\mathrm{OI}} C$, and we say that C is equivalent to D under θ_{OI}-subsumption.

Like θ-subsumption, θ_{OI}-subsumption induces a quasi-ordering upon the space of the clauses. A thorough analysis of θ_{OI}-subsumption as a generalization model can be found in the work by Semeraro et al. (1996).

When performing the search in the space of the VL_{21} linked range-restricted definite clauses ordered by θ_{OI}-subsumption, INDUBI/CSL implements a separate-and-conquer search strategy, while at the low level, it adopts a beam search strategy. A thorough description of the algorithms can be found in the work by Malerba et al. (1997). Figure 1 shows the Pascal-like procedure that implements this search strategy.

Algorithm 1

```
procedure separate_and_conquer(Examples, Class)
    $E^+$ := set of positive examples of Class
    $E^-$ := set of negative examples of Class
    LearnedRules := $\emptyset$
    while $E^+ \neq \emptyset$ do
        randomly select $e^+$ from $E^+$
        MQ := Beam_Search_for_consistent_hypotheses($e^+$, $E^+$, $E^-$, m)
        Best := FindBest(MQ)
        LearnedRules := LearnedRules $\cup$ {Best}
        $E^+$ := $E^+$ - Covers(Best, $E^+$)
    endwhile
    return the set LearnedRules of $VL_{21}$ generalization rules
endproc
```

Figure 1. High-level strategy for learning a set of VL_{21} generalization rules.

More precisely, INDUBI/CSL starts with a positive example E^+ (designated as the *seed* of the search) and generates a set MQ of at least m distinct maximally general generalizations under θ_{OI}-subsumption, expressed as VL_{21}-linked range-restricted definite clauses, which are consistent with respect to all the negative examples in the training set and θ_{OI}-subsumes E^+.

The best generalization is selected from MQ according to a lexicographic evaluation functional (LEF) (Michalski, 1980), which takes into account several criteria.

Then, positive examples covered by the best generalizations are removed from the set of positive examples, and a new clause, with the same head as the previous one, is generated if the set of remaining positive examples is not empty.

At the low level, INDUBI/CSL proceeds top-down, specializing the unit clause $[f(t_1, t_2, ..., t_n) = \lambda] \leftarrow$ by adding one of the selectors in the positive example E^+, after turning the constants that occur as arguments of the descriptor into variables. It has been recently proved that this downward refinement operator satisfies the properties of local finiteness, properness, and completeness (*ideality*) when applied in the space of clauses ordered by θ_{OI}-subsumption rather than by θ-subsumption (Esposito et al., 1996*a*).

Only a subset of n selectors among all selectors in the example E^+ is considered. Such selectors are chosen according to the cost assigned to each descriptor in the selector and according to the arity of the descriptors (descriptors with greater arity are preferred to those with lower arity). Selectors that violate the constraint of linkedness are not taken into account at all. All the clauses resulting from this step θ_{OI}-subsume E^+ (and as many other positive examples as possible).

The best p clauses are selected among them, according to another user-defined LEF and stored in the set PS. The consistent ones are removed from PS and stored in MQ after a process of extension-against (Michalski, 1980).

Figure 2 expresses in a Pascal-like language the procedure that implements the low-level beam search strategy.

Algorithm 2

```
procedure Beam_Search_for_consistent_hypotheses(e⁺, E⁺, E⁻, m)
    PS := {[f(t1,t2, ..., tp)=λi] ← }
    OldPS := Ø
    MQ := Ø
    while PS ≠ OldPS and |MQ| ≤ m do
        OldPS := PS
        PS := Ø
        foreach VL21 generalization G in OldPS do
            S := choose_best_linked_selectors(e⁺, G, n)
            PS := PS ∪ specialize_G_by_adding_a_selector_in_S(G,S)
        endforeach
        CONS := consistent(PS)
        PS := PS - CONS
        PS := select_best_p_generalizations(PS)
        foreach VL21 generalization G in CONS do
            MQ := MQ ∪ extension_against(G, E⁺, E⁻)
        endforeach
    endwhile
    return MQ
endproc
```

Figure 2. Low-level beam search strategy for learning a single VL_{21} clause.

Another characteristic of INDUBI/CSL is that it is a multiple predicate learner, that is, it can learn several concepts, even though they depend on each other (Malerba et al., 1997). Specifically, differently from most learning systems, INDUBI/CSL does not make the assumption that concepts to be learned are independent of one another. Nonetheless, in order to deal with concept dependencies, it requires the user to be able to specify a *dependency hierarchy,* which takes the form of a directed acyclic graph (*dag*), whose nodes represent concepts to be learned. The order in which concepts are learned by INDUBI/CSL is completely defined by the dependency hierarchy. In particular, the concepts at the lowest level of a dependency hierarchy have to be learned first, since their definition does not depend on other concepts (*minimally dependent* concepts).

ATRE

The second learning system used in our experiments is ATRE, which has been implemented in PROLOG.

Similar to INDUBI/CSL, ATRE induces complete, consistent hypotheses from a set of training examples. More precisely, the learning problem solved by ATRE can be formulated as follows:

Given

- a set of concepts $C_1, C_2, ..., C_r$ to be learned
- a set of observations O described in a language of observations L_O
- a background knowledge BK described in a language L_B
- a language of hypotheses L_{Hi} for each concept C_i

- a preference criterion PC
- a generalization model Γ over the space of hypotheses

Find

a theory T, which includes possibly dependent hypotheses H_1, H_2, ..., H_r for the concepts C_1, C_2, ..., C_r, respectively. The hypotheses, described in their corresponding language L_{Hi}, are complete and consistent with respect to the set of observations and satisfy the preference criterion PC.

The main differences between INDUBI/CSL and ATRE are the languages L_O, L_B, L_{Hi}, the interpretation of the preference criterion, the generalization model adopted over the space of the hypotheses, and the way in which the set of hypotheses is searched for. Here, only some of them will be discussed because of space constraints.

First, literals used by ATRE have three distinct forms:

1. $f(t_1, ..., t_n) = Value$ (simple literal)
2. $f(t_1, ..., t_n) \in Range$ (set literal)
3. $f(t_1, ..., t_n)$ # $g(s_1, ..., s_m)$ (relational literal)

where f and g are function symbols called descriptors, t_i and s_i are terms, *Range* is a set of possible values taken by f, and # is a relational operator ($<$, $\leq$, $\geq$, $>$). Some examples of literals are the following:

$color(x_1)=red$, $height(x_1) \in [1.1 .. 1.2]$, and $ontop(x_1, x_2)=true$.

The last example points out the lack of predicate symbols in the representation language adopted by ATRE. Thus the first-order literals $P(x_1, x_2)$ and $\neg P(x_1, x_2)$ will be represented as $f_P(x_1, x_2)=true$ and $f_P(x_1, x_2)=false$, respectively, where f_P is the function symbol associated to the predicate P. Therefore ATRE can deal with *classical negation* ($\neg$) but not with *negation by failure, not* (Lloyd, 1987). This is the main difference with respect to other well-known systems of inductive logic programming (ILP) that express negative information exclusively by means of negation as failure (Bergadano et al., 1996). In this way, ILP systems are in line with the traditional semantics of logic programming that applies the *closed-world assumption* to all predicates. Each ground atom that cannot be derived from a logic program is assumed to be false. Thus a ground query has only two possible answers, *yes* or *no,* because it is not possible to deal directly with incomplete information. On the contrary, the answer to a ground query *Q,* given a logic program with classical negation, can be *yes, no, unknown,* or *inconsistent* (De Raedt, 1992).

Literals can be combined to form *definite clauses,* which can be written as L_0 :- L_1, L_2, ..., L_m, where the *simple* literal L_0 is called *head* of the clause, while the conjunction of simple or set literals L_1, L_2, ..., L_m is called the *body.* It is important to point out that a definite clause should be interpreted as an inference rule, rather than a conditional. Indeed the following classical implications,

$$P(x) \Leftarrow R(x, y), \neg Q(y) \qquad Q(y) \Leftarrow R(x, y), \neg P(x)$$

are identical, while their corresponding representations in ATRE have different meanings. Henceforth, for the sake of simplicity, we will adopt the usual notation $P(x, y)$ and $\neg P(x, y)$ instead of $f_P(x, y)=$*true* and $f_P(x, y)=$*false,* respectively.

In ATRE the concept of definite clause has been generalized to *multiple-head clause.* A multiple-head clause is a clause with a conjunction of simple literals in the head. In the special case of only one literal in the head, a multiple-head clause is reduced to a definite clause. An example of a multiple-head clause that may be used to describe morphological elements in a map is reported below:

class(a)=system_of_cliffs $\wedge$ *class(b)= fluvial_landscape :-*
next(a,b), contain(a,c), contain(b,d), brown(c), blue(d)

which is semantically equivalent to the definite program

class(a)=system_of_cliffs :- next(a,b), contain(a,c), contain(b,d),
brown(c), blue(d)

class(b)=fluvial_landscape :- next(a,b), contain(a,c), contain(b,d),
brown(c), blue(d)

but is not equivalent to the *disjunctive* clause

class(a)=system_of_cliffs, class(b)= fluvial_landscape :-
next(a,b),contain(a,c), contain(b,d), brown(c), blue(d)

whose comma in the head is interpreted as a disjunction and not a conjunction.

In ATRE, training examples are represented by means of *objects,* that is, ground multiple-head clauses with only simple literals. The two main advantages offered by this object-centered representation are higher comprehensibility and efficiency. The former is basically due to the fact that multiple-head clauses provide us with a compact description of multiple properties to be predicted in a structured object. The second advantage derives from the possibility of representing the known properties of an object only once.

The language of hypotheses adopted by ATRE is that of linked range-restricted definite clauses with only simple and set literals in the body. Contrary to INDUBI/CSL, ATRE can learn *simple recursive theories* (Malerba, 1996). In such theories, it is possible to express both dependencies among concepts and recursive clauses (Idestam-Almquist, 1993), but there is no way to express mutual recursion.

The language of the background knowledge has the same constraints as the language of hypotheses, the only difference being that relational literals are allowed in the body of definite clauses.

Regardless of the representation language adopted, a key part of the induction process is the search through a space of hypotheses. A generalization model provides a basis for organizing this search space, since it establishes when a hypothesis covers a positive/negative example and when an inductive hypothesis is more general/specific than another. As already pointed out, the generalization model Γ used by INDUBI/CSL is θ_{OI}-subsumption, which is strictly weaker than θ-subsumption. Unfortunately, neither is appropriate for organizing the space of recursive theories.

The generalization model adopted in ATRE is a variant of *relative implication,* which is based on the following definition of resolution closure of a theory *T,* $R^*(T)$:

$$R^*(T) = R^0(T) \cup R^1(T) \cup R^2(T) \cup \ldots$$

where

$$R^0(T) := T$$

$$R^n(T) := R^{n-1}(T) \cup \{C \mid C_1, C_2 \in R^{n-1}(T), C \text{ is the resolvent of } C_1 \text{ and } C_2\}$$

Definition 4 (relative implication)

Let *C* and *D* be the following two nontautological clauses:

$$C\colon C_0 \leftarrow C_1, C_2, \ldots, C_n$$

$$D\colon D_0 \leftarrow D_1, D_2, \ldots, D_m$$

where the variables in *C* and *D* are distinct. Then clause *C implies* (or is more general than) clause *D* with respect to *T,* $C \leq_{T,\Rightarrow} D$, if a substitution σ exists such that $C_0\sigma = D_0$ and C' exists in $R^*(T \cup \{C\})$ such that $C' \leq_\theta D$.

According to this definition, the following clause

$$C\colon odd(s(x)) \leftarrow even(x)$$

implies

$$D\colon odd(s(s(s(0)))) \leftarrow zero(0)$$

with respect to

$$C_1{:}even(s(x)) \leftarrow odd(x)$$

$$C_2{:}even(x) \leftarrow zero(x)$$

Thus the theory *T* reported in the previous example explains the logical consequence *D*.

The main procedure of ATRE is shown in Figure 3. The input to the procedure is actually the input to the system, that is, the set of objects, the background knowledge (BK), the set of concepts to be learned (grouped into levels of the dependency hierarchy provided by the user), and the set of parameters that guide the heuristic search in the space of possible hypotheses.

The function implemented by the procedure *update_object_description* is the saturation of a set of examples given a set of clauses. The first step toward the generation of inductive hypotheses is the saturation of all examples with respect to the given BK (Rouveirol, 1994). In this way, information that was implicit in the example, given the background knowledge, is now made explicit.

Initially, the set of learned concepts is empty. The depth of the dependency graph is *d;* thus there are *d* disjoint sets of concepts to be learned. All concepts at the same level can be learned independently, by invoking the procedure *separate_and_conquer.* When all concepts at the same level have been learned, it is necessary to perform the appropriate shift of language before moving to the next level. The shift of language is actually obtained by saturating all training objects with the rules just learned.

ATRE implements a separate-and-conquer search strategy at the high level in order to generate the hypothesis H_i for a concept C_i (see Figure 4).

The generation of recursive theories is possible only after the first nonrecursive clause of a concept definition has been found. This approach is based on the principle that a correct recursive definition of a function should always start from the

Algorithm 3

```
procedure learn_multiple_concepts(Objects, BK, {{$C_1^0$, ..., $C_{r1}^0$}, ..., {$C_1^d$, ..., $C_{rd}^d$}}, Parameters)
  update_object_description(Objects, BK)
  AllLearnedRules := Ø
  for i := 0 to d do
      LearnedRulesSameLevel := Ø
      foreach $C_j^i$ in { $C_1^i$, $C_2^i$, ..., $C_{ri}^i$ } do
          LearnedRule := separate_and_conquer(Objects, $C_j^i$, Parameters)
          LearnedRulesSameLevel := LearnedRulesSameLevel ∪ LearnedRule
      endforeach
      if i≠d then update_object_description(Objects, LearnedRulesSameLevel) endif
      AllLearnedRules := AllLearnedRules ∪ LearnedRulesSameLevel
  endfor
  return AllLearnedRules
endproc
```

Figure 3. Main procedure for learning multiple dependent concepts.

Algorithm 4

```
procedure separate_and_conquer(Objects, Concept, [BestLEF | OtherParameters])
  LocalObjects := Objects
  $E^+$ := set of positive examples of Concept generated from Objects
  $E^-$ := set of negative examples of Concept generated from Objects
  $E_0^+ := E^+$
  LearnedRule := $\emptyset$
  while $E^+ \neq \emptyset$ do
      select a seed object Obj from LocalObjects such that it generates an example in $E^+$
      $O^+$ := set of positive examples of Concept generated from Obj
      while $O^+ \neq \emptyset$ do
          Cons := beam_search_for_consistent_hypotheses(Obj, $E^+$, $E^-$, $E_0^+$, OtherParameters)
          Best := find_best(Cons, BestLEF)
          update_object_description(LocalObjects, { Best })
          LearnedRule := stratify(LearnedRule, Best, $E_0^+$, $E^-$)
          $E^+ := E^+$ - covers(Best, $E^+$)
          $O^+ := O^+$ - covers(Best, $O^+$)
      endwhile
  endwhile
  return LearnedRule
endproc
```

Figure 4. High-level separate-and-conquer search strategy for learning a single concept.

axiomatic level. Indeed, as soon as a nonrecursive clause for a concept $f(x_1, x_2, ..., x_n)=\lambda$ has been learned, the procedure *update_example_description* is called in order to include the positive literals $f(x_1, x_2, ..., x_n)=\lambda$ in the body of all objects. This shift of language of hypotheses allows ATRE to extend the search to recursive clauses as well. The procedure *stratify* in Algorithm 4 checks that the best consistent clause selected according to one of the LEF of the preference criterion (BestLEF) does not cause inconsistency of clauses already learned. When this happens, a further change of language is performed by introducing completely new predicates. This operation, called *stratification,* has the effect of recovering the consistency of previously generated recursive clauses, without throwing away the best clause selected in the last iteration.

Finally, ATRE is able to deal with both numeric and symbolic descriptions. More precisely, given an n-ary function symbol, $f(x_1, x_2, ..., x_n)$, taking values in a numerical domain, ATRE is able to produce hypotheses with range literals $f(x_1, x_2, ..., x_n) \in [a .. b]$, where $[a .. b]$ is a numerical interval computed according to an information theoretic criterion.

APPLICATION

In order to automatically process the maps, a grid is superimposed that divides the represented region into regular elements of observation. The corresponding real dimensions of the grid elements depend on the scale factor and are related to the definition of a suitable level of details in the formal representation language in which the relevant morphological elements will be described (grain size). The system simulates the behavior of a human expert when forced to observe through a narrow window a well-defined zone on the map, coinciding with the grid element. Then the window is enlarged, allowing him/her to observe a wider area and to search for a context. Studying the behavior of 15 geomorphologists and experts in territory planning, it was observed that the human expert uses a widening exploratory model in the map interpretation process: he/she moves the narrow observation window over the map so as to individuate and better categorize the morphological elements to be traced. Indeed, the entire interpretation process is carried out within frames that are related to the context, both in a spatial sense (the region's lithological structure, the nature of surrounding territory, the density of human settlements, the presence in the neighborhood of railways or highways, etc.) and in a temporal sense (the historical process of territorial transformation).

In the following, an application is described concerning the automatic acquisition of rules for the recognition of some concepts relevant for studying the environmental change and the planning strategies in a wide region characterized by the presence of a river. After a training phase, the grid elements of the maps are automatically labeled with their corresponding concepts, and this facilitates a quick indexing of the areas in which it is possible to apply the planning rules suggested by the expert system. Moreover, both the labeling process and the planning intervention could be controlled by the final user. The former provides justifications of classifications expressed in a comprehensible language of descriptors recognizable in the map, while the latter is expressed in a sequence of production rules defining the set of actions to be performed.

Preprocessing Phase

During the preprocessing phase, a symbolic description, extracted directly from the digital map, is automatically generated. First, the input map, which is in vectorized form, is segmented into regular cells, each of which constitutes an example used for the learning phase or an elementary unit to be classified. Sometimes it is necessary to convert cells that are not spatially contiguous on the same map and to use more maps. The preprocessing module allows the conversion of an entire map or the conversion of selected cells specified by means of their localization. Map interpretation requires a higher level of information than that contained in a vectorized map. The attributes of the cartographic objects and the geometric relationships among them, which have been used in this study, are listed in Table 1.

Particular attention has been paid to the definition of general descriptors, so that they are appropriate for describing maps even when there is a change in the representation scale of the concepts to be learned.

The transformation of the map from numeric to symbolic form is achieved by means of algorithms that have been specifically developed for this task (Esposito et al., 1995).

Table 1. Map descriptors

Descriptor	Meaning	Type domain	Values	Cost
color(y)	*Denotes the color of the object y*	Nominal	{blue, brown, black}	6
contain(cell,y)	*The observed cell contains the object y*	Boolean	{false, true}	8
density(*y*)	*Denotes the part of cell covered by the object y, where y is vegetation or buildings*	Linear	{low, medium, high}	8
distance(*y*,*z*)	*Represents the distance between the object y and the object z*	Linear	{1..150}	10
geographic_direction (cell,*y*)	*Denotes the geographic direction of the object y in the cell*	Nominal	{north, north_west, north_east, south_east, south_west, south, east, west}	8
inside(*y*,*z*,*k*)	*Denotes the presence of the object k between the objects y and z*	Boolean	{false, true}	10
orientation(cell)	*Defines if the cell orientation is homogeneous or heterogeneous*	Nominal	{homogeneous, heterogeneous}	10
outside(*y*,*z*,*k*)	*Denotes the presence of the object k outside the objects y and z*	Boolean	{false, true}	10
relation(*y*,*z*)	*Denotes the relation between the objects y and z*	Nominal	{almost perpendicular, almost parallel}	10
shape(*y*)	*Defines the form of the contour slope y*	Nominal	{cuspidal, normal}	8
trend(*y*)	*Represents the trend of the object y in the cell*	Nominal	{curvilinear, straight}	8
type_of(*y*)	*Defines the type of the object y*	Nominal	{river_line, farm_road, interfarm_road, main_road, primary_road, contour_slope, railway, vineyard, arboreal_vegetation, slope, affluent, partition_wall, building, fluvial_isle, deep_embankment, regular_embankment}	6

For example, the generation of the descriptor *shape* is based on the following method. First, the attribute *shape* is meaningful only for contour slopes, that is, when the cartographic object belongs to the orographic features of the map and it has a linear geometry; this information is derived by analyzing the header of the map file and the geometric label of the object. Two different values are considered for the descriptor *shape,* namely, *cuspidal* and *normal.*

Let a contour slope Y be defined through n points. The angles w_i are determined as

$$w_i = \operatorname{arctg} \frac{x_{i+1} - x_i}{y_{i+1} - y_i} \qquad i = 1, 2, \ldots, n-1$$

The differences dw_i are calculated as

$$dw_i = w_{i+1} - w_i \qquad i = 1, 2, \ldots, n-1$$

A cuspidal value is associated to the shape of Y if there exists a difference dw_j whose value exceeds a threshold τ, which depends on the examined territory. For contour slopes similar to that represented in Figure 5, this algorithm provides the value *cuspidal,* while it associates the value *normal* to contour slopes like that shown in Figure 6.

A cost is associated to each descriptor. The cost of a descriptor represents the amount of human and machine resources required for its generation; that is, when the algorithm for the extraction of a certain descriptor is more complex or requires more processing time than for generating another descriptor, a greater cost is attributed to it.

In fact, a low cost has been associated to the descriptor *color,* since it entirely depends on the type of the object.

The benefit deriving from the specification of a cost for each descriptor is relevant during the learning phase, when the preference criteria are adopted. In fact, when varying the set of costs associated to the descriptors, different rules can be obtained. An example of a symbolic description produced by the preprocessing module is shown in Figure 7.

Experimentation

The territory used for this study is in the Apulian region around the Ofanto River, from the zone of Canosa to its mouth. The same territory is represented by one map

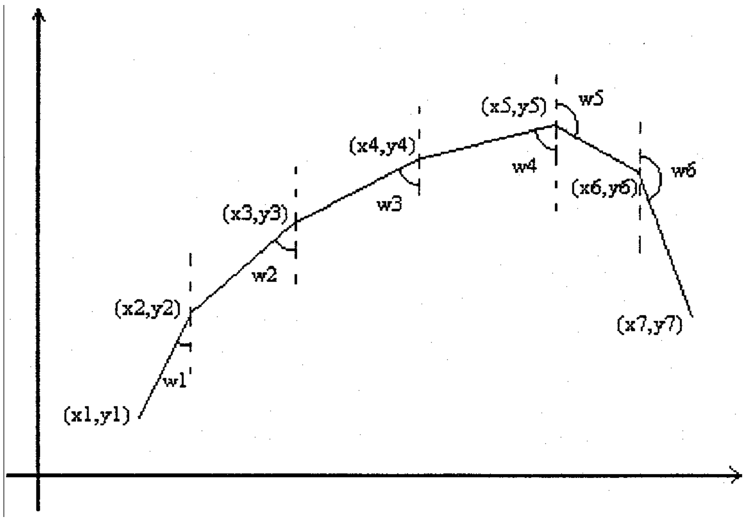


Figure 5. Cuspidal-shaped contour slope.

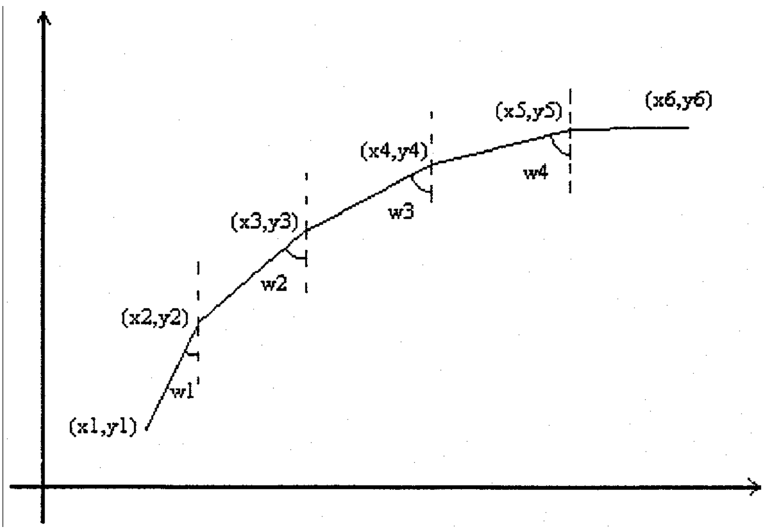
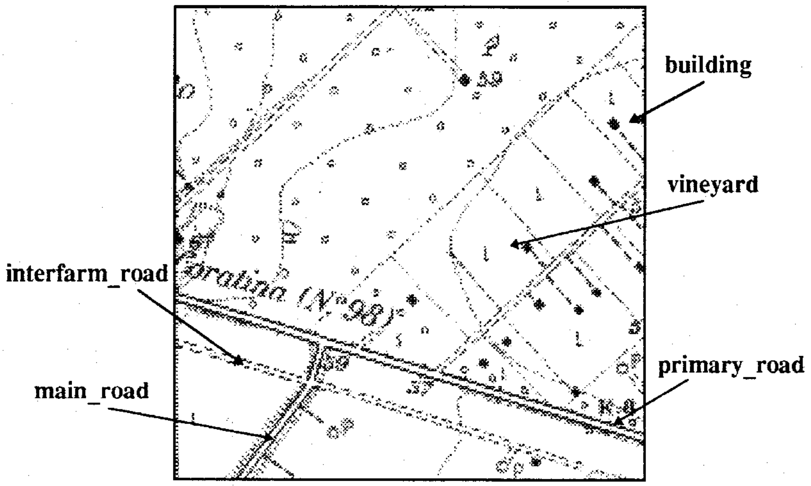


Figure 6. Normal-shaped contour slope.



[class(x1)=ROYAL_CATTLE_TRACK] :-
contain(x1,x2), contain(x1,x3), contain(x1,x4), contain(x1,x5), contain(x1,x6), type_of(x2)=primary_road, type_of(x3)=interfarm_road,type_of(x4)=main_road, type_of(x5)=vineyard, type_of(x6)=building, color(x2)=black, color(x3)=black, color(x4)=black, color(x6)=black, relation(x2,x3)=almost_parallel, relation(x2,x4)=almost_perpendicular, relation(x3,x4)=almost_perpendicular, distance(x2,x3)=115, density(x5)=low, density(x6)=low, outside(x2,x3,x5), geographic_direction(x1,x2)=north_west, geographic_direction(x1,x3)=north_west

Figure 7. Example of a symbolic description.

at a scale of 1:50000 and five maps at 1:25000, which have been produced by the Italian Military Geographic Institute (IGM).

The experimentation concerns four environmental concepts that are relevant for environmental protection: *regular grid system of farms, fluvial landscape, system of cliffs,* and *royal cattle track.* The regular grid system of farms consists of partitions of farms arranged in a rectilinear manner. The fluvial landscape is individuated by the presence of waterways, fluvial islands, and embankments. The system of cliffs is related to the emergence of limestone as a single block. Finally, the royal cattle track, exclusively present in southern Italy, is a road for transhumance. It is particularly interesting because it is characterized by the presence of an uncultivated, quite regular trace, about 80–100 m wide, with a defined orientation, testifying to ancient paths of flocks and herds, which have now partially disappeared and been incorporated into the road network.

In Figure 8 the following data are shown:

- the name of the topographic chart and the information for its localization,
- the examined area for each map, expressed in square kilometers, and
- some details concerning the use for the training or the testing process.

Experiment A concerns maps at 1:25000 scale, while Experiment B concerns smaller-scale maps. The examined area covers 246 km^2. In order to have a fair comparison of results, most of the territory considered in Experiment B is also considered in Experiment A.

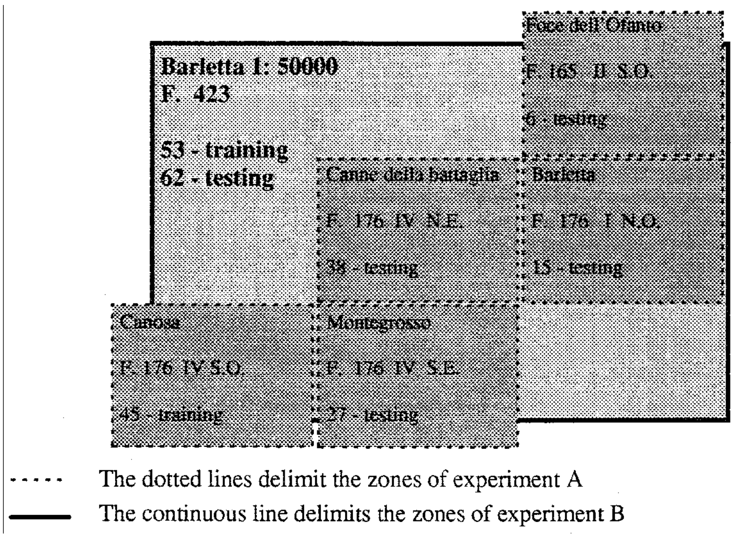


Figure 8. Examined territory.

The experiments have been conducted using both INDUBI/CSL and ATRE. Experimental setting data are described in Table 2.

Experiment A

Each training/test example is a 1 km^2 cell on the map. The selection of the shape and size of the examples coincides with the gridding system superimposed over IGM geographic charts at 1:25000 scale. All the training examples (45) are taken from the map of Canosa. In addition to the four classes, another class—*other*—containing only counterexamples, has been used. For the testing phase, 86 instances from adjacent zones (Canne della Battaglia, Montegrosso, Barletta, and Foce dell'Ofanto) have been chosen. They represent nearly 65% of the total experimental data set.

The preference criteria used for the selection of hypotheses are (1) preference for the most complex generalized descriptions, i.e., those expressed by the maximum number of literals, and (2) minimizing the cost of a generalization. The first criterion is intended to attain the agreement of the experts. Generally, when human planners are required to judge multiple equivalent solutions produced by the systems, they prefer the more articulated generalizations. The second criterion aims at satisfying an efficiency principle, that is, the most favored solution has the minimum cost, which corresponds to the sum of the costs associated to each descriptor appearing in the generalization.

Here we report some generalizations produced by INDUBI/CSL.

```
[class(x1)=ROYAL_CATTLE_TRACK]:-
        [inside(x2,x3,x4)=true], [outside(x2,x3,x5)=true], [density(x5)=high],
        [density(x4)=low], [geographic_direction(x1,x2)=north_west],
        [geographic_direction(x1,x3)=north_west], [distance(x2,x3)=95],
        [relation(x2,x3)=almost_parallel], [contain(x1,x2)=true],
        [contain(x1,x3)=true], [contain(x1,x4)=true]
```

which can be automatically translated into the following natural language sentence: "if there are two almost parallel objects 95 m apart, directed toward northwest, and

Table 2. Sizes of the example sets in the experimentations

	Experiment A		Experiment B	
Class	Training	Testing	Training	Testing
System of cliffs	12	20	10	13
Royal cattle track	5	4	1	0
Fluvial landscape	13	18	13	23
Regular grid system of farms	11	27	21	21
Other	4	17	8	5
Total no. of instances	45	86	53	62

between them there is low-density vegetation, while outside them there is high-density vegetation, then the cell contains a royal cattle track."

```
[class(x1) = FLUVIAL_LANDSCAPE]:-
        [geographic_direction(x1,x3)=north_east], [trend(x3)=curvilinear],
        [geographic_direction(x1,x2)=north_east], [trend(x2)=curvilinear],
        [distance(x2,x3)=35],[relation(x2,x3)=almost_parallel],[color(x2)=blue],
        [color(x3)=blue], [type_ of(x3)=river_line], [type_of(x2)=river_line],
        [contain(x1,x3)=true], [contain(x1,x2)=true]
```

which states that, "if there are two blue curvilinear stretches that are almost parallel, 35 m apart, and both directed toward northeast, then the cell contains a fluvial landscape."

The rules generated by ATRE for the same concepts are

```
class(x1)=ROYAL_CATTLE_TRACK:-
      contain(x1,x2), distance(x3,x2) in [90.0 .. 130.0],
      type_of(x2)=interfarm_road.

class(x1)=FLUVIAL_LANDSCAPE:-
      contain(x1,x2), type_of(x2)=river_line, color(x2)=blue.
```

The meaning of these rules can be easily grasped. In the testing phase, the reserved cells have been correctly recognized by the induced rules. More precisely, the predictive accuracy was over 95% for both systems, neither of which made any commission errors. Only INDUBI/CSL made some omission errors concerning the class fluvial landscape. The errors were caused by the inability of INDUBI/CSL to apply the extension-against rule to relational numeric descriptors, such as the descriptor *distance*. No rejection occurred when using the generalizations induced by ATRE.

Experiment B

The representation scale of the map used in this experiment is 1:50000. The classes are the same as for Experiment A, and as can be seen from Table 2, about 46% of the instances were used for the training phase, while the remaining 54% were used for testing. The same preference criteria specified for Experiment A were adopted for both systems. The generalizations produced by the two systems differed, but only slightly, from those obtained in Experiment A, because the training set was different.

At this scale factor the most important concept is the *fluvial_landscape,* which constitutes the major morphological pattern appearing in the entire observed region. In particular, INDUBI/CSL generates a new rule for this class:

```
[class(x1)=FLUVIAL_LANDSCAPE]:-
          [outside(x3,x4,x5)=true], [geographic_direction(x1,x4)=north],
          [geographic_direction(x1,x3)=north], [distance(x3,x4)=50],
          [relation(x2,x3)=almost_parallel], [color(x3)=blue], [color(x4)=blue],
          [contain(x1,x5)=true, [contain(x1,x2)=true], [contain(x1,x3)=true]
          [contain(x1,x4)=true]
```

For the concept *fluvial_landscape,* ATRE generated a rule identical to that already reported in Experiment A. The recognition of the 62 km^2 of the territory used for the testing phase was totally correct. In fact, also for this experiment there was no misclassification in either system.

DISCUSSION OF RESULTS

The positive results reported above for both experiments prove the strength of the adopted approach for interpretation tasks. Some factors that may have influenced the results are the high dimensionality of the learning database and the homogeneity of the testing territories with respect to the zones selected for the training set.

The cells containing the river, which were rejected in Experiment A on the basis of the rules produced by INDUBI/CSL, are localized near the mouth, and so the rejections are explained by the increased distance between the two banks of the Ofanto. On the contrary, the possibility given by ATRE of dealing properly with numerical descriptions led to the generation of more accurate rules.

However, the predictive accuracy is not the only criterion to be adopted in the evaluation of the system performance. Since the main task of the interface is the classification of morphological elements with the aim of a semantic indexing, the classification rules must be self-explanatory and measured in terms of comprehensibility, as well.

It is well known that, at the moment, rigorous methods allowing an objective evaluation do not exist (Bratko et al., 1996). We set up a simple empirical procedure for measuring the *comprehensibility* of the learned rules. It consisted in interviewing the 15 experts who were involved in the study, asking them first to grade the intelligibility of the rules obtained by the two systems, specifying one of the five permitted values (very poor, poor, sufficient, good, very good), and then to briefly comment on the rules. In order to avoid influencing their judgment, when this investigation was carried out, the experts were not yet aware of the results of the testing phase.

Generally, they judged the rules generated by both systems to be fairly good. Nevertheless, they manifested their preference for those produced by INDUBI/CSL because they found them more meaningful and similar to their own classification rules.

This confirms that, although the rules produced by ATRE gave the best results in terms of predictive accuracy, experts preferred less synthetic rules. Therefore the first preference criterion adopted in this study is suitable for attaining the agreement of the experts.

In the course of our study, the possibility of reading the evolution of the landscapes emerged as a very interesting result; in fact, the relevance of some signs is not static but varies with time. For example, by analyzing the training zones for the two experiments, a discrepancy emerged as to the number of the instances of the *royal_cattle_track* class present on the same area. More precisely, in the zone of Canosa, five examples of this class occur when considering the map at a scale 1:25000 (Experiment A), while only one example of such a class is found when the map at scale 1:50000 is used in Experiment B. At first glance, such a difference would be ascribed to the diverging classification made by the experts involved in the two experiments. However, the main reason is due to the difference of the periods of time in which the two maps were drawn up and to the different viewpoints of the map designer. Indeed, the map at scale 1:50000 was drawn out in 1976 by deriving it, with a suitable updating, from the same maps at scale 1:25000 that were traced out in 1957. Thus the presence of the royal cattle track has been simply "over-shadowed" by a new main road, Highway 98, which was built in the 1970s, following the same course of the royal cattle track.

When automatically acquiring the knowledge base of an expert system, the system's capability of updating the rules in the knowledge base emerged as the main requirement (Tecuci, 1992). Indeed, when the nature of the concepts evolves dynamically or when the area of interest for the planning actions changes, the rules available in the knowledge base of the advisory system turn out to be no longer appropriate for the task.

As a matter of fact, environmental planning is a task affected by a striking evolution of the objects involved. This requires that the knowledge base of an advisory system for this domain must be updated regularly. Sometimes the volumes of knowledge to be updated can be significant, and eventually, this can lead to giving up on using the advisory system itself.

Once again, machine learning techniques provide an efficient and practically useful solution to this problem. They contribute to improve the flexibility of the advisory system, since it can be customized quickly and easily according to the specific areas of territory the end user is interested in.

In this article, the experimentation conducted using two *batch* learning systems, namely, INDUBI/CSL and ATRE, is described. These systems start from an empty logical theory and stop the learning process when the current set of hypotheses is able to explain all the available examples. When new evidence contradicts the learned theory, the whole learning process must be repeated, taking no advantage of the previous version of the hypotheses. Such a drawback can be overcome by means of *incremental* learning systems. These systems are able to revise and refine a theory in an incremental way; thus the previously generated hypotheses are not completely rejected, but they are taken as the starting point of a search process whose goal consists of a new theory that explains both old and new observations.

Currently, we are carrying out a new experiment in which we use the incremental learning systems INCR/H (Semeraro et al., 1995) and INTHELEX (Esposito et al., 1996b), the former written in C language and the latter in PROLOG, which are the systems that revise the theories produced by INDUBI/CSL and ATRE, respectively.

CONCLUSIONS AND FUTURE WORK

Using learning algorithms on images to interface machines and humans offers an exciting new application domain for machine learning. In fact, the possibility of applying symbolic high-level representations allows us to fill the gap between the visual data and more abstract representations, and to view the problem of image interpretation as a representation change problem.

For the future, we are planning to study how to integrate information from different sources: descriptions extracted from thematic charts and normative texts of the same territory, with the aim of a more accurate semantic interpretation. This should allow us to represent different layers of the same reality and improve both the efficiency and the quality of the interpretation process, thus improving the content-based indexing of the spatial elements on a map.

REFERENCES

Bergadano, F., D. Gunetti, M. Nicosia, and G. Ruffo. 1996. Learning logic programs with negation as failure. In *Advances in inductive logic programming,* ed. L. De Raedt, pp. 107–123. Amsterdam: IOS Press.

Bratko, I., B. Cestnic, and I. Kononenko. 1996. Attribute-based learning. *AI Communication.* (March) 27–32.

Chang, C.-L., and R.C.-T. Lee. 1973. *Symbolic logic and mechanical theorem proving.* San Diego: Academic Press.

De Raedt, L. 1992. *Interactive theory revision: An inductive logic programming approach.* London: Academic Press.

Esposito, F., D. Malerba, and G. Semeraro. 1994a. Multistrategy learning for document recognition. *Applied Artificial Intelligence* 8:33–84.

Esposito, F., D. Malerba, G. Semeraro, C. Brunk, and M. Pazzani. 1994b. Traps and pitfalls when learning logical definitions from relations. In *Methodologies for intelligent systems,* eds. Z. W. Ras, and M. Zemankova, pp. 376–385. Berlin: Springer-Verlag.

Esposito, F., A. Lanza, D. Malerba, and G. Semeraro. 1995. Algorithms for the extraction of symbolic descriptors from digital maps. LACAM Technical Report, University of Bari, Italy.

Esposito, F., A. Laterza, D. Malerba, and G. Semeraro. 1996*a.* Locally finite, proper and complete operators for refining Datalog programs. In *Foundations of intelligent systems,* eds. Z. W. Ras and M. Michalewicz, pp. 468–478. Berlin: Springer.

Esposito, F., D. Malerba, G. Semeraro, L. Abbrescia, S. Ferilli, and M. Lupo. 1996*b.* INTHELEX: A full incremental learning system. LACAM Technical Report, University of Bari, Italy.

Esposito, F., and A. Lanza. 1996. The application of machine learning techniques to map interpretation. In *Soft Computing in Remote Sensing Data Analysis,* eds. E. Binaghi, P. A. Brivio, and A. Rampini, pp. 101–112. Singapore: World Scientific.

Grant, J., and V. S. Subrahmanian. 1995. Reasoning in inconsistent knowledge bases. *IEEE Transactions on Knowledge and Data Engineering* (February):177–189.

Helft, N. 1987. Inductive generalization: A logical framework. In *Progress in machine learning—Proceedings of EWSL 87: Second European Working Session on Learning,* eds. I. Bratko and N. Lavrac, pp. 149–157. Wilmslow: Sigma Press.

Idestam-Almquist, P. 1993. Generalization under implication. Ph.D. thesis, Department of Computer and Systems Sciences, Stockholm University, Sweden.

Kodratoff, Y., V. Moustakis, and N. Graner. 1994. Can machine learning solve my problem? *Applied Artificial Intelligence* 8:1–31.

Kuutti, K., and L. J. Bannon. 1993. Searching for unity among diversity: Exploring the "interface" concept. In *Proceedings INTERCHI' 93, New York,* pp. 263–268.

Lloyd, J. W. 1987. *Foundations of logic programming,* 2nd ed. Berlin: Springer-Verlag.

Malerba, D. 1996. Learning recursive theories with ATRE. LACAM Technical Report, University of Bari, Italy.

Malerba, D., G. Semeraro, and F. Esposito. 1997. A multistrategy approach to learning multiple dependent concepts. In *Machine learning and statistics: The interface,* eds. G. Nakhaeizadeh and C. Taylor, pp. 87–106. New York: Wiley.

Michalski, R. S. 1980. Pattern recognition as a rule-guided inductive inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (July):349–361.

Muggleton, S., and C. Feng. 1992. Efficient induction of logic programs. In *Inductive logic programming,* ed. S. Muggleton, pp. 281–298. London: Academic Press.

Plotkin, G. D. 1970. A note on inductive generalization. In *Machine intelligence 5,* eds. B. Meltzer, and D. Michie, pp. 153–163. Edinburgh: University Press.

Plotkin, G. D. 1972. Building-in equational theories. In *Machine intelligence 7,* eds. B. Meltzer, and D. Michie, pp. 73–90. Edinburgh: University Press.

Quek, F. K. H., and M. C. Petro. 1993. Human-machine perceptual cooperation. In *Proceedings INTERCHI' 93, New York,* pp. 123–130.

Reiter, R. 1980. Equality and domain closure in first order databases. *Journal of ACM* 27:235–249.

Rouveirol, C. 1994. Flattening and saturation: Two representation changes for generalization. *Machine Learning* (February):219–232.

Semeraro, G., F. Esposito, N. Fanizzi, and D. Malerba. 1995. Revision of logical theories. In *Topics in artificial intelligence,* eds. M. Gori and G. Soda, pp. 365–376. Berlin: Springer.

Semeraro, G., F. Esposito, and D. Malerba. 1996. Ideal refinement of datalog programs. In *Logic program synthesis and transformation,* ed. M. Proietti, pp. 120–136. Berlin: Springer.

Tecuci, G. 1992. Automating knowledge acquisition as extending, updating, and improving a knowledge base. *IEEE Transactions on Systems, Man and Cybernetics.* (Nov./Dec.):1444–1460.