Presented at the UQÀM Summer Institute in Cognitive Sciences,
Montreal, June 30th-July 11th, 2003.

Inferential Theory of Learning and Inductive Databases

Ryszard S. Michalski,
Machine Learning and Inference Laboratory
George Mason University

and

Institute of Computer Science
Polish Academy of Sciences

1 Introduction

Amongst the most striking aspects of human learning is the ability to employ diverse learning strategies in an integrated, flexible, and goal-oriented manner. Given a learning goal, people are able to determine and apply a learning strategy, or a combination of them that is most suitable to achieving this goal[1]. People can learn from inputs in a vast array of forms, using different types of inference, and generate many kinds of knowledge represented in a boundless number of ways.

In contrast, most existing machine learning programs execute a single inferential strategy (defined by the primary type of inference) that employs a specific representational and computational method. For example, a decision tree learning program can, given examples of different classes, build a decision tree that inductively generalizes these examples. The input examples must be in the form of attribute-value vectors. They cannot be in the form of decision trees, graphs, nor complex relational descriptions. The output from the program is a decision tree. It cannot be a structural description, nor a semantic net, nor an analogy, nor a deductive consequence of the examples, etc. Such a program does not know what its learning goal is, nor why it is learning the decision tree. Its learning goal is defined implicitly by the way the program operates and by its output.

Similar limitations apply to rule learners, support vector machines, Bayesian learning, neural nets, and other popular learning and data mining programs (e.g., Mitchell, 1997; Kubat, Bratko, and Michalski, 1998; Paliouras, Karkaletsis, and Spyropoulos, Springer-Verlag, 2001). Such *monostrategy learning systems* not only do not model human learning adequately, but also lack flexibility in practical applications. They can work only for very narrowly and suitably defined learning problems. Machine learning research has clearly a long way to go to approach human learning capabilities.

---

[1] By a learning strategy we mean here a combination of the primary type of inference, a knowledge representation, and a computational method that is employed in a given learning process (Michalski, 1987, 1993).

The foundations for the development of *multistrategy learning systems*, capable of employing a range of learning strategies in an integrated and goal-oriented fashion, have been proposed in the

If knowledge is used to perform a task, then an improvement of this knowledge may lead to a better performance of that task. Therefore, a performance improvement is viewed in ITL as a consequence of knowledge improvement, rather than as a defining criterion of learning, as is often expressed in the literature. The central research topics in ITL are strategies, methods, and operators for knowledge generation and improvement, the analyses of their inter-relationships, the ways of integrating these strategies, methods for performing complex learning tasks, and, ultimately, the development of a general theoretical framework for characterizing all forms of learning, regardless of whether they occur in natural systems or in machines.

The objectives of this paper are to briefly elucidate the basic concepts of ITL, adding to them recently developed ideas, and to discuss their application to an important practical area, namely, the development of *inductive databases* and *knowledge scouts.*

2 What is Knowledge?

Because ITL views learning as a process of creating knowledge in the system, the concept of knowledge is fundamental to its development. The concept of knowledge is very old, and everyone seems to have an intuitive understanding of it. However, when people are asked to define what they mean by knowledge, they typically provide a vague and sometimes circular explanation. Therefore, before proceeding further, we begin with the definition of knowledge as used in ITL.

Recorded efforts to define knowledge go back at least to Socrates, who in Plato's Dialogues is credited with a view that "Knowledge was said by us to be true opinion" (Plato, Theaetetus, c.428- c.348, B.C.*).* Socrates' characterization of knowledge is that it is an opinion, not an observable fact, and that it needs be true, not just to be an arbitrary statement. To create an opinion, one has to reason. Hence, such a characterization implies a close link between knowledge and inference. A similar view of knowledge can be found in Antoine Arnauld's book of 1662, entitled "The Port-Royal Logic" in which he wrote, "Logic is the art of directing reason to knowledge of things for the instruction of both ourselves and others."

These characterizations of knowledge are consistent with a "computational definition" of knowledge that we proposed for ITL, that is:

"Knowledge is inference-enriched and validated information."

By "inference-enriched" is meant that, given information about some phenomenon, a learning agent relates it to its prior knowledge, and extends it by conducting inferences engaging that knowledge. All forms of inference may be used in this process, deductive (truth-preserving), inductive (falsity-preserving), and/or analogical (neither truth- nor

falsity-preserving). A deductive extension will produce a logical consequence of the information (which has the same truth-status as the original information). An inductive or analogical extension will produce a hypothesis that needs to be validated, by relating it to new facts and/or by conducting experiments. When the hypothesis reaches a sufficient level of confirmation, it is added to the agent's knowledge, that is, it becomes a "true opinion".

Knowledge has three aspects: *content*, *organization*, and *certainty*. By knowledge content is meant the mapping that the knowledge conveys. A specific function that maps a set to another set would constitute the knowledge content. The knowledge organization is the way this mapping is actually represented. If there are many different ways to represent the same function, then we say that they convey the same knowledge content, but differ in the knowledge organization. For example, the two pairs of implications (1) and (2):

$$\sim xz \vee \sim zy \vee \sim yz \Rightarrow A$$
$$\sim x\sim y\sim z \Rightarrow B \qquad (1)$$

and

$$\sim x\sim yz \vee \sim xy \vee xy\sim z \vee x\sim yz \Rightarrow A$$
$$\sim(x \vee y \vee z) \Rightarrow B \qquad (2)$$

represent exactly the same mapping, thus have the same knowledge content, but differ in knowledge organization. This is easily seen in Figures 1a and 1b that show visualize these expressions using a diagrammatic representation (Michalski, ???).

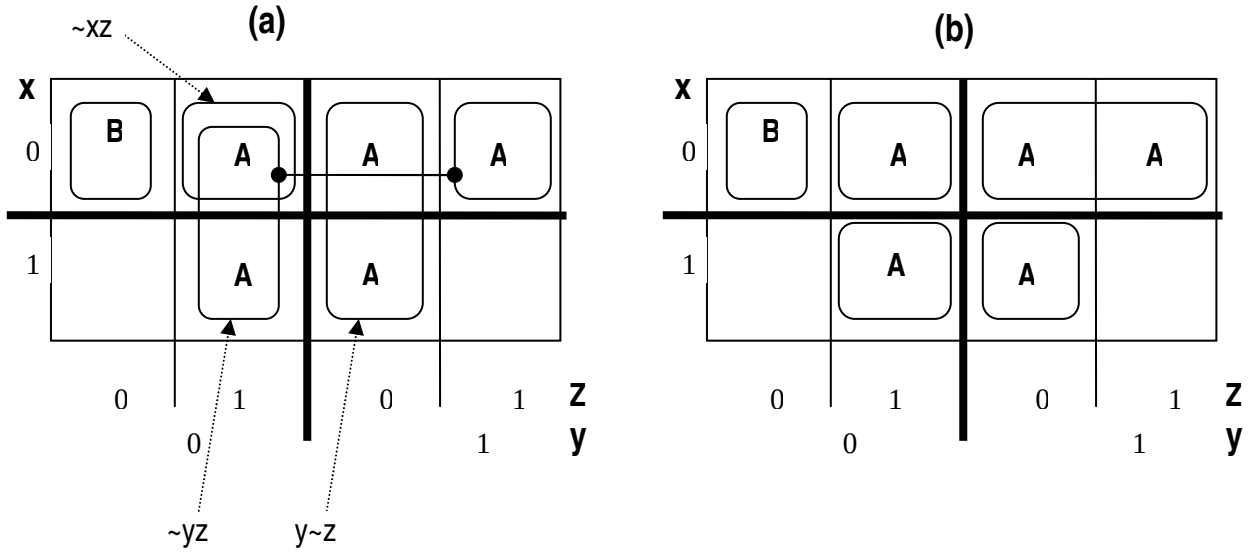

Figure 1. A diagrammatic visualizations of expressions (1) and (2).

Assume now that the expression ~x~y~z => B in (1) is replaced in the learning system by ~y~z. The unknown function value for the input "x~y~z" is now known, as it is assigned the value B. Such an operation increases knowledge in the system, thus it is a learning operation.

By the certainty of knowledge is meant a measure of the truth-relationship between the given knowledge and the reality it represents. If expressions (1) and (2) exactly characterize the real system they are supposed to describe, then the certainty measure would take the highest possible value on the truth-value scale that is used, for example, value 1, if the scale is [0,…,1]. If these expressions only approximate this function, then the certainty is a measure of function approximation.

3 Basic Concepts of ITL

As mentioned above, ITL considers learning to be a process of increasing an agent's knowledge. Frequently , by learning is meant an increase of the content of knowledge.

Such a process will typically involve conducting various forms of inference, validating their results, and memorizing the results for future use. In an extreme case, the inference can be reduced to simply memorizing input, as in rote learning. Learning can then be characterized succinctly by an "equation":

Learning = Inferencing and Memorizing

Given some information and a capability for performing basic forms of inference, deduction, induction, and analogy, a learner may potentially generate an unlimited number of inferences, which can potentially lead to new knowledge. Consequently, any practically realizable learning process must be guided by a goal that provides constraints on which inference paths to pursue and which to ignore.

Therefore, to characterize learning processes, ITL introduces the concepts of a *knowledge space*, *knowledge operators*, and a *knowledge goal*. A knowledge space is spanned over the concepts and the knowledge representations available to the learner. It is an abstract space of possible representations that includes the representation of knowledge already possessed by the learner (background knowledge), as well as of knowledge that the learner is capable of generating, using the available *knowledge operators*. A learning process can then be abstractly characterized as moving in the space from the point representing input knowledge (input information and background knowledge) to the point representing the learning goal. A learning goal is an abstract specification of the knowledge or skill that the learner wants to acquire, and is a necessary component of any learning process. It can be defined implicitly or explicitl`y, externally provided or internally generated, domain-independent or domain-dependent, one-time or recurrent (e.g., Ram and Leake, 1995).

Given an input, and some nontrivial background knowledge, a learner could potentially

generate an unbounded number of inferences (Ram & Hunter, 1992; Rieger, 1975). Many of these inferences, while "correct" in a purely logical sense, may not be useful in performing the overall tasks of the system. In fact, as has been demonstrated by several researchers, learning may sometimes even cause the performance of the system to deteriorate (e.g., Etzioni, 1990; Minton, 1990; Tambe, Newell, & Rosenbloom, 1990). To limit the proliferation of choices, and to ensure that the learning that occurs is actually useful

Knowledge operators are moves in the space that the learner is able to perform. Figure 1 presents a sketch of an imaginary learning process, presented as traversing an abstract knowledge space from the initial knowledge to the output (target) knowledge. Individual segments represent knowledge changes due to the application of some knowledge operator.

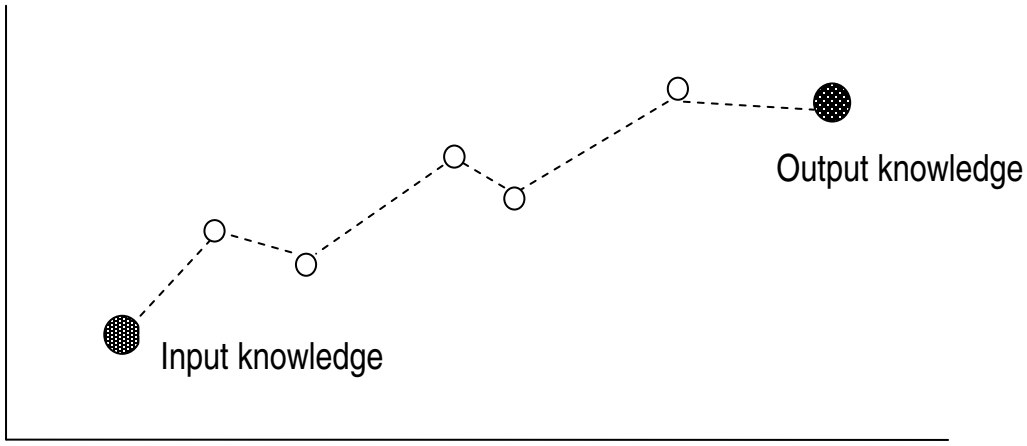

Figure 1. An imaginary learning process in a knowledge space.

To give a very simple example, suppose that the learner receives as input a collection of pairs <**x**, f(**x**)>, where **x** = (x_1, x_2, x_3,…, x_n) is a vector of values of variables x_1, x_2, x_3,…, x_n, and f(**x**) is the value of the function f for the given **x**. The collection (the training set) gives the value of f only for a subset of the function domain D (all possible vectors **x)**. The learner's background knowledge is that the function is symmetric (i.e., a permutation of its arguments does not change the function value). The learning goal is to hypothesize a complete description of the function f on the basis of the training examples and the background knowledge. The description should be expressed in a given language L, and be as simple as possible. It should assign a value to every vector in the function domain, with a probability of error less than some threshold T.

The knowledge space in the above example is the space of all possible functions, completely or incompletely defined over the function domain. The input knowledge is a point in the space that corresponds to a function defined by the training set (i.e., an incompletely-defined function f). The background knowledge restricts the knowledge space to a subspace of points corresponding to symmetric functions. The goal of learning is to find a point in the space that corresponds for a completely defined function f^*, which approximates f with an error rate below T over the domain D (assuming, for

simplicity, an even distribution of vectors in D). The error rate of the hypothesis usually cannot be measured; it can only be estimated by applying the hypothesis to another subset of the function domain (the testing set).

The above is an example of a generally defined problem of learning from examples (a.k.a. supervised learning). If the range of the function is a discrete set, then the above is an example of a classification learning problem, widely studied in machine learning. In these studies, L is often a language of decision trees or decision rules. If the range of the function is a continuous set, then the above is a function approximation problem, known in statistics as a regression problem. In machine learning, such problems were studied under the name of quantitative discovery. In regression analysis, L is assumed to be of a certain form, for example, a linear function (linear regression), or a polynomial function (non-linear regression). In quantitative discovery, L is much less restricted, e.g., it may be the language of arbitrary arithmetic expressions (e.g., Falkenheimer and Michalski, 1990).

In the case of a classification learning problem, knowledge operators are inductive generalization and specialization rules that define different ways of generalizing and specializing descriptions (Michalski, 1983; Bergadano, Giordana, and Saitta, 1991). In the case of a regression problem, knowledge operators are steps of regression analysis, or steps and heuristics in scientific discovery (Langley, Simon, Bradshaw and Zytkow, 1987). The learning strategy described above is called *batch inductive learning*. If the training examples are supplied in portions, and the learner improves the currently held hypothesis each time after receiving a new portion of the examples, this is called an *incremental inductive learning* strategy.

As an example of another strategy, a *deductive learning strategy*, assume that the input consists of a description of the function f in an abstract form. Suppose that the learning goal is to create an algorithm for efficiently computing the value of f(**x**) for each input **x.** If the value f(**x**) cannot be computed for a given **x** (the function is unspecified for that vector), then the output is "don't know." Vectors **x** may appear with different frequencies. In this case, the knowledge operators are different steps that the learner is able to apply to transform the initial abstract function description (for example, a formal mathematical equation) into an efficient computational algorithm for an execution on a given computer. Properties of the computer constitute the background knowledge (BK) for this learning process. The above described learning strategy is called deductive (a.k.a. analytical, or speed-up learning), because the output from a learning process is a deductive consequence of the input. Knowledge generated is a new form that is suitable for an efficient execution.

As illustrated above, different learning strategies are applicable to different learning tasks. A learning task is defined by the input knowledge, the learner's background knowledge, and the learning goal. To accomplish a given learning task, the learner executes various knowledge operators. Such operators are specific realizations of patterns of knowledge change, called *knowledge transmutations* or *knowledge transforms* (see Sec. 5).

A learning process is characterized in ITL as a transformation:

Given:
- Input knowledge (I)
- Goal of learning (G)
- Background knowledge (BK)
- Transmutations (T)

Determine:
- Output knowledge (O) that satisfies goal G, by applying transmutations from T to input I and the background knowledge BK.

Assuming that the learner is able to represent the input knowledge and the learning goal, its learning capabilities are directly dependent upon the background knowledge and upon the knowledge transmutations it is capable of performing. Both humans and monkeys can perform various mental functions, but the range of different types of knowledge transmutations that a human can perform, vastly exceeds those of a monkey. Therefore, from the same input stimuli, a human can potentially derive a much larger number of inferences than a monkey.

4 Types of Inference

Since learning processes may involve any type of inference, a complete learning theory must include a theory of inference. Therefore, we attempted to identify and classify all major types of inference (Figure 2).

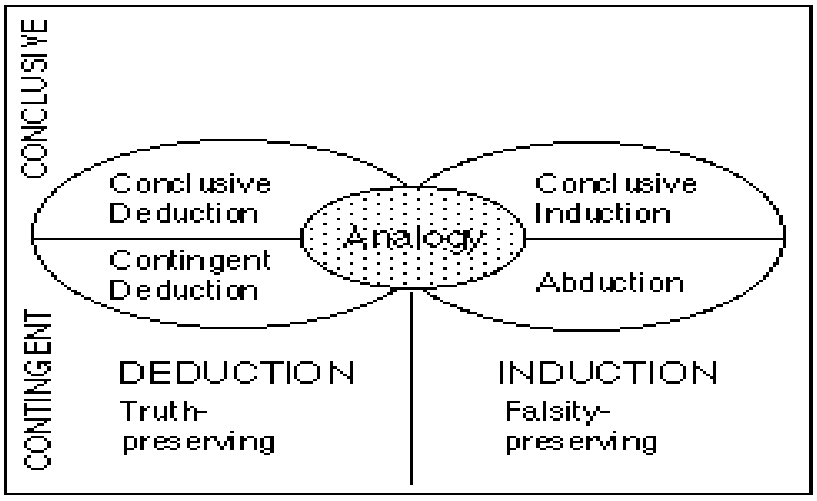

Figure 2. A classification of basic types of inference.

One classification criterion divides inferences into those that are deductive or inductive. Different authors define these inferences somewhat differently, so we explain below the way they are characterized in ITL. Consider what we call a *fundamental equation for inference*:

$$P \cup BK \models C \quad (1)$$

where P stands for the premise, BK for the reasoner's background knowledge, |= for semantic entailment, and C for the consequent.

To be more specific, consider P, BK and C to be sets of declarative statements (descriptions) in some language. Deductive inference is to derive C, given P and BK, and is truth-preserving. Inductive inference is to hypothesize P, given C and BK, and is falsity-preserving.

Another classification criterion divides inferences into conclusive (strong) and contingent (weak). Conclusive inferences involve domain-independent inference rules, whereas contingent inferences involve domain-dependent rules. For example, modus ponens, is a domain-independent rule of inference. Contingent deduction produces *likely* consequences of given causes, and contingent induction hypothesizes causes that *may* lead to given consequences. For example, consider a domain-dependent rule “If a car looses a wheel while driving, this may cause an accident.” If one sees that the car is loosing a wheel while driving, then a contingent deduction would be that it may have an accident. If one observes that a car had an accident, then a continent inductive hypothesis would be that the car lost a wheel.

Note: A conclusive induction generates hypotheses that, if true, will cause given consequences, not that they may cause these consequences. Analogy can be characterized as a combination of induction and deduction (Michalski, 1993).

5 Content-modifying Transmutations

Transmutations are classified by the type of inference they perform and the aspect of knowledge they change. ITL distinguishes between three aspects of knowledge, namely, *content*, *organization,* and *confidence*. Briefly, content is a function (a mapping) that knowledge conveys, organization refers to the form in which knowledge is represented and organized, and confidence is a measure of the correspondence between knowledge and the reality it refers to. Figure 3 presents a selection of content-modifying knowledge transmutations (based on Michalski, 1993).

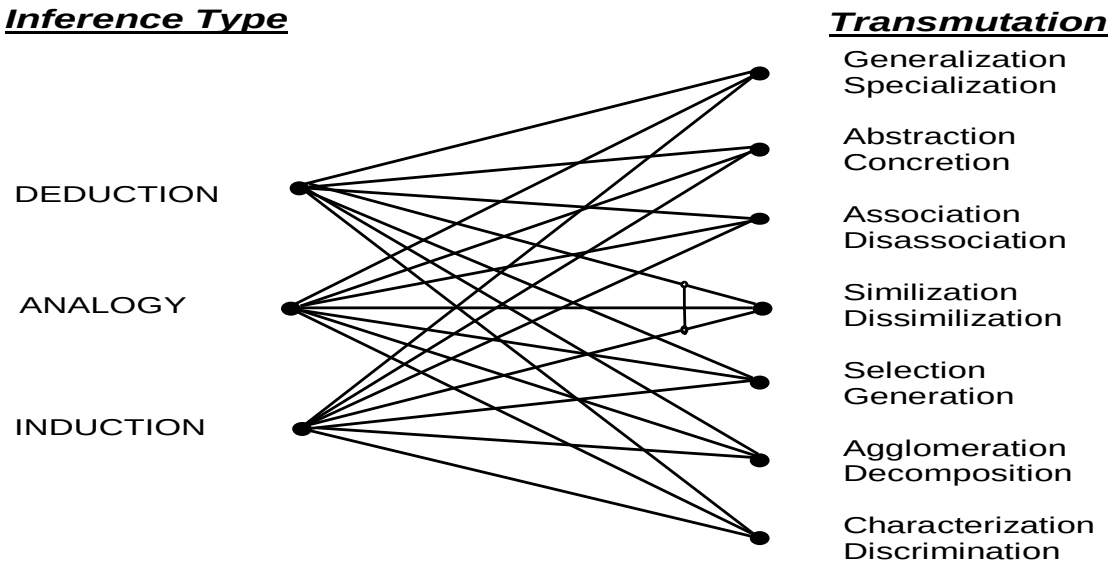

Figure 3. A selection of content-modifying knowledge transmutations.

Each link in Figure 3 represents a specific content-modifying transmutation. Such a transmutation is characterized by the primary inference involved (deduction, analogy or induction), and the type of knowledge that is changed or generated. The right column lists opposing classes of transmutations, e.g., {generalization, specialization}, {abstraction, concretion}, etc. Transmutations in each class can be classified on the basis of the types of inference being used to perform them.

A full analysis of the different transmutations lies outside the scope of this paper. For further details on transmutations, see (Michalski, 1993). Here, we will briefly focus upon one transmutation, namely, inductive generalization, which has been the central topic of most of machine learning research. The next section discusses inductive generalization transmutation in more detail and compares it to abstraction.

6 Generalization vs. Abstraction

To explain the ITL view of generalization and abstraction, we need first to explain the concept of a description. In ITL, a description is a statement, or a set of statements, in which three components can be distinguished, a *reference set*, a *descriptive schema*, and an *annotation*. The reference set is the set of entities (objects, processes, concepts, etc.) characterized or referred to by the description. The descriptive schema expresses knowledge about the reference set.

The annotation includes *merit parameters* that characterize the validity of knowledge expressed by the description, and a link to the context and relevant background knowledge (BK).

Here, we will use just one merit parameter, α, that estimates the probability that the description is true. Different merit parameters that affect human reasoning are discussed in (e.g., Collins and Michalski, 1989). In a description, some of the above components may be only implicit.

Consider, for example, a description: "A chair near the door is likely to be made of wood." The reference set is: "A chair near the door." The descriptive schema is: "made of wood." The hedge "likely to be" is a merit parameter. The context may be, e.g., the conversation in which this statement is expressed, and BK is the knowledge about concepts used in this statement. As another example, consider a statement: "The speed of light, c, in the equation $E = mc^2$ has been constant throughout the history of the universe." The descriptive schema is "The speed of light, c, in the equation $E = mc^2$ has been constant." The reference set is "the history of universe." The probability of this statement being true is a merit parameter. The value of this (implied) parameter is considered to be 1, although this belief has been recently questioned by the cosmologist João Magueijo of Cambridge University.

A generalization is a transmutation that enlarges the reference set of a description, without changing its descriptive schema. For example, the statement "All chairs near the door are made of wood" is a generalization of the previously mentioned statement about a single chair near the door. A reverse transmutation is a specialization.
An abstraction increases the granularity (or decreases the level-of-detail) of the descriptive schema without changing the reference set. For example, the statement "A piece of furniture near the door is made of wood" is an abstraction of the original statement about a chair near the door. An opposite transmutation is concretion.

As shown in Figure 3 by the links between the four transmutation classes mentioned above (generalization, specialization, abstraction, and concretion) and inference types, these transmutations can be performed by any of the three forms of inference (deduction, induction and analogy). When a generalization is performed by induction, it is called an inductive generalization.

According to ITL, generalization can be also performed by deduction, in which case it is called a *deductive generalization*. This type of generalization has been studied in explanation-based learning (Mitchell, 1997). Generalization can also be performed by analogy, in which case it is called an *analogical generalization*. For example, if given the statements: "John is an excellent programmer. He always completes his projects on time. Mike is similar to John.", one derives a statement: "Maybe Mike always completes his projects on time" then this would be an analogical generalization.

Allowing generalization to be performed by all three types of inference is an extension of a popular view in which generalization is always inductive. This popular view is probably due to the fact that inductive generalization is the most common and important form of

generalization, serving as the primary vehicle for hypothesizing new knowledge in every area of human activity.

Since abstraction reduces information about the reference set, the most typical form of abstraction is deductive. ITL also recognizes, however, a non-deductive abstraction, which refers to either inductive abstraction (when one hypothesizes an approximate abstraction from imperfect details), or an analogical abstraction (when one creates it by an analogy). Inductive generalization is helped by abstraction, because the latter removes details that differentiate entities, and thus allows them to be characterized by a general description. For this reason, abstraction is sometimes confused with generalization.

To illustrate simply the difference between generalization and abstraction in ITL, we employ a *knowledge packet*---a graphical representation of a description (Figure 4). A rectangle represents a descriptive schema, a circle represents a reference set, and an annotation is attached to the arrow linking the descriptive schema with the reference set. A shaded (larger) circle represents an enlarged reference set. A shaded (smaller) rectangle represents reduced information conveyed by the descriptive schema. Merit parameters, α, α_g, α_a, estimate the probability that the original description, a generalized description, and an abstracted description, respectively, are true.

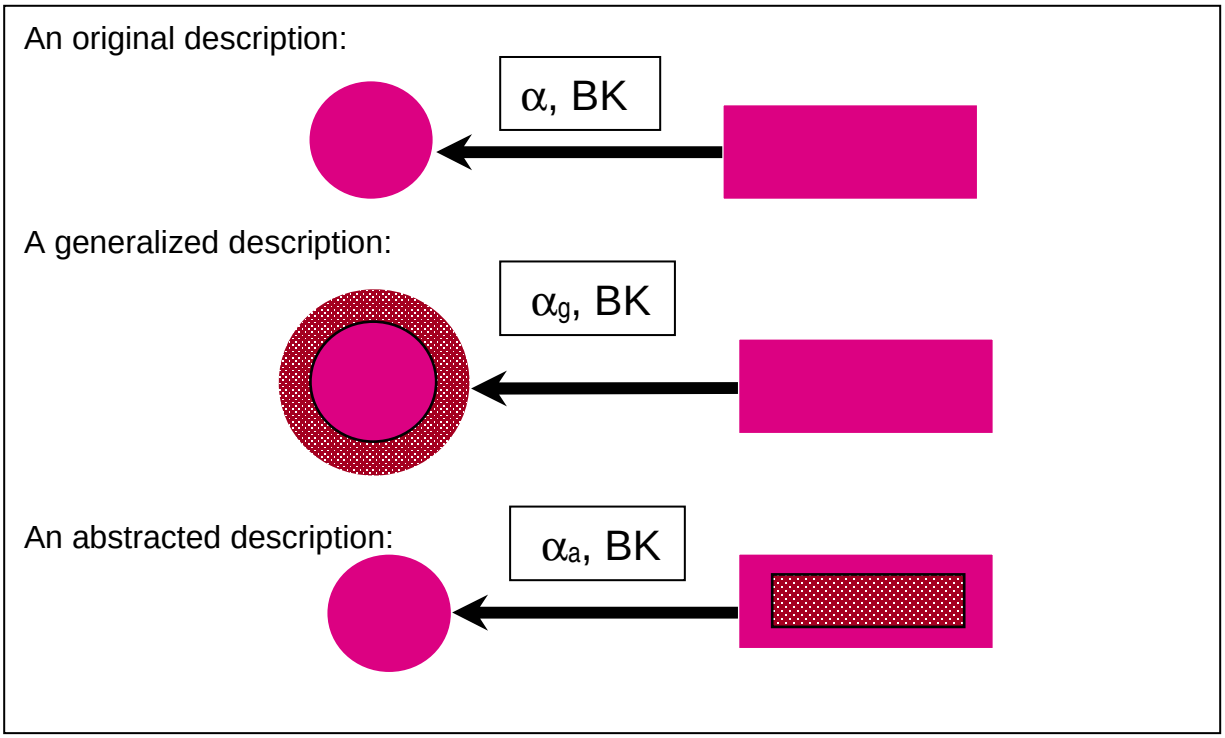

Figure 4. An illustration of the generalization and abstraction transmutations ITL.

BK in Figure 4 is a link to the relevant background knowledge and the context of the description. In the case of inductive generalization, $\alpha_g < \alpha_.$ In the case of deductive generalization, $\alpha_g = \alpha_.$ and in the case of abstraction, $\alpha_a = \alpha_.$

7 Learning Performance and Natural Induction

The fields of machine learning, pattern recognition and statistics have developed a range of methods for inductive learning (learning through inductive generalization). These methods include, for example, decision tree learning, rule learning, nearest neighbors, Bayesian nets, support vector machines, neural nets, regression analysis, inferential statistics, and others(e.g., Sharma,1996; Mitchell, 1997; Kubat, Bratko and Michalski, 1998; Scholkopf and Smola, 2002). A widely used measure of performance of these methods is the predictive accuracy, typically estimated by using a testing dataset, or by cross-validation.

The performance accuracy is usually represented by a single measure (the percentage of correctly and uniquely classified testing examples). It can also be represented by a vector of measures, which characterizes the performance in a more general setting, namely, when more than one decision is suggested by the system, or no decision (Michalski, 2003).

The predictive accuracy alone can be a sufficient measure of the quality of learning in applications when there is no need to understand the results. In many application domains, such as medical, agricultural, and economical decision making, engineering design, bioinformatics, business, defense, and others, the computer-created knowledge has to be not only accurate, but also to be easy to interpret and understand by people. This requirement reflects an observation that human experts are reluctant to employ computer-generated knowledge blindly, without understanding its meaning (Michalski, 1986). Although there is wide agreement on the importance of the understandability of the results from learning, this issue has so far received insufficient attention.

To explicitly address this issue, we introduced the concept of *natural induction*, defined as inductive generalization that strives to generate hypotheses that are both highly accurate and in forms “natural” for people, easy to interpret and understand. This requirement has it origin in the *postulate of comprehensibility* presented in (Michalski, 1983). What forms are natural is, however, a subjective and context-dependent matter. Therefore, it is inherently impossible to develop a universal, context-independent measure of “representation naturalness,” and a corresponding measure of knowledge understandability.

One can, however, make a simplifying assumption that “natural” knowledge representations are akin to those in which people typically like to represent scientific knowledge, such as structured natural language descriptions, special notations, and

visual forms. Research on natural induction requires developing a new measure for evaluating the performance of learning systems, namely, a description of cognitive complexity. Such a measure should reflect the complexity of interpreting and understanding learned descriptions.

To facilitate the development of natural induction systems, we developed *attributional calculus* (**AC**), a logic and representation system which combines elements of propositional logic, predicate logic and many-valued logic (Michalski, 2003). To facilitate natural induction, **AC** adds to standard logic operators, new operators and forms, which can significantly simplify some logic expressions and make them closer to their equivalent natural language expressions. Such operators and forms include internal disjunction and conjunction, counting conditions, and exceptions. Conventional decision rules, association rules, and N-of-M rules can be viewed as special cases of attributional expressions. **AC** has three forms that have an increasing representational power, core form, annotated form, and extended form, and two interpretation modes, crisp and flexible. The crisp mode interprets **AC** statements as true-false expressions, and the flexible mode interprets them as continuously-valued expressions. **AC** stems from *Variable-Valued Logic One* (VL_1), and serves as a basis for implementing advanced AQ-type inductive learning programs.

8 Application to the development of inductive databases

The inferential theory of learning has influenced efforts in several different research directions. These include, multistrategy task-oriented learning (Michalski, 1993), introspective multistrategy learning (Cox, 1996), and the development of inductive databases (Michalski and Kaufman, 2000). Here we will briefly review some of the recent results in the latter direction.

Inductive databases (IDBs) extend conventional databases by seamlessly integrating into them a wide range of *knowledge generation operators*. Due to these operators, IDBs can answer not only queries for which answers are stored in the database, but also queries requiring synthesizing plausible knowledge, generated by inductive inference from the facts in the database and prior knowledge. An IDB combines three technologies—databases, knowledge bases, and machine learning and knowledge discovery.

The Machine Learning and Inference Laboratory at George Mason University is currently conducting research on the development of an integrated inductive database and problem solving system, VINLEN. In VINLEN, inductive inference capabilities are combined with standard relational database operators through a *knowledge query language*. Specifically, knowledge query language integrates a standard database language, SQL, with a range of *knowledge generation operators*, which implement many different knowledge transmutations. The arguments of knowledge generation operators are *knowledge segments* that combine one or more relational tables with the related knowledge in the knowledge base. A knowledge generation operator takes as input one or more knowledge segments, and generates an output, which is also a knowledge segment.

VINLEN imposes two related constraints upon knowledge generation operators. Firstly, that the knowledge they generate be easy to understand and interpret, and, secondly, that it can be used efficiently in inference. These capabilities are achieved by applying ideas and methods of natural induction, which are being implemented in the concept learning program AQ20, the conceptual clusterer CLUSTER3, and a number of related programs. VINLEN's main objective is to provide an easy to use system for non-technically trained people that will help them to conduct complex analyses of data, search for knowledge of interest, and use the created knowledge for decision making. Figure 5 presents the Main Panel of VINLEN.

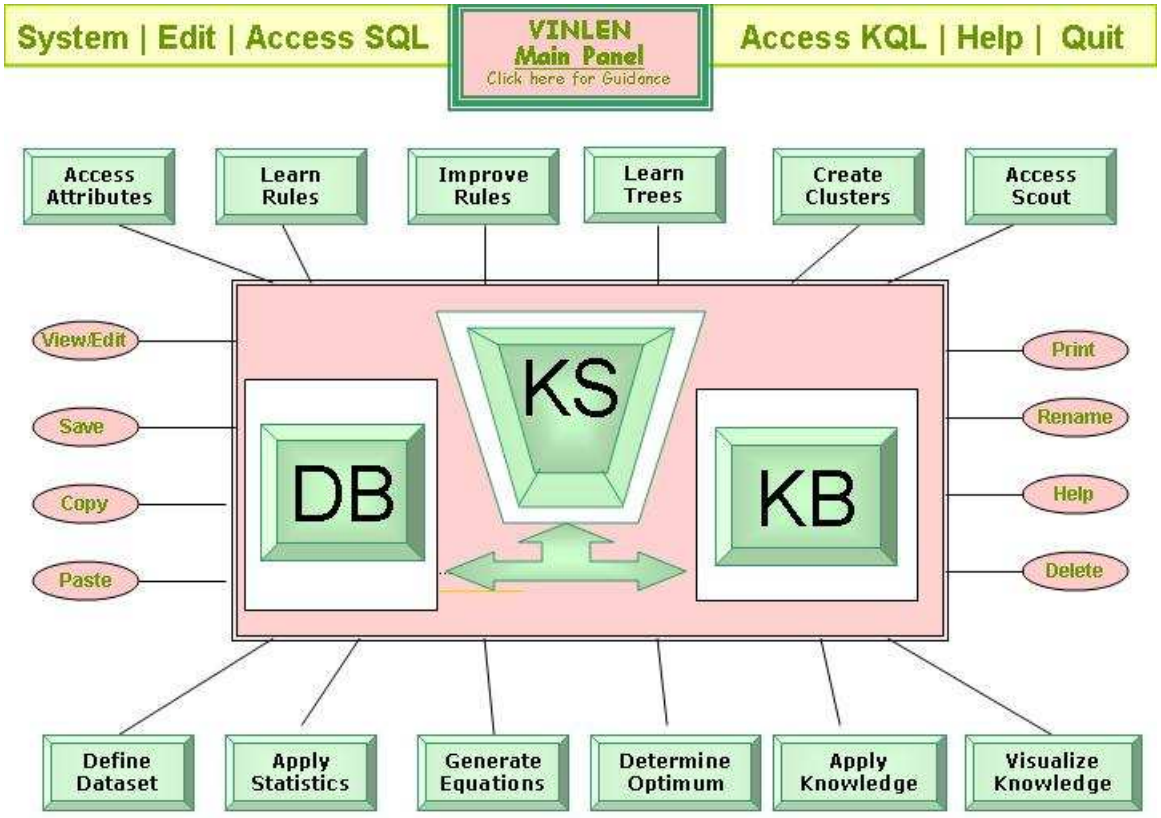

Figure 5. The main panel of VINLEN inductive database and problem solving system

The central buttons allow a user an easy access to available databases (DB), knowledge bases (KB), and *knowledge systems* (KS). A knowledge system integrates knowledge created by the system, or supplied to it, with a relevant data in the database, and supports problem solving. Buttons in the top and low row are menus that allow a user to invoke available knowledge generation operators. The top right button invokes capabilities for defining or executing *knowledge scouts*. Knowledge scouts are intelligent software agents that "live" in a database, and automatically search for knowledge of interest to the user, and/or employ the created knowledge in problem solving. Knowledge scouts are implemented as scripts in the knowledge query language.

Among already available features of VINLEN are the ability to generating different kinds of attributional rules from datasets, and visualizing these rules using concept association graphs. Figure 7 shows a concept association graph representing attributional rules, learned by VINLEN, for characterizing relationships among lifestyle factors and diseases of men 50-65.

Nodes of a concept association graph represent concepts (here, diseases and life style factors), and links represent relationships among these concepts. Conjunctive rules are indicated by arches linking conditions in the rules. Thicknesses of links characterize various measure of strength of relationships. For example, in one graph the thickness may represent the support, and in another graph the thickness may represent the confidence of attributional rules.

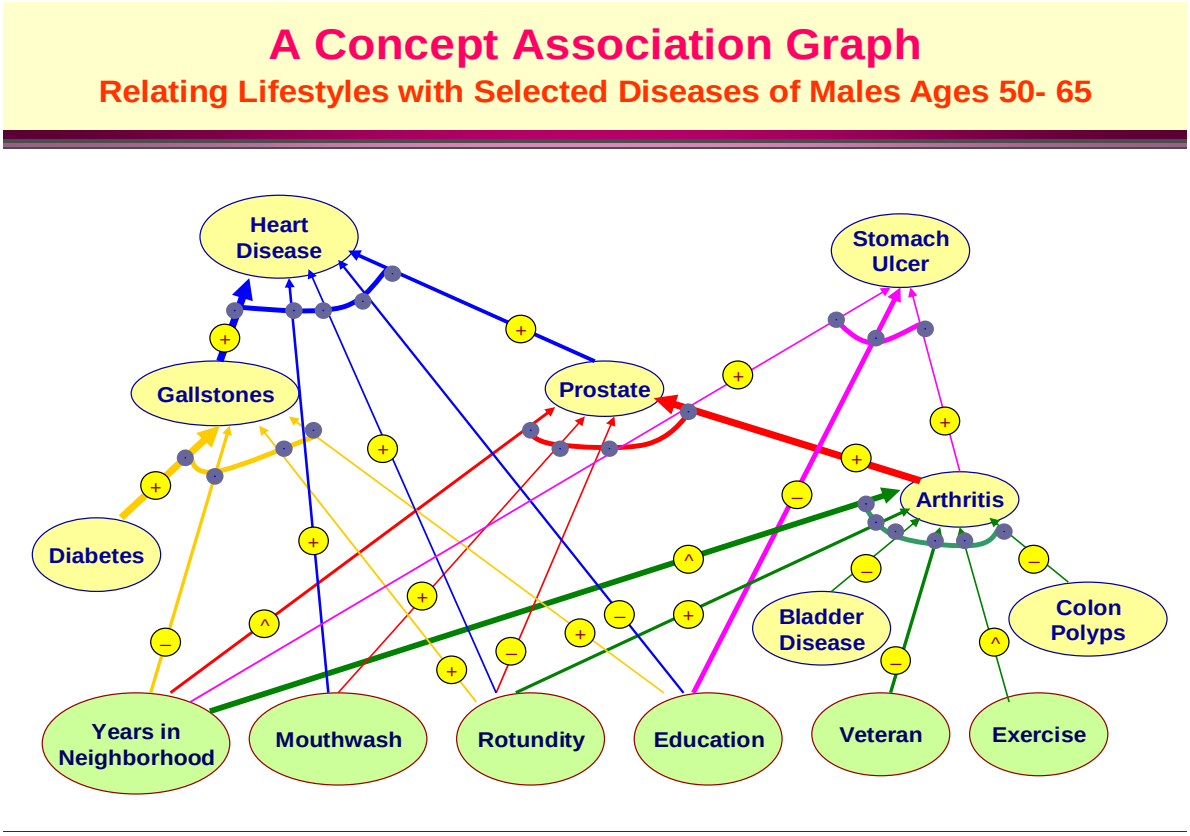

Figure 7. An example of a concept association graph.

9 Conclusion

The presented inferential theory of learning views any learning as a process of increasing knowledge in a system. Such an approach to learning provides a theoretical framework for characterizing processes in both natural and designed learning systems.

Inferential theory of learning has provided a basis for the development of an important new research direction concerned with implementing inductive databases and knowledge scouts. Inductive databases and knowledge scouts offer a powerful integrated tool for deriving knowledge from data and using this knowledge for problem solving.

Acknowledgements

The author thanks Ken Kaufman and Jaroslaw Pietrzykowski for collaboration on the development of VINLEN, and Mike Draminski for his contribution to the VINLEN implementation. This research was performed at the Machine Learning and Inference Laboratory at George Mason University. Laboratory's research activities are supported in part by the National Science Foundation under Grants No. IIS-9906858 and IIS-0097476, and in part by the UMBC/LUCITE #32 grant.

References

Alkharouf, N.W. and Michalski R.S., "Multistrategy Task-Adaptive Learning Using Dynamic Interlaced Hierarchies: A Methodology and Initial Implementation of INTERLACE," *Proceedings of the Third International Workshop on Multistrategy Learning (MSL-96)*, Harpers Ferry, WV, pp. 117-124, May 23-25, 1996.

Bergadano, F., Giordana, A.,and Saitta, L, *Machine Learning: An Integrated Framework and its Applications*, Ellis Horwood, 1991.

Collins, A. and Michalski R.S., "The Logic of Plausible Reasoning: A Core Theory," *Cognitive Science*, Vol. 13, pp. 1-49, 1989.

Cox, M. "Introspective Multistrategy Learning: Constructing a Learning Strategy under Reasoning Failure," Ph.D. Thesis, *Technical Report GIT-CC-96/06*, Georgia Institute of Technology, February, 1996.

Cristianini, N. and Shawe-Taylor, J., *An Introduction to Support Vector Machines,* Cambridge University Press, 2000.

Falkenhainer, B. and Michalski R.S., "Integrating Quantitative and Qualitative Discovery in the ABACUS System," *Machine Learning: An Artificial Intelligence Approach, Vol. III*, Y. Kodratoff and R.S. Michalski (Eds.), San Mateo, CA, pp. 153-190, Morgan Kaufmann Publishers, June 1990.

Kubat, M., Bratko, I. and Michalski R.S., "A Review of Machine Learning Methods," *Machine Learning and Data Mining: Methods and Applications*, R.S. Michalski, I. Bratko and M. Kubat (Eds.), pp. 3-69, London: John Wiley & Sons, 1998.

Langley, P., Simon, H.A., Bradshaw, G.L., and Zytkow, J. M., Scientific Discovery: Computational Explorations of the Creative Processes, The MIT Press, 1987.

Michalski R.S., "Toward a Unified Theory of Learning," *Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems*, Buchanan, B.G. and Wilkins, D.C. (Eds.), Morgan Kaufmann, 1993.

Michalski R.S., "Inferential Theory of Learning as a Conceptual Basis for Multistrategy Learning," *Machine Learning, Special Issue on Multistrategy Learning*, Vol. 11, pp. 111-151, 1993 (an extended version is in "Inferential Theory of Learning: Developing Foundations for Multistrategy Learning," in *Machine Learning: A Multistrategy Approach, Volume IV*, Morgan Kaufmann Publishers, 1994).

Michalski R.S. and Kaufman K., "Data Mining and Knowledge Discovery: A Review of Issues and a Multistrategy Approach," *Machine Learning and Data Mining: Methods and Applications*, R.S. Michalski, I. Bratko and M. Kubat (Eds.), pp. 71-112, London: John Wiley & Sons, 1998.

Michalski R.S. and Kaufman K., "Building Knowledge Scouts Using KGL Metalanguage," *Fundamenta Informaticae* , 40, pp 433-447, 2000.

Michalski, R.S., Bratko, I., Kubat, M., *Machine Learning and Data Mining: Methods and Applications*, John Wiley & Sons, 1998.

Mitchell, T., *Machine Learning*, McGraw-Hill Companies, Inc., 1997.

Paliouras, g., Karkaletsis, V., and Spyropoulos, C.D. (eds.), *Machine Learning and Its Applications*, Springer-Verlag, 2001.

Plato, *The Dialogues of Plato*, translated by B. Jowett, *Great Books of the Western World*, Encyclopedia Britannica, Inc., Chicago, 1952.

Ram, A., and and Leake, D. (eds.), *Goal-driven Learning*, MIT Press/Bradford Books, 1995.

Scholkopf, B. and Smola, A.J., *Learning with Kernels,* The MIT Press, Cambridge, MA, 2002.

Sharma, S., *Applied Multivariate Techniques*, John Wiley & Sons, Inc., 1996.