MACHINE LEARNING

One of the main concerns of computer science in general and artificial intelligence in particular is to understand, design, and implement computer systems that operate with different degrees of *intelligence* and *autonomy*. Such systems find a wide range of applications in many areas of science, engineering, medicine, and commerce. Examples of such *intelligent systems* that are in use today include automated theorem provers, game playing programs, automated diagnostic systems (e.g., for breast cancer diagnosis), customizable personal information assistants (e.g., software agents for selective information retrieval), credit risk analysis programs, data-driven scientific knowledge discovery and theory refinement systems, among others. Intelligent behavior in any domain requires adequate *knowledge* of the domain. For instance, a clinical diagnostic system needs rules that reliably identify diseases on the basis of observed symptoms. A personal information assistant must have knowledge about the needs and interests of the user to selectively, proactively, and reactively retrieve information (e.g., electronically available scientific articles) likely to be useful in a given context. A chess-playing program needs a great deal of knowledge to evaluate and choose among the moves available so as to maximize the likelihood of winning the game. A program for solving problems in mathematics or physics needs, in addition to general knowledge of the domain (e.g., rules of calculus or laws of physics), the skills to apply its knowledge effectively to solve the problems of interest efficiently.

In many domains, the necessary knowledge is not available in a form suitable for explicit programming into a *knowledge-based* system for solving problems. For example, in building an intelligent system for medical diagnosis, typically, we do not have a precise model of the human body's responses to various disease-causing agents. In this case, one might attempt to model the diagnostic behavior of expert physicians by eliciting the knowledge that they bring to bear when they diagnose. Then this knowledge is encoded in a form (e.g., if-then rules) that lends itself to use in automated inference. Traditional *expert systems* are examples of such systems which are typically built through a careful, tedious, and often expensive process of knowledge engineering. Knowledge engineering is the task of eliciting and codifying the knowledge of the domain expert in a form that can be used by the system. In practice, knowledge engineering presents some major difficulties. Experts are often unable (and sometimes unwilling) to articulate their reasoning process sufficiently precisely to be encoded in a form usable by a machine. Elicitation of knowledge from experts is time-consuming and expensive. Lack of precision, combined with the fact that knowledge is elicited from experts at different times (and perhaps in different contexts) often leads to internal inconsistencies in an evolving knowledge base.

Knowledge-based systems or *expert systems* for problem solving in narrow, specialized domains were among the first major commercial applications of artificial intelligence. The number of expert systems deployed grew from a mere handful in the mid-1980s to more than 10,000 in the early 1990s. Learning systems (i.e., programs that improve their performance on a task with experience) currently offer one of the most practical and cost-effective approaches to automated or semiautomated knowledge acquisition, thereby obviating the need for tedious and expensive knowledge engineering. The availability of vast amounts of potentially useful data on the Internet has led to an interest in intelligent agents that can be customized for different users. For example, an intelligent news reader can help selectively retrieve news articles that are of potential interest to a specific user. Because interests vary from one user to another, it is not practical to program such a system for each user. Hence, *intelligent agents* that learn a user's interests by observing their behavior are useful. All of this has led to increased interest in *machine learning*—the subfield of artificial intelligence primarily concerned with the design, analysis, implementation, and applications of *learning systems*.

Machine learning algorithms have been successfully used to acquire knowledge (e.g., in the form of predictive rules that capture observed regularities) from data in a number of different domains. Examples of problems where machine learning has produced knowledge that is competitive with that of human experts include diagnosis and credit risk assessment.

LEARNING DEFINED

Learning may be (informally) defined as the process of acquiring new knowledge; organizing the acquired knowledge into effective representations; developing perceptual, motor, and cognitive skills; and discovering new facts, hypotheses, or theories about the world through exploration, experimentation, induction, deduction, or abduction.

Researchers in machine learning often use a definition of learning that is motivated by pragmatic considerations. Learning is defined as any process by which a system quantifiably improves its *performance* on a *task or a set of tasks* with *experience* in some *environment.* Therefore, to make a specific machine learning problem well defined, it is necessary to define precisely the tasks, the performance measure, the nature of the experience, and the environment. For example, in the case of a chess-playing program, the task is to play a game of chess. A performance measure might be the proportion of games won against a well-specified population of players. Experience might be in the form of examples of games that were played (including the specific sequence of moves made by each player and the resulting board configurations), and environment might, in addition to providing the players to play against, include a teacher that occasionally offers advice or rewards the learner for making winning moves.

Therefore, the learners have to reason about their experience on a given task or set of tasks in their environments in the context of any relevant knowledge that they have about the task domain to acquire and store knowledge likely to improve their performance on the same or similar tasks in the future. In short, *Learning = Inference + Memorization*.

LEARNING CATEGORIZED

A broad range of learning scenarios and performance measures (e.g., speed, accuracy, robustness, efficiency) are conceivable. The learning scenarios can be classified into a number of (not necessarily disjoint) categories the basis of different criteria. Some of the criteria include the form of inference used in the learning process; the structure of the knowledge representation employed; the nature of interaction allowed between the learner and the environment (e.g. the nature of the experience available to the learner); and the domain of application (e.g., vision, language, robotics).

To provide a glimpse of the broad range of approaches available in machine learning, we start with a categorization of learning systems based on the primary inference mechanism used.

Rote Learning and Learning from Instruction

Rote learning systems operate by directly memorizing a piece of observed data and storing it in a form useful for later use. Consider an information retrieval task which involves consulting multiple distributed data sources to retrieve some piece of data of interest. The learner retrieves the relevant data after an expensive and time-consuming search process. If the task domain is such that the data does not change often (as in the case of physical constants, highway maps, etc.), the retrieved data can be memorized by the learner so as to avoid searching for it when the same information is sought in the future. This is a simple instance of rote learning. Rote learning is particularly effective in domains where the cost of computation (e.g., searching for data and retrieving it from remote sources) far exceeds maintaining the data in local storage and retrieving it on demand. Rote learning has also been used with considerable success to increase the effective depth of search used to select among alternative paths in problem solving and game playing.

Systems that learn by being instructed operate by taking advice from a teacher, transforming it if necessary into a form (e.g. a condition-action rule) suitable for incorporation into an existing knowledge base. Such systems can be quite effective in domains that provide knowledge-rich data sources. For instance, one could envision a learning system that extracts knowledge (using domain-specific natural language processing techniques) from sources, such as scientific dictionaries.

Analytic Learning or Learning from Deduction

Analytic learning systems in its pure form uses *deductive inference* (e.g., truth-preserving inference rules (such as *modus ponens* and *resolution principle* in first-order logic) to transform their experience on a task into a form useful for efficient performance of the same or similar tasks. A commonly used analytical learning strategy is *explanation-based learning*. For example, given a solution to a particular problem in integral calculus (perhaps discovered with time-consuming search of a suitably represented state space), an explanation-based learning mechanism generates an explanation from background knowledge (general rules of integration) showing how the solution deductively follows from what the system knows. Then the resulting explanation is used to reformulate the solution given to the particular problem into a form that makes it possible to recognize and solve similar problems efficiently in the future (for example, by reducing the amount of search effort needed to arrive at a solution because now the necessary knowledge is in a better usable or *operational* form). Other related forms of analytic learning include deductive derivation of abstractions or generalizations using facts provided by the environment and the learner's background knowledge.

If the background knowledge is incomplete, inconsistent, or imprecise, explanation generation, the key component of explanation-based learning cannot proceed without postulating changes in background knowledge. In such cases, it is often necessary to treat the knowledge base as though it were *nonmonotonic* and the derived explanations as though they were tentative and use a variety of nondeductive learning strategies for hypothesizing candidate revisions of the background knowledge. Other possibilities include nondeductive generalization of explanations as hypotheses to be validated by further experimentation with the environment.

Synthetic Learning or Inductive Learning

Induction is the primary inferential mechanism used in synthetic learning. Unlike deduction which cannot lead to fundamentally new knowledge (because all inferences logically follow from the assumed background knowledge and the given facts), inductive inference allows creating new knowledge from experience. A common example of an inductive learning task involves learning an unknown binary concept C given a set of examples. Each *example* is an ordered pair (I_k, c_k) where I_k is an *instance* represented in the chosen *instance description language* and c_k indicates the membership of the instance I_k in the concept C. Thus, c_k takes binary values in binary concepts. The system is also equipped with a *concept description language* which is used to express *candidate concept descriptions* or hypotheses H and a means of matching an instance I_k with a hypothesis $h \in \mathrm{H}$ to determine whether or not the instance belongs to h. Note that concepts are essentially compact descriptions (expressible in the chosen concept language) of subsets of instances that are expressible in the chosen instance language. The task of the binary concept learner is to learn the unknown concept $C \in \mathrm{H}$ from a set of labeled examples of C. An instance is labeled as a positive example if it belongs to the concept and as a negative example otherwise.

Thus, in a medical diagnostic task, the instances may be represented by a set of attribute values that denote the various observed symptoms, and the target concept can be a set of rules that output the appropriate diagnosis. It is easy to generalize the previous description of concept learning to multivalued (as opposed to binary) concepts or even real-valued functions from examples.

In the framework outlined before, the problem of concept learning reduces to a search for a concept description $C \in \mathrm{H}$ that *agrees with* or *approximates* the unknown concept C on the set of examples S. When the multiple candidate concept descriptions qualify this criterion, additional preference criteria (e.g., simplicity) have to be used to choose one such C. The choices of the concept description language and the instance description language constitute important representational commitments. The size of the search space with which the learner must deal is a function of these choices. Computational learning theory attempts to quantify the complexity of learning in terms of the number of examples and the running time of the learning algorithm. The feasibility of learning depends critically on the choice of languages (representations) used for describing instances and concepts.

Because concepts are descriptions of sets of instances, a variety of mechanisms can be used to search the space of candidate concepts. Examples of search algorithms include heuristic search from more specific to more general concept descriptions or from more general to more specific descriptions, bidirectional search, gradient-guided search, and evolutionary or genetic algorithms. For example, if the concept description language uses first-order logic predicates, inductive inference rules might include turning constants into variables, adding disjuncts, dropping conjuncts, and extending the domain of variables.

Rule Induction Techniques. These techniques employ *condition-action rules, decision trees, decision lists,* or similar knowledge representation structures. Information about classes or predictions are stored in the action components of the rules or leaves of the decision tree. A variety of rule-learning or decision-tree induction algorithms are available in the literature. Some advantages of such algorithms for knowledge acquisition include ability to extract knowledge in a form that is relatively easy to understand by humans; relatively low computational requirements; suitability for use in extending or refining existing knowledge in the form of decision trees or rules.

Artificial Neural Networks. These employ densely interconnected, massively parallel networks of simple computing elements (neurons) connected by weighted links to represent concepts or functions. Learning modifies the weights, the network connectivity (by addition and deletion of neurons and links), or both. Some advantages of neural networks include their robust performance in the presence of noise or missing attributes in the data. They are particularly effective for tasks, such as image classification. A broad range of neural networks (e.g., perceptrons, radial basis function networks, networks of sigmoid neurons) and algorithms (e.g., backpropagation and constructive learning algorithms) are available. Recent work has also focused on techniques for incorporating prior knowledge (e.g., in the form of approximate rules that describe a concept) into a neural network and techniques for extracting the learned knowledge from a network.

First-Order Rule Induction Techniques. These techniques employ relationships in first-order logic or logic programs (typically Prolog programs) to represent knowledge. Data is usually provided in the form of ground instances of relationships. Learning entails discovering logical expressions (typically, although not necessarily, single clauses) given the background knowledge and data. Some advantages of inductive logic programming include the ability to represent a wide range of relational knowledge (for which propositional rules of the sort encoded by decision trees do not offer the necessary expressive power) in domains, such as temporal reasoning and language processing, using a subset of first-order logic; ability to use background knowledge; and the ability to use the inferential capabilities of deductive databases. A practical disadvantage of rule induction methods based on *inverse resolution* is that they are susceptible to combinatorial explosion.

Automata Induction Techniques. These techniques represent knowledge in the form of finite state automata or state-transition networks, pushdown automata, or other computing devices that serve as language recognizers, or equivalently represent rules of grammar corresponding to a formal language (e.g., a regular language in the case of a finite-state automation). The input is in the form of sequences of variable length. Some advantages of automata induction include the ability to handle variable length sequences (e.g., strings over some alphabet, sentences in some language) which are often natural ways to represent temporal or sequential data. A variety of automata induction algorithms exist for induction of finite-state automata, recursive transition networks, context-free languages, and their stochastic variants. Such automata induction techniques are of interest in some scientific (e.g., characterizing the structure of DNA sequences) and situational assessment (e.g., learning the temporal structure of events) knowledge discovery applications.

Statistical Methods. These methods represent knowledge in the form of probabilistic rules, class-conditioned probability density on functions, or probabilistic inferential networks. Learning typically entails constructing a probabilistic model which allows computing the likelihood of each hypothesis of classification given the data. Such a *Bayesian* model of learning has several advantages including ability to handle noisy data; ability to use and refine existing knowledge; and optimal prediction (in the sense of minimizing the expected predictive error). Algorithms for induction of probabilistic networks and statistical pattern classification algorithms fall in this category.

Instance-Based Learning Techniques. These techniques store some or all of the examples in memory. Unlike most inductive learning methods, instance-based learning does not result in a concise description of the unknown concept. Instead, classification of an instance is usually based on calculating *distance* from the stored instances. For example, a nearest neighbor classifier assigns an instance to the classification of the nearest stored instance. A variety of distance metrics may be used depending on the instance representation. For example, if the instances are binary, Hamming distance would be a suitable distance measure. In the case of real-valued instances, Euclidian distance is commonly used. Instance-based learning is often called lazy learning because the learner seldom performs any inference during the learning phase. Almost all of the computation is done during the performance phase.

In addition to purely data-driven inductive learning approaches, there are techniques that build on and extend existing knowledge using training data. Other methods, in addition to passively processing the examples provided, actively interact with the environment or teacher by actively selecting instances and by querying the teacher.

Other Forms of Learning

Abduction involves generating candidate explanations for observed or otherwise given facts in terms of background knowledge and additional assumptions or hypotheses. The utility of such explanations is primarily to guide the search for useful theories about particular domains of interest. When abduction is used with analytic explanation-based learning it can help extend background knowledge by proposing and evaluating candidate explanations. Unlike induction and deduction, abduction in machine learning has not been widely explored yet abduction appears to play a central role in learning and discovery.

Analogy is mapping between two entities (objects, events, problems, behaviors, etc.). Relatively little is known about the formation of analogical mappings or analogical inference. Analogical inference appears to be an integral part of human reasoning so we mention it here for completeness.

SUMMARY

Learning structures and processes are essential components of adaptive, flexible, robust, and creative intelligent systems. Knowledge representational mechanisms play a central role in problem solving and learning. Indeed, learning can be thought of as the process of transforming observations or experience into knowledge to be stored in a form suitable for use whenever needed. Theoretical and empirical evidence emerging from investigations of learning within a number of research paradigms through a variety of mathematical and computational tools strongly suggests the need for systems that can successfully exploit a panoply of representations and learning techniques.

It is unlikely that a single knowledge representational scheme or a single reasoning or knowledge transformational mechanism would serve all of the system's needs effectively in a complex environment. For example, answering questions about a road map is better handled given an iconic or picture-like representation with a suitable set of operations to extract the necessary information than by using a bit-vector encoding that fails to make important spatial relations explicit. Similarly, a vision system capable of learning to recognize and describe complex three-dimensional objects at different levels of detail must have the representational structures at its disposal to do so efficiently. When multiple tasks need to be learned and performed in a coordinated manner (e.g., in the case of a robot interacting in a complex environment), it is necessary for the representations to be seamlessly integrated in a structured fashion. The choice of the right representation(s) to use depends heavily on the task(s) to be performed and the inferential mechanisms available.

ACKNOWLEDGMENT

This work was partially supported by grants from the National Science Foundation (NSF IRI-9409580) and the John Deere Foundation.

VASANT HONAVAR
Iowa State University

MACHINE LEARNING. See GENETIC ALGORITHMS; SOFTWARE QUALITY.

MACHINE VISION. See COMPUTER VISION.