APPLIED STOCHASTIC MODELS AND DATA ANALYSIS, VOL. 7, 153-166 (1991)

SEARCHING FOR KNOWLEDGE IN A WORLD FLOODED WITH FACTS

R. S. MICHALSKI
Center for Artificial Intelligence, George Mason University, Fairfax, VA 22030, U.S.A.

SUMMARY

The wide availability of computer technology and large electronic storage media has led to an enormous proliferation of databases in almost every area of human endeavour. This naturally creates an intense demand for powerful methods and tools for data analysis. Current methods and tools are primarily oriented toward extracting numerical and statistical data characteristics. While such characteristics are very important and useful, they are often insufficient. A decision maker typically needs an interpretation of these findings, and this has to be done by a data analyst. With the growth in the amount and complexity of the data, making such interpretations is an increasingly difficult problem. As a potential solution, this paper advocates the development of methods for *conceptual data analysis*. Such methods aim at semi-automating the processes of determining high-level data interpretations, and discovering *qualitative* patterns in data. It is argued that these methods could be built on the basis of algorithms developed in the area of machine learning. An exemplary system utilizing such algorithms, INLEN, is discussed. The system integrates machine learning and statistical analysis techniques with database and expert system technologies. Selected capabilities of the system are illustrated by examples from implemented modules.

KEY WORDS Data analysis Conceptual data analysis Machine learning Knowledge discovery in databases Inductive inference

1. INTRODUCTION

The current information age is characterized by an enormous expansion of data generated and stored about all kinds of human activities. An increasing proportion of these data is recorded in the form of computer databases. This makes the data easy to access and easy to analyse by computer technology. The rapid growth of such databases has not however, been matched by an equally intensive development of new types of method and powerful new tools for analysing and interpreting the data. As a result, we face today the growing problem of how to extract desirable knowledge from the large accumulations of data.

The existing data analysis tools are very useful and important for a whole range of data analysis tasks. They are, however, primarily oriented toward the extraction of quantitative and statistical data characteristics, and as such have certain inherent limitations. These tools include cluster analysis, numerical taxonomy, regression analysis, multivariate statistical methods, stochastic models, multidimensional analysis, times series analysis, non-linear estimation techniques, and others. For a sample of research on data analysis see, e.g., papers by Tukey,[1] Diday,[2] Daniel and Wood,[3] and Morgenthaler and Tukey.[4]

These traditional data analysis techniques are particularly useful for such tasks as producing statistical data summaries, fitting equations to data, revealing data organization on the basis

8755-0024/91/020153-14$07.00

Received 3 January 1991

of certain numerical measures, developing mathematical data models, etc. Their results facilitate useful data interpretations, and can help us to gain important insights into the processes that generated the data. These interpretations and insights are the ultimate knowledge sought for by those who build databases. Yet, this knowledge is not created by the current tools, but has to be derived by human data analysts. As the quantity of available data increases, the complexity of extracting knowledge from the data also increases, and may outstrip the capabilities of data analysts.

Summarizing, traditional numerically-oriented techniques offer powerful tools and have important practical applications, but they can solve only a limited range of problems. For example, a statistical data analysis can discover a correlation between certain variables. It cannot, however, produce a conceptual characterization or a casual explanation of why such a correlation exists. Nor can it develop a justification of this correlation in terms of higher-level concepts or analogies to known phenomena. A statistical analysis can determine a central tendency and the variability of various properties; a regression analysis can fit a complex curve to a set of datapoints. These techniques cannot, however, develop a qualitative characterization of the regularities in the datapoints and their dependence on various qualitative factors, nor can they draw an analogy between this characterization and some regularity or law in another domain, or relate it to similar past experiences.

A numerical taxonomy technique can create a classification of entities, and specify a numerical similarity between the entities assembled in the same or different classes. It will not, however, hypothesize reasons for the entities being in the same class, or build qualitative descriptions of the classes created.

Attributes that define the similarity, as well as the similarity measures, must be defined by a data analyst in advance. These techniques cannot generate relevant attributes and appropriate similarity measures by themselves. All the above processes seem to require complex symbolic reasoning that relates high level concepts and analysis goals to available quantitative measures and transformations relevant to these goals.

A question then arises to the possibility for performing some of the above data analysis functions by an automatic or semi-automatic process. The paper tries to address this question, and discusses some novel data analysis tools.

2. CAN MACHINE LEARNING HELP DATA ANALYSIS?

The major premise of this paper is that various artificial intelligence methods, in particular, symbolic methods of machine learning and discovery, offer new and potentially useful tools for data analysis. These tools can perform new types of operations on data, and thus widen the scope of data analysis tasks that can be automated or semi-automated. Specifically, they can perform certain tasks of *conceptual data analysis*, that is, determine high-level symbolic data characterizations, and discover *qualitative* patterns in data. Let us try briefly to review some of these methods in the context of database applications.

2.1. Rule learning from examples

One class of machine learning methods potentially useful for data analysis is based on methods for inductive learning from examples. Given a set of examples of different classes (or concepts), and certain problem-relevant knowledge ('background knowledge'), an inductive learning method hypothesizes a general description of each class. The description is usually expressed as a set of decision rules or as a decision tree.

A decision rule can have different forms; here we assume the following form:

CLASS <:: CONDITION

where CLASS denotes a class or a concept that is assigned to an entity if that entity satisfies the CONDITION. The CONDITION is often a conjunction of some elementary conditions on the values of attributes, or a disjunction of such conjunctions (a DNF form). We assume here that if the CLASS needs a disjunctive description, then several conjunctive rules are linked to the same CLASS. For example, Figure 1 gives an example of a disjunctive description of Class 1 in the form of two rules:

Class 1 <:: Jacket Color is Red, Green or Blue &
Head Shape is Round or Octagonal
Class 1 <:: Head Shape is Square and Jacket Color is Yellow

Figure 1. A two-rule description of Class 1

These rules characterize a class of robot-figures used in the EMERALD system for demonstrating machine learning capabilities. Paraphrasing, 'A robot belongs to Class 1, if the colour of its jacket is red, green or blue, and his head is round or octagonal; or, alternatively, if the colour of its jacket is yellow and its head is square.'

In a decision tree representation, nodes correspond to attributes, branches stemming from the nodes to attribute values, and the leaves to individual classes (see, e.g. Quinlan[5]). A tree can be simply transformed into a set of decision rules (a rule set) by traversing all paths from the root to individual leaves. The opposite process, i.e. transforming a ruleset into a decision tree, is not so direct. The reason is that a rule representation is generally more powerful than a tree representation. The word 'powerful' means here that, e.g., for some simple rulesets, the equivalent decision tree may be more complex (e.g. Michalski[6]).

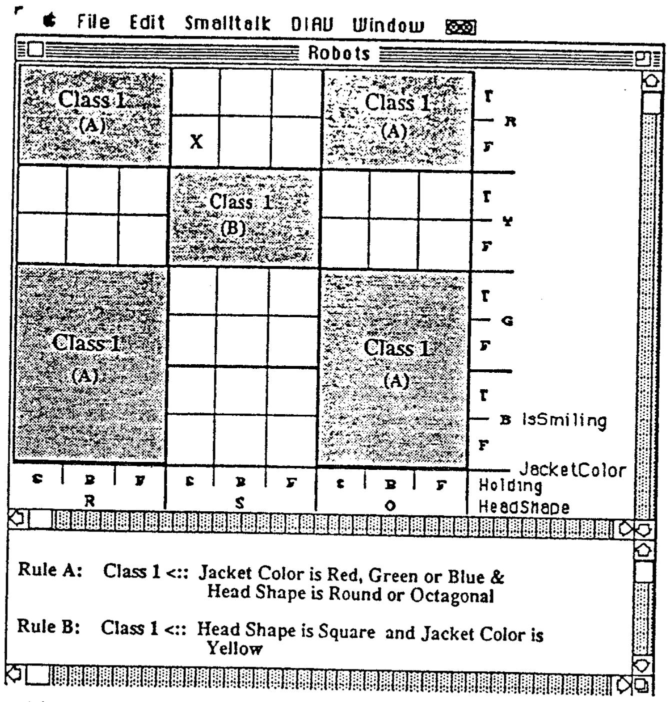
The EMERALD system, mentioned above, combines five programs that display different kinds of learning capabilities (Kaufman *et al.*[7,8]). These capabilities include rule learning from examples, learning distinctions between structures, conceptual clustering, prediction of object sequences, and derivation of equations and rules characterizing data about physical processes. The rules in Figure 1 were generated by the rule learning program (version AQ-15; Michalski *et al.*[9]) from a set of 'positive' and 'negative' examples of robot-figures (Kaufman and Michalski[8]). This paper concentrates on the applicability of two of the above capabilities to data analysis, namely, rule learning and conceptual clustering. For a description of other capabilities see, e.g. Kaufman *et al.*[10]

Most inductive rule learning methods learn *attributional* descriptions of entities in a class, i.e., descriptions that involve only binary or multiple-valued attributes. Some methods learn *structural descriptions*, which characterize entities in terms of both attribute values and relationships that hold among components of the entities. Such relationships are represented by multiplace predicates (Michalski[11]). For data analysis, the most interesting programs seem to be ones for learning attributional descriptions, because typical databases characterize entities in terms of attributes.

The input to attributional learning programs are sets of examples of individual classes, and some 'background knowledge' relevant to the learning problem. The examples are in the form of vectors of attribute-value pairs associated with some decision class. In many cases, background knowledge (BK) is limited to the information about the legal values of the attributes, their type (the scale of measurement), and the *preference criterion* for choosing between candidate hypotheses. Such a criterion is defined by the user in advance. In addition to BK, a learning method may have a *representational bias*, i.e. it may constrain the form of descriptions to only a certain type of expression, e.g. single conjunctions, decision trees, sets of conjunctive rules or DNF expressions. In some methods, BK may include more information, e.g. constraints on the interrelationship between various attributes, rules for generating higher level concepts or attributes, and/or some initial hypothesis (e.g. Michalski[11]). Learned rules are usually *consistent* and *complete* with regard to the input data. This means that they completely and correctly classify all the original 'training' examples. Section 4 presents example solutions from the inductive concept learning program AQ15. In some applications, especially those involving learning rules from noisy data or learning *flexible* concepts (Michalski[6]), it may be advantageous to learn descriptions that may be incomplete or inconsistent (Bergadano *et al.*[12]).

Attributional descriptions can be easily visualized by mapping them into a set of cells in a certain diagram. For example, Figure 2 shows a diagrammatic visualization of the rules from Figure 1.

Such a diagram is a planar representation of a multidimensional space spanned over the set of attributes (Michalski[13]; Wnek *et al.*[14]). The diagram in Figure 2 was generated by the visualization program DIAV (Wnek[14]). Each cell in the diagram represents one combination of values of the attributes. For example, the cell marked by an X represents the vector: (HeadShape = S, Holding = S, Jacket Colour = R, IsSmiling = F). The four shaded areas



Rule A: Class 1 <:: Jacket Color is Red, Green or Blue & Head Shape is Round or Octagonal

Rule B: Class 1 <:: Head Shape is Square and Jacket Color is Yellow

Figure 2. A diagrammatic visualization of rules from Figure 1.

marked Class 1 (A) represent rule A, and the shaded area marked Class 1 (B) represents rule B. In such a diagram, conjunctive rules correspond to certain regular arrangements of cells and can be easily recognized (Michalski[13]).

The diagrammatic visualization can be used for displaying the *target concept* (i.e. the concept to be learned), the training examples (the examples and counter-examples of the concept), and the actual concept learned by a method. By comparing the target concept with the learned concept, one can determine the *error image*, i.e. the area containing all examples that would be incorrectly classified by the learned concept. Such a diagrammatic visualization method can illustrate any kind of attributional learning process (Wnek *et al.*[14]). Because a data table used in the data analysis can be viewed as a set of points in a multidimensional space, the visualization technique can be a useful tool for representing the data and the learned symbolic descriptions (when the space does not exceed the capabilities of a display terminal).

Two types of data analysis operators can be based on methods for learning concept descriptions from examples:

(1) an operator for determining a general symbolic description of any designated group or groups of entities in a data set (such a description expresses the common properties of the entities in the group; it can use abstract concepts in the description that are not present in the original set via the mechanism of constructive induction (see later). This operator is based on a program for learning the so-called *characteristic concept descriptions*);
(2) determining differences between different groups of entities (such differences are expressed in the form of rules that define the properties that characterize one group but not the other; this operator is based on a program for learning the so-called *discriminant concept descriptions*).

Section 3 will illustrate these two types of descriptions. For their definitions, see Reference 11. Basic methods for concept learning assume that examples do not have errors, that all attributes have a specified value in them and that concepts to be learned have a precise ('crisp') description. In many situations, one or more of these assumptions may not hold. Therefore, some more advanced research problems in the area of concept learning from examples include:

(a) Learning from incorrect data (i.e. learning from examples that may contain a certain number of errors or noise);
(b) learning from incomplete data (i.e. learning from examples in which the values of some attributes are unknown);
(c) learning flexible concepts (i.e. concepts that lack precise definition and whose meaning is context-dependent).

All these problems are relevant to data analysis, and thus the methods for solving them being developed in machine learning can be potentially useful. For some of these methods see, e.g., References 6, 12 and 15.

2.2. Conceptual clustering

Another class of machine learning methods that are important to data analysis are those concerned with developing 'conceptual' classifications of a given set of entities. The problem is similar to that considered in traditional cluster analysis, but is defined in a different way. Given a set of attributional descriptions of some entities, a description language for characterizing classes of such entities, and a 'classification quality' criterion, the problem is to split the entities into classes that maximize this criterion, and simultaneously determine general (extensional) descriptions of these classes in the assumed language. Thus, a conceptual clustering program seeks not only a classification (a dendrogram), but also a symbolic description of the proposed classes (clusters). Also, unlike in cluster analysis, the process of determining the classes is dependent on the properties of descriptions of the generated classes.

To clarify the difference between conceptual clustering and conventional clustering, notice that a conventional method typically determines clusters on the basis of a similarity measure that is a function solely of the properties (attribute values) of the entities being compared, and not of any other factors:

$$\text{Similarity}(A, B) = f(\text{properties}(A), \text{properties}(B))$$

In contrast, a conceptual clustering program clusters entities on the basis of a *conceptual cohesiveness* that is not only a function of properties of the entities, but also of two other factors: the description language L, which the system knows *a priori* and uses for describing the classes of entities, and of the set of neighbouring examples, the *environment*, E:

$$\text{Conceptual cohesiveness}(A, B) = f(\text{properties}(A), \text{properties}(B), L, E)$$

Thus, two objects may be similar, i.e. close according to some distance measure, but have a low conceptual cohesiveness. An example of such a situation is shown in Figure 3. According to conceptual clustering, the points (black dots) in Figure 3 would be clustered into two 'ellipses.' The points A and B, which are 'close' to each other, have small conceptual cohesiveness because they belong to configurations representing different concepts.

A classification quality criterion may take into consideration several factors, such as the 'fit' of a *cluster description* to the data, the simplicity of the description, and/or other properties of the entities or of the concepts that describe them (Michalski[16]). Section 4 gives an example of conceptual clustering.

2.3. Other related symbolic operations on data

Methods for learning rules from examples usually assume that attributes used for describing the examples are given *a priori*. These attributes must be sufficiently relevant to the learning problem, otherwise, the learned rules will be poor. An important advantage of symbolic methods over, e.g., statistical methods, is that they can relatively easily handle *non-essential* attributes. An attribute is *non-essential*, if there is a complete and consistent description of the

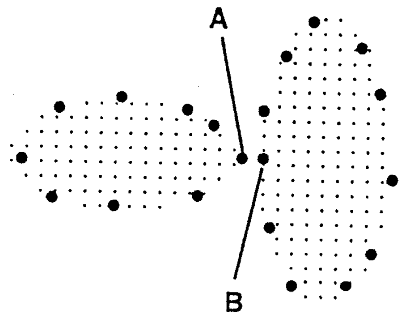


Figure 3. An illustration of the difference between closeness and conceptual cohesiveness

class(s) under consideration that does not use this attribute. Thus, a non-essential attribute is either just irrelevant, or relevant but dispensable. Inductive learning programs such as the rule-learning program AQ, or the decision tree learning ID3, can relatively easily cope with a large number of non-essential attributes in data descriptions.

If there are, however, very many non-essential attributes in the initial description of the examples, the efficiency of a learning program would decrease. Such situations call for a method that would determine the most relevant attributes for a given classification from among those given initially. Only attributes that are most relevant would then be used in the description learning process. Determining the most relevant attributes can therefore be a useful data analysis operator. Such an operator can also be useful on its own merit, as it may be of interest to know which attributes are, e.g., most discriminatory for a given set of classes.

Another related operator is to determine the most representative or, generally, the most important examples in a given set. Such an operator would be needed when there are very many examples of a given class. The most important examples would be those that are either most typical or most extreme. A method for determining the latter, the so-called 'outstanding representatives', has been described by Michalski and Larson.[17]

In many applications, it is not easy to determine *a priori* what attributes will be most relevant to the classification problem at hand. Often, attributes are dictated by a given application or already supplied in the database. Yet these attributes may not be the most relevant for the task. Such situations require a method for generating new attributes, and selecting from them the most relevant ones (e.g. Bongard[18]). This type of problem is considered in *constructive induction* (Michalski[11]). A recently developed program for constructive induction, AQ17, can generate new attributes by combining initially given attributes in many different ways, by creating complex relationships among them, or by obtaining advice from an expert (Bleodorn *et al.*[19]).

3. LEARNING PROBLEMS AND A GENERAL DATA TABLE

The above described learning problems can be simply illustrated by means of a '*general data table*' (GDT). Such a table is a generalization of a typical data table used in data analysis (Figure 4). The columns in a GDT correspond to attributes. These may be some initial attributes, given *a priori*, or additional ones generated through a process of constructive induction. Each attribute is assigned a *domain* and a *type*. The domain specifies the set of all legal values that the attribute can take on in the table, which includes '?' ('unknown') and N/A ('not applicable'). The type defines the order of the values in the domain (the scale). For example, the AQ15 learning program (Michalski *et al.*[9]) allows three types of attributes: nominal (no order), linear (total order), and structured (a hierarchical order). The attribute type determines the kinds of operations that are allowed on this attribute during a learning process.

Rows in the table correspond to individual entities that are being characterized by the attributes. An entry in the table can be a specific value of the corresponding attribute, the symbol '?', meaning that the value is unknown for the given entity, or the symbol N/A, if the given attribute does not apply to the given entity. For example, the colour attribute usually applies to physical objects, but does not apply to relations or abstract entities.

An important problem of data analysis is to determine whether some attribute in a table depends on some other attributes. A related problem is to determine a specific form of this relationship. A case of the latter problem is when is it known that a nominal-scale attribute

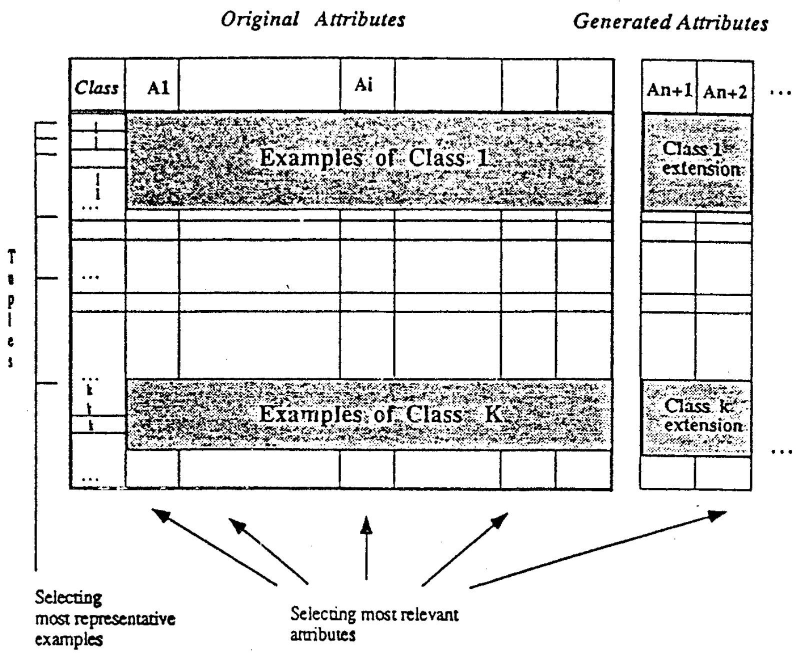


Figure 4. An illustration of the role of different symbolic operators

depends on other attributes, and the problem is to hypothesize a general description that characterizes this relationship. The latter problem is equivalent to a typical problem of concept learning from examples. In such a case, one column in the data table is designated to represent the *output attribute*. The values of that variable are classes whose descriptions are to be learned. In Figure 4, it is the first column that represents values of the output variable (in this case, the class of the examples). In conceptual clustering, there is no such column, as there are no *a priori* classes which entities belong to (as in any other 'unsupervised learning' problem).

Using a GDT, one can characterize the problems described above in a simple way as follows.

(i) *Learning from examples.* Classes of examples are sets of rows in the table which have the same value of the output attribute (variable). The problem is to determine a set of general rules characterizing these classes of examples.

(ii) *Example selection.* The problem is to select from the table rows that correspond to the most representative examples (rows) of different classes.

(iii) *Attribute selection.* The problem is to select columns that correspond to the most relevant attributes for characterizing given classes or the differences between classes.

(iv) *Generating new attributes.* The problem is to generate additional columns that correspond to new attributes generated by a constructive induction procedure. These new attributes are created by using the problem background knowledge, and/or special heuristic procedures (Michalski *et al.*[19]).

(v) *Conceptual clustering.* The problem is to split the rows of the table into groups of rows that correspond to 'conceptual clusters', i.e. sets of entities with high conceptual cohesiveness. An additional column is added to the table that corresponds to a new 'output attribute'. The values of this attribute in the table denote the proposed class of each entity. Rules that describe clusters are stored separately (see Section 5).

(vi) *Learning from imperfect data.* In this case, some entries of the table are missing, or are incorrect. The problem is to determine the best (e.g. the most plausible) hypothesis that accounts for the data, or for most of the data.

Methods for performing the above operations have been developed and implemented in various machine learning programs (see, e.g., Michalski;[20,21] Forsyth and Rada;[22] Kodratoff,[23] and Kodratoff and Michalski.[24]

4. EXAMPLES OF KNOWLEDGE GENERATION OPERATORS

As mentioned above, the programs for different forms of symbolic learning and knowledge transformation can be used as data analysis operators. These operators are applied to a data table and produce certain data characteristics. To illustrate this idea, we use examples of how a conceptual-clustering program and a learning-from-examples program can be used as data analysis operators (based on the material in Reference 10).

Suppose we are given a table as shown in Figure 5.

Suppose now that the task is to produce the 'best' conceptual classification of the data into two and three classes. The table in Figure 5 is an input to a conceptual clustering program, acting as a CLUSTER operator.

The results of applying this operator have two components: a new extended data table, and a set of rules. The new table has two additional columns; the first column indicates the class assigned to the tuples in the generated two-class clustering, and the second column indicates the class assigned to the tuples in the generated three-class clustering (Figure 6).

The second component consists of two sets of rules; the first ruleset describes classes in the

Microcomputer	Display	RAM	ROM	Processor	No_Keys
Apple II	Color_TV	48K	10K	6502	52
Atari 800	Color_TV	48K	10K	6502	57–63
Comm. VIC 20	Color_TV	32K	11–16K	6502A	64–73
Exidi Sorceror	B/W_TV	48K	4K	Z80	57–63
Zenith 118	Built_in	64K	1K	8080A	64–73
Zenith 1189	Built_in	64K	8K	Z80	64–73
HP 85	Built_in	32K	80K	HP	92
Horizon	Terminal	64K	8K	Z80	57–63
Challenger	B/W_TV	32K	10K	6502	53–56
O-S 11 Series	B/W_TV	48K	10K	6502C	53–56
TRS-80 I	B/W_TV	48K	12K	Z80	53–56
TRS-80 III	Built_in	48K	14K	Z80	64–73

Figure 5. Illustrating learning problems by a general data table

two-class clustering, and the second set describes classes in the three-class clustering (Figure 7).

Suppose now that we use the extended data table in Figure 6 as an input to a program for learning concept learning from example. Suppose that the parameters of the program are set so that the program should try to differentiate among the classes, i.e. to determine the simplest discriminant class descriptions (Michalski[11]). In this case, the program plays the role of the so-called DIFF operator. The results are shown in Figure 8.

INPUT						OUTPUT (added to input table)	
Microcomputer	Display	RAM	ROM	Processor	No_Keys	2-Group	3-Group
Apple II	Color_TV	48K	10K	6502	52	1	1
Atari 800	Color_TV	48K	10K	6502	57–63	1	1
Comm. VIC 20	Color_TV	32K	11–16K	6502A	64–73	1	2
Exidi Sorceror	B/W_TV	48K	4K	Z80	57–63	1	2
Zenith 118	Built_in	64K	1K	8080A	64–73	2	3
Zenith 1189	Built_in	64K	8K	Z80	64–73	2	3
HP 85	Built_in	32K	80K	HP	92	1	2
Horizon	Terminal	64K	8K	Z80	57–63	1	2
Challenger	B/W_TV	32K	10K	6502	53–56	1	1
O-S 11 Series	B/W_TV	48K	10K	6502C	53–56	1	2
TRS-80 I	B/W_TV	48K	12K	Z80	53–56	1	1
TRS-80 III	Built_in	48K	14K	Z80	64–73	1	1

Figure 6. An extended table generated as a result of the CLUSTER operator

Rules characterizing the generated 2-class clustering:

[Class 1] ⇐ [RAM = 32K..48K]
[Class 1] ⇐ [No_Keys $\leqslant$ 63]

[Class 2] ⇐ [RAM = 64K] & [No_Keys > 63]

Rules characterizing the generated 3-class clustering:

[Class 1] ⇐ [Processor = 6502 v Z80] & [ROM = 10K..14K]

[Class 2] ⇐ [Processor = 6502A v 6502C v HP]
[Class 2] ⇐ [ROM = 1K..8K] & [Display $\neq$ Built_in]

[Class 3] ⇐ [Processor = 8080A v Z80] & [ROM = 1K..8K] & [Display = Built_in]

Figure 7. Rules characterizing classes created by the CLUSTER operator

Rules for 2-class clustering obtained by the DIFF operator

[Class 1] ⇐ [Display or Built_in]
[Class 1] ⇐ [ROM $\geqslant$ 14K]

[Class 2] ⇐ [RAM = 64K] & [No_Keys = 64..73]

Rules for 3-class clustering obtained by the DIFF operator

[Class 1] ⇐ [Processor = Z80 v 6502] & [ROM = 10K..14K]

[Class 2] ⇐ [Processor = 6502C v 6502A v HP]
[Class 2] ⇐ [ROM = 4K..8K] & [Display = B/W_TV v Term]

[Class 3] ⇐ [ROM = 1K..8K] & [Display = Built_in]

Figure 8. Discriminant rules generated by the DIFF operator

Comparing the rules in Figure 7 with those on Figure 8 (note that the latter were generated without knowledge of the former), one can see that they are similar but not identical. The reason for the difference is that given a set of examples, there are usually many different ways of creating a general description of them. The rules in Figures 7 and 8 are both complete and consistent with all the examples in Figure 6, i.e. they classify these examples in the same way.

5. INTEGRATING DIFFERENT OPERATORS IN ONE SYSTEM

To make symbolic data analysis operators, such as CLUSTER or DIFF, easily available to a data analyst, it is desirable to integrate them into one system. This idea was implemented in the INLEN system (Kaufman *et al.*[25,10]). INLEN incorporates a whole spectrum of machine learning operators, and also conventional statistical data analysis operators. To facilitate the application of all the operators, INLEN combines relation *database* technology with a *knowledge base* technology. The database technology is used for storing and updating data tables, and the knowledge base technology is used for storing updating and applying rules.

A general diagram of INLEN is presented in Figure 9. The name is an acronym from **in**ference and **le**ar**n**ing.

INLEN offers to a data analyst the following three classes of operator.

(1) *Data management operators* (DMO). These operators are conventional data management operators used for creating, modifying and displaying relational tables.
(2) *Knowledge management operators* (KMO). These operators play a similar role to the DMOs, but they apply to the rules and other structures in the knowledge base.
(3) *Knowledge generation operators* (KGO). These operators perform symbolic and numerical data analysis operations on data. They are based on various machine learning and inference programs, on conventional data analysis techniques, and visualization operators for graphically displaying the results of analysis. The diagrammatic visualization method, described briefly above, is used for displaying the effects of symbolic learning operations on data.

The KGOs are at the heart of the INLEN system. To facilitate their use, the concept of a *knowledge segment* was introduced. A knowledge segment is a structure that links one or more tables from the database with one or more structures from the knowledge base. Such

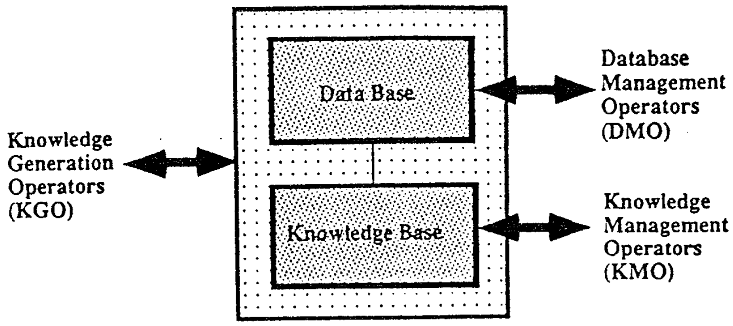

Data Base
Knowledge Base
Knowledge Generation Operators (KGO)
Database Management Operators (DMO)
Knowledge Management Operators (KMO)

Figure 9. A general diagram of the INLEN system for conceptual data analysis

knowledge segments are both the inputs to and outputs from KGOs. Thus, KGOs can be viewed as modules for performing complex inferences on knowledge segments in order to create new knowledge segments.

The execution of a KGO usually requires some background knowledge (BK) from the knowledge base, and is guided by various parameters. The BK specifies facts about the application domain, and provides information about attributes, the constraints and relationships among attributes, etc. The parameters specify how to choose an output description from among multiple candidates. KGOs can usually work in either incremental or batch mode. In the incremental mode, they try to improve or refine the existing knowledge; while in the batch mode, they try to create entirely new knowledge from facts in the database, using knowledge in the knowledge base.

KGOs in INLEN can be classified into several groups, based on the type of the output they generate. Each group includes a number of specific operators that are instantiated by a combination of the parameters.

(a) GENRULE operators generate various kinds of rules from given facts. They include operators that generate symbolic descriptions of data, e.g. generate rules characterizing a set of facts, discriminate between groups of facts, build decision trees, characterize a sequence of events, and determine differences between sequences. They also include operators generating equations qualitatively and quantitatively characterizing numerical data sets, and build conceptual hierarchies.
(b) TRANSRULE operators perform various transformations of the rules, e.g. generalize or specialize, abstract or concretize given rules.
(c) GENATR operators generate new attributes, or select the most representative attributes from a given set.
(d) GENEVE operators generate events, facts or examples that satisfy given rules, select the most representative events from a given set, determine an example that is similar to a given example, or predict a value of a given variable.
(e) ANAREL operators analyse various relationships that exist in the data, e.g. determine the degree of similarity between two examples, check if there is an implicative relationship between two variables, determine various statistical properties of the data.
(f) TEST operators test the performance of a given set of rules on an assumed set of facts. The primary output from these operators is a confusion matrix, i.e. a table whose (i, j)th element shows how many examples from the class i were classified by the rules to be in class j.

For more details about these operators the reader can consult the paper by Kaufman *et al.*[10] Summarizing, INLEN integrates a large range of operators for creating data, creating rules, and for performing many different symbolic and numeric operations on the data and/or rules.

6. CONCLUSION

An enormous growth of databases has created an intense need for developing a new type of data analysis tool that cannot only numerically but also conceptually characterize data. Such conceptual characterizations include symbolic descriptions, logical relationships and qualitative evaluations, as well as causal dependencies among the entities in the data. These characterizations represent knowledge that may more directly usable in human decision-making than numerical characterizations.

To determine such knowledge, a data analysis system has to be able to represent, and take into consideration, various kinds of background knowledge about the data and the domain of discourse. This background knowledge may include, for example, a specification of the domain and the type of the attributes, the relationships among them, causal dependencies, theories about the objects or processes that generated the data, and other high-level knowledge.

This main idea of the paper is that modern methods developed in symbolic machine learning can be a basis for the development of such conceptual data analysis tools. As examples, we described methods for learning general concepts from examples and conceptual clustering. We have also described an architecture of a large-scale system, INLEN, which integrates machine learning and statistical analysis techniques with database and knowledge base technologies.

The machine learning techniques allow the system to perform a whole range of symbolic data manipulation and knowledge extraction operations. These operations include, e.g., the generation of rules characterizing or discriminating between groups of facts, the creation of symbolic descriptions of sequences, the determination of new attributes, the building of conceptual hierarchies, and the construction of equations and logical preconditions for their application among others.

INLEN is a very complex system, and its development requires a significant amount of effort. Its experimental version is currently being implemented at the Center for Artificial Intelligence of George Mason University. Some of the INLEN components are based on programs already developed, and some require new research. Even those that are based on existing programs need to be properly modified and/or improved to be adequate, or sufficiently efficient, for the data analysis tasks. Further research is also needed to determine what other types of symbolic learning techniques might be applicable to data analysis.

ACKNOWLEDGEMENTS

The author thanks Michael Hieb, Ken Kaufman, and Elizabeth Marchut-Michalski for very useful comments and criticism. This research was done at the Artificial Intelligence Center of George Mason University. The activities of the Center are supported in part by the Defense Advanced Research Projects Agency under Grant No. N00014-87-K-0874, administered by the Office of Naval Research, and in part by the Office of Naval Research under Grant No. N00014-88-K-0226, No. N00014-88-K-0397 and No. N00014-91-J-1351.

This paper was read as an invited talk at the Fifth International Symposium on Applied Stochastic Models and Data Analysis, Granada, April 23–26, 1991.

REFERENCES

1. J. W. Tukey, *The Collected Works of John W. Tukey*, Vol. V, *Philosophy and Principles of Data Analysis: 1965–1986*, L. V. Jones (ed.), Wadsworth & Brooks/Cole, Monterey, California, 1986.
2. E. Diday (ed.), *Proc. Conf. on Data Analysis, Learning Symbolic and Numeric Knowledge*, Antibes, September 11–14, Nova Science Publishers, Inc., 1989.
3. C. Daniel and F. S. Wood, *Fitting Equations to Data*, Wiley, New York, 1980.
4. S. Morgenthaler and J. W. Tukey, 'The next future of data analysis', in *Proc. Conf. on Data Analysis, Learning Symbolic and Numeric Knowledge*, E. Diday (ed.), Antibes, September 11–14, Nova Science Publishers, Inc., 1989.
5. J. R. Quinlan, 'Induction of decision trees', *Machine Learning J.*, 1, 81–106 (1986).
6. R. S. Michalski, 'Learning flexible concepts: fundamental ideas and a method based on two-tiered representation', in *Machine Learning: an Artificial Intelligence Approach*, Vol. III, Y. Kodratoff and R. S. Michalski (eds), Morgan Kaufmann Publishers, 1990.
7. K. A. Kaufman, R. S. Michalski and A. C. Schultz, 'EMERALD 1: an integrated system of machine learning and discovery programs for education and research, user's guide', Rep. Machine Learning and Inference Lab., MLI 89-11, Center for Artificial Intelligence, George Mason University, Fairfax, Virginia, 1989.
8. K. A. Kaufman and R. S. Michalski, 'EMERALD 1: An Integrated System of Machine Learning and Discovery Programes for Education and Research: Programmer's Guide for the VaxStation,' *Reports of Machine Learning and Inference Laboratory*, MLI 90-14, Center for Artificial Intelligence, George Mason University, Fairfax, VA, December 1990.
9. R. S. Michalski, I. Mozetic, J. Hong and N. Lavrac, 'The AQ15 inductive learning system: an overview and experiments', ISG Rep., 86-20, UIUCDCS-R-86-1260, Department of Computer Science, University of Illinois, Urbana, 1986.
10. K. A. Kaufman, R. S. Michalski and L. Kerschberg, 'An architecture for knowledge discovery from facts: integrating database, knowledge base and machine learning in INLEN', in *Machine Learning and Inference Reports*, Center for AI, George Mason University, to appear in 1991.
11. R. S. Michalski, 'A theory and methodology of inductive learning', in *Artificial Intelligence*, 1983, pp. 111–161.
12. F. Bergadano, S. Matwin, R. S. Michalski and J. Zhang, 'Learning two-tiered descriptions of flexible concepts: the POSEIDON system', Rep. Machine Learning and Inference Lab., MLI 90-9, Center for Artificial Intelligence, George Mason University, Fairfax, Virgina, September 1990.
13. R. S. Michalski, 'A planar geometrical model for representing multi-dimensional discrete spaces and multiple-valued logic functions', ISG Rep. No. 897, Department of Computer Science, University of Illinois, Urbana, January 1978.
14. J. Wnek, J. Sarma, A. Wahab and R. S. Michalski, 'Comparing learning paradigms via diagrammatic visualization: a case study in single concept learning using symbolic, neural net and genetic algorithm methods', Rep. Machine Learning and Inference Lab., MLI 90-2, Center for Artificial Intelligence, George Mason University, Fairfax, Virginia, January 1990.
15. J. R. Quinlan, 'Probabilistic Decision Trees', in *Machine Learning: an Artificial Intelligence Approach*, Vol. III, Y. Kodratoff and R. S. Michalski (eds), Morgan Kaufmann Publishers, 1990, pp. 140–152.
16. R. S. Michalski, R. Stepp and E. Diday, 'A recent advance in data analysis: clustering objects into classes characterized by conjunctive concepts', invited chapter in *Progress in Pattern Recognition*, Vol. 1, L. Kanal and A. Rosenfeld (eds), 1981, pp. 33–55.
17. R. S. Michalski and J. B. Larson, 'Selection of Most representative Training Examples and Incremental Generation of VL1 Hypotheses: An Underlying Methodology and a Description of Programs ESEL and AQ11' Report No. 867, Department of Computer Science, University of Illinois, Urbana 1978.
18. N. Bongard, *Pattern Recognition*, Spartan Books, New York, 1970 (translated from the Russian).
19. R. S. Michalski, E. Bleodorn and J. Wnek, 'Inventing relevant attributes and relations in the AQ17 program for constructive induction', to appear in *Rep. Machine Learning Inference*, 1991.
20. R. S. Michalski, J. Carbonell and T. Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach*, Tioga Press, Palo Alto, CA., 1983.
21. R. S. Michalski, J. Carbonell and T. Mitchell (eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol. II, Morgan Kaufmann, Los Altos, CA., 1986.
22. R. Forsyth and R. Rada, *Machine Learning: Applications in Expert Systems and Information Retrieval*, Pitman, 1986.
23. Y. Kodratoff, *Introduction to Machine Learning*, Pitman, 1988.
24. Y. Kodratoff and R. S. Michalski (eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol. III, Morgan Kaufmann Publishers, 1990.
25. K. A. Kaufman, R. S. Michalski and L. Kerschberg, 'Mining for knowledge in databases: goals and general description of the INLEN system', *Proc. IJCAI-89 Workshop on Knowledge Discovery in Databases*, Detroit, Michigan, August 1989.