Symbolic and Subsymbolic Learning for Vision: Some Possibilities

Vasant Honavar*
Department of Computer Science
Iowa State University
Ames, IA 50011-1040, USA

September 1, 1993

Abstract

Robust, flexible and sufficiently general vision systems such as those for recognition and description of complex 3-dimensional objects require an adequate armamentarium of representations and learning mechanisms. This paper briefly analyzes the strengths and weaknesses of different learning paradigms such as symbol processing systems, connectionist networks, and statistical and syntactic pattern recognition systems as possible candidates for providing such capabilities and points out several promising directions for integrating multiple such paradigms in a synergistic fashion towards that goal.

1 Introduction

No computational account of perceptual and cognitive processes — including vision — can be complete without a detailed specification of structures and processes that support *learning*. Learning, defined informally, is the process that enables a system to absorb *information* from its environment. Central to the more powerful types of learning is the process of incremental acquisition, modification, and consolidation of of adequate *internal representations* of the environment in which the learner operates. Learning (or evolution) must build the representations that perception and cognition utilize and modify (through learning). Computer vision offers a challenging testbed for exploring sophisticated representations and learning algorithms.

It can be argued that many learning tasks entail the discovery of useful mappings between representations. For example, learning to recognize and describe 3-dimensional objects requires the identification of a suitable mapping from the 2-dimensional visual image of the scene to words or sentences in a language that is expressive enough to describe the 3-dimensional structure of the object at a level of detail demanded by the tasks that the learner has to perform. This involves a search of a space of possible mappings permitted by the particular representations in question.

*This work was partially supported by Iowa State University College of Liberal Arts and Sciences.

The amount of search effort necessary is a good measure of the difficulty of learning, and it is invariably a function of the representations at the disposal of the learner. Recent results in computational learning theory reinforce this intuition. This suggests that successful learning in complex environments such as those that a child or a robot has to interact with require an adequate armamentarium of representations to choose from as well as learning algorithms that can exploit the strengths of available representations and/or readily and efficiently transform between representations as necessary.

Empirical and theoretical results in computational characterization of learning from several hithertofore disparate paradigms is a rich source of ideas for this endeavor. The following sections briefly analyze the applicability of different paradigms to recognition and description of complex 3-dimensional objects and outline some possibilities for their synergistic integration.

2 Symbol processing systems

Learning in symbol processing systems primarily involves acquisition of, and (inductive, deductive, abductive, analogical) inference using, knowledge in the form of complex symbolic data structures, and the memorization of the results of inference in a form suitable for use in the future. In short, *Learning* = *Inference* + *Memorization* (Michalski, 1993). This takes place in the context of background knowledge that the learner has, the environmental input, and the goals or needs of the learner. Such learning systems are prone to brittlenes (primarily due to the choice inflexible matching and inference strategies). Their primary strength is that they provide a range of powerful tools to reason with predominantly symbolic structured representations.

3 Connectionist networks

Learning in connectionist networks (massively parallel, shallowly serial, highly interconnected networks of relatively simple computing elements (neurons)) primarily involves on similarity-driven feature discovery, associative

storage and recall of binary or analog pattern vectors, induction of input-output mappings between boolean or real-valued vector spaces, and modeling of temporal dynamics of systems using input-output behavior sampled in time for prediction and control applications (Rumelhart & McClelland, 1986; Carpenter & Grossberg, 1990). The primary learning strategy is one of modifying a set of numerical parameters (e.g., weights) so as to minimize a suitably defined criterion function. In order for this to work, an acceptable solution must in fact lie in the space being searched and the particular search algorithm used must in fact be able to find it. Except in some restricted cases, there is no way to guarantee that an ad hoc choice of network architecture would be adequate for learning a mapping implicitly specified in terms of a set of sample input-output pairs. Generative or constructive algorithms (Honavar & Uhr, 1993a) attempt to discover an adequate network architecture for a given task by extending the search to the space of network topologies (which, unless suitably constrained, can easily get out of hand).

Much of the attraction of connectionist networks is due to their massive parallelism that is amenable to cooperative and competitive computation, potential for limited noise and fault-tolerance, discovery of simple, mathematically elegant learning algorithms for certain classes of such networks, and to those interested in cognitive and brain-modelling, their (albeit superficial) similarity with networks of neurons in the brain. Connectionist approaches generally fail to use prior knowledge that can potentially ease the burden on the learner. But recently, several researchers have begun to investigate the use of prior knowledge in the form of rules (Shavlik, 1993), specific inductive biases that can be captured in terms of constraints on network connectivity (Honavar, 1992b), and the organization of a task into more or less independent sub-tasks (Jacobs, Jordan, & Barto, 1990) to facilitate learning. The applicability of connectionist learning methods to complex tasks such as 3-dimensional object recognition and description is limited by the representation of input patterns as unstructured numeric vectors. Thus iconic or structural information (e.g., relative spatial arrangement of features in a 2-dimensional image) implicitly encoded in an input pattern is lost when such patterns are mapped in an ad-hoc fashion into numeric vectors to facilitate the use of standard connectionist learning algorithms.

With the exception of some recent work in connectionist architectures for variable binding and unification in slot and filler type representations, very little systematic study has been done on the representation and use (especially in learning) of structured objects such as frames (Minsky, 1975), schemas (Arbib, 1993) and conceptual graphs (Sowa, 1984) in connectionist networks. One possibility is to use structured object representations of the sort developed in the context of syntactic pattern recognition (see below) along with suitably defined similarity metrics with generative learning algorithms. Such systems extract, abstract, and encode instances of complex structured objects as necessary and use parameter modification algorithms to fine-tune the representations thus constructed (Honavar, 1992a).

4 Statistical pattern recognition

Learning in statistical pattern recognition systems (Duda & Hart, 1973; Fukunaga, 1990) is a form of inductive learning. It is typically assumed that the objects in the domain of interest can be distinguished among each other based on a set of measurements. Learning involves the analysis of the statistical distribution of the measurement vector $\mathbf{X}$ for different classes of objects in the domain of interest using a training set of measurement vectors with known class memberships.

Once the properties of different object classes are learned in terms of the distribution of the measurement vectors in $\Re^n$, classification of a novel sample is conceptually rather straightforward: Simply assign the sample to the class that is most likely given the measurement vector for the sample. This is usually accomplished by defining suitable ***discriminant functions*** that divide the n-dimensional space into regions that correspond to different classes of objects (or equivalently, patterns). There are basically two broad types of statistical learning: *parametric* methods which assume a particular mathematical form for the unknown density functions; and *non-parametric* methods that don't make such an assumption (nearest neighbor classification is a common example of a non-parametric method).

Current applications of statistical learning methods suffer from the limited expressive power the measurement vector based representation which fails to capture efficiently the *structural* relationships among the various parts of objects in the domain. Clearly, alternative frameworks for knowledge representation are worth investigating for use with statistical learning methods (see below). Also, it must be pointed out that statistical methods (like their connectionist counterparts), by their very nature rely on the availability of a sufficiently large number of samples to come up with good estimates of the distributions of interest. This limits their applicability in a number of practical learning scenarios where samples are scarce or expensive.

5 Structural Pattern Recognition

Syntactic or *structural* pattern recognition methods (Fu, 1982; Miclet, 1986) emphasize the structural relationships among the components of patterns in the domain of interest. Thus, if the domain of interest consists of curves in a 2-dimensional plane, we can define a finite alphabet V_T, each of whose letters corresponds to a direction. Thus, we can use an 8-letter alphabet to represent 8 discrete

directions. Any curve can be represented in discrete form as an ordered sequence of suitable letters chosen from V_T — that is, a string X ($X \in V_T^*$ — the set of all possible strings over the alphabet V_T). Pattern classes then correspond to subsets of $V_T{}^\star$ that share the desired properties. The structure of complex patterns (wholes) are described in terms of (possibly hierarchical) compositions of (simpler) sub-patterns (parts).

The simplest sub-patterns used to describe patterns or objects in a domain of interest are called *pattern primitives*. The *language* that provides the structural description of patterns in terms of a set of pattern primitives and their rules of composition may be thought of as the *representation language*. The rules governing the composition of primitives into patterns constitute the *grammar* of the representation language. For example, a class of patterns C_i (say houses) may be described in terms of a finite set of easily recognizable pattern primitives and a grammar G_i that specifies the possibly infinite number of patterns of that class. By placing different restrictions on the form of rules of the grammar, we get different classes (e.g., *regular*, *context-free*, and *context-sensitive*) of grammars. Recognition of patterns is reduces to parsing sentences.

A number of special languages have been proposed for description of special classes of patterns such as hand-written characters, chromosome images, line drawings, three-dimensional objects, etc. (Rosenfeld, 1981; Pavlidis, 1977; Winston, 1975). When we allow more general classes of n-ary relations, we can define a wide range of high-dimensional pattern grammars. For example, a simple 2-dimensional generalization of string grammars gives *array grammars*. Other generalizations include *web grammars*, *tree grammars*, *plex grammars* and *graph grammars*.

Assuming that a satisfactory solution for the selection of pattern primitives and the task of recognizing grammatical sentences can be found for a given application, learning to recognize classes of patterns reduces to learning the corresponding grammars (or equivalently the corresponding automata) - the grammar inference problem (see Fu, 1982; Miclet, 1986; Parekh & Honavar, 1993 for details). In many practical problems, it becomes necessary to supplement classical grammars which are purely syntactic in nature with *semantic* information. *Attributed grammars* offer such a model in which semantics of symbols can be learned in terms of functions that map a set of atrribute values into other attribute values (see Fu, 1982; Honavar, 1993 for details).

As a possible alternative to using the rules of a grammar, the process of pattern recognition may be formulated as a process of *template matching* wherein a pattern is matched (using a suitable *similarity metric* or *distance measure* against one or more stored patterns or template(s) or prototype(s) or reference pattern(s) for each of the pattern classes of interest and assigned to the pattern class with the best match. In this case, learning reduces to the acquisition of a suitable collection of templates (and perhaps selection of suitable similarity metrics) adequate for the task.

It is natural to generalize the idea of edit distance (Levenshtein, 1966) to arbitrary structured representations (Goldfarb, 1990; Honavar, 1992a). Consider a representational framework $\mathcal{R}$ (e.g., a space of appropriately defined vectors, strings, trees, graphs). Let I_1 and I_2 be objects represented in $\mathcal{R}$. That is, if $\mathcal{R}$ is the space of strings defined over a specified alphabet say, V_T, then $I_1, I_2 \in V_T{}^\star$. Key to the notion of generalized distance measure is a set of *elementary operations*, each with its associated non-negative cost that can be used to transform I_1 into I_2. The *generalized distance* between I_1 and I_2, $\Delta_{\mathcal{R}}(I_1, I_2)$ is given by the *minimum cost* (or some heuristic approximation of the minimum cost when an exact calculation of the minimum is not feasible for computational reasons) of transforming I_1 into I_2. This notion of generalized distance allows us to specify classes of inductive learning algorithms for families of representations.

Distance computation algorithms not only provide a degree of mismatch (or conversely similarity) between two objects I_1 and I_2, but also identify the sequence of transformations that would make I_1 identical to I_2. Thus we get for free, the ability to describe how two objects differ from each other. This opens up the possibility of interactive learing systems which can seek and use a rich variety of feedback from a teacher than is typically provided in empirical inductive learning. For example, the feedback might inform the learner how the object in question differs from a familiar object.

Tree-based representations permit efficient encoding of hierarchical relations among successively complex subpatterns. Quad-trees, oct-trees, and more generally k-trees and pyramids are widely used in computer vision (Rosenfeld, 1984; Uhr, 1987; Tanimoto & Klinger, 1980). This opens up the possibility of using tree grammars and edit distances between trees for learning in such domains.

Distance between two graphs can be defined in a manner analogous to that in the case of strings and trees as the minimum cost of transforming one graph into another although practically useful yet computationally tractable heuristics for computing distance between graphs are only beginning to be explored.

The template matching approach to pattern recognition is reminiscent of exemplar-based or case-based reasoning in artificial intelligencei. It is possible to adapt much of the work in exemplar-based learning to structured templates.

It is rather straightforward to adapt generative or constructive learning algorithms developed in the context of connectionist networks (see above) to add new structured templates through a feedback-guided process of extraction and abstraction of patterns and subpatterns from the environmental input once suitable similarity or distance measures are defined and efficient algorithms for computing such distances are specified (Honavar, 1992a).

Many of the generalization and abstraction algorithms

used in symbol processing approaches can find use in the modification of acquired templates. Learning can also modify the non-negative weights associated with particular transformations so as to reduce (increase) the distance between an input pattern and a stored template if the feedback suggests they are similar (dissimilar) as done in the case of some connectionist learning systems.

Another interesting possibility is for learning to extend the set of primitives used by adding potentially useful or meaningful compositions of existing primitive patterns. The primitives so added have an effect analogous to extending the vocabulary of natural language by defining new terms or concepts. Similar *chunking* mechanisms have been explored in symbol processing systems (Laird, Rosenbloom, & Newell, 1986; Uhr & Vossler, 1963), generative learning algorithms for connectionist systems (Honavar & Uhr, 1993a) and in some evolutionary learning systems (Koza, 1992).

It is possible to combine the template–based and grammar–based approaches to syntactic pattern recognition. For example, a pre-specified or learned grammar G_i for a class of patterns C_i can be treated in a manner analogous to a template for the class. One can then define the distance between a given string γ and the grammar G_i as the minimum cost of transforming the string γ so as to order to render it syntactically acceptable by the grammar G_i This distance can be used to assign patterns to one of several candidate classes.

Our discussion of grammars has relied on an implicit assumption, namely that every pattern belonging to the language $L(G)$ is equally significant in the definition of the grammar G. In practice however, some patterns are more likely to appear than others. This is easily modeled using *stochastic grammars* which associate a probablity with each of the rules of the grammar. These probabilities can be estimated from samples using standard statistical estimation methods (Fu, 1982). Stochastic extensions of attributed grammars present additional possibilities that remain to be explored.

6 Summary and Discussion

Learning structures and processes are essential components of adaptive, flexible, robust, and creative intelligent systems. Knowledge representation 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, using 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 representation scheme or a single reasoning or knowledge transformation 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 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 3-dimensional objects at different levels of detail must have at its disposal the representational structures 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 inference mechanisms available.

We have examined some of the promising directions for synergistic integration of symbolic and subsymbolic learning techniques from closely related but hithertofore largely disparate lines of research in artificial intelligence, connectionist networks, statistical and syntactic methods in pattern recognition. Of these, traditional artificial intelligence and syntactic pattern recognition provide a number of powerful tools and techniques for manipulating predominantly symbolic structured representations while statistical pattern recognition and connectionist networks provide the potential for robustness and fault-tolerance that comes with the use of numeric weights and statistics. A careful examination of these paradigms suggests that there is really no fundamental incompatibility between them (Honavar & Uhr, 1993b; 1993c). The space of systems resulting from their systematic integration is full of potentially interesting possibilities. A priori it is difficult to determine exactly which subsets of these would be appropriate for a broad range of computer vision systems. To answer this question effectively, extensive empirical and (whenever feasible) theoretical studies are needed with a wide range of such systems (for some preliminary work on such systems, see Honavar, 1992c; Yang & Honavar, 1993) on a range of problems.

7 References

Arbib, M. A. (1993). Schema Theory: Cooperative Computation for Brain Theory and Distributed AI. In: *Symbol Processors and Connectionist Networks in Artificial Intelligence and Cognitive Modelling.* Honavar, V. & Uhr, L. (ed). New York: Academic Press. To appear.

Buchanan, B. G., & Wilkins, D. C. (1993). Readings in Knowledge Acquisition and Learning. San Mateo, California: Morgan Kaufmann.

Carpenter, G., & Grossberg, S. (1991). (Ed). Pattern Recognition by Self-Organizing Neural Networks. Cambridge, Massachusetts: MIT Press.

Duda, R. O., & Hart, P. E. (1973). Pattern Classifica-

tion and Scene Analysis. New York: Wiley.

Fu, K. S. (1982). Syntactic Pattern Recognition and Applications. Englewood Cliffs, New Jersey: Prentice-Hall.

Fukunaga, K. (1990). Introduction to Statistical Pattern Recognition. New York: Academic Press.

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. New York: Addison Wesley.

Goldfarb, L. (1990). An Evolving Model for Pattern Learning. *Pattern Recognition 23*, 595-616.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. Ann Arbor, Michigan: University of Michigan Press.

Honavar, V. (1992a). Inductive Learning Using Generalized Distance Measures. In: *Proceedings of the SPIE Conference on Adaptive and Learning Systems.* Orlando, Florida.

Honavar, V. (1992b). Some Biases for Efficient Learning of Spatial, Temporal, and Spatio-Temporal Patterns. In: *Proceedings of the International Joint Conference on Neural Networks.* Beijing, China.

Honavar, V. (1992c). Learning Parsimonious epresentations of Three-Dimensional Shapes. In: *NATO Advanced Research Workshop on Mathematical Representations of Shape*, Driebergen, Netherlands. To appear.

Honavar, V. & Uhr, L. (1993a) Generative Learning Structures and Processes for Generalized Connectionist Networks. *Information Sciences* (Special Issue on Artificial Intelligence and Neural Networks) 70 75-108.

Honavar, V. (1993). Learning Systems That Use Multiple Strategies and Multiple Representations. In: *Symbol Processors and Connectionist Networks in Artificial Intelligence and Cognitive Modelling.* Honavar, V. & Uhr, L. (ed). New York: Academic Press. To appear.

Honavar, V., & Uhr, L. (1993b). Symbol Processing Systems, Connectionist Networks, and Beyond. In: *Intelligent Hybrid Systems.* Goonatilake, S. & Khebbal, S. (Ed). London: Wiley. To appear.

Honavar, V., & Uhr, L. (1993c). Integration of Symbol Processing and Connectionist Systems in Artificial Intelligence and Cognitive Modelling. In: *Computational Architectures for Integrating Symbolic and Neural Processes.* Sun, R., & Bookman, L. (Ed). New York: Kluwer. To appear.

Jacobs, R. A., Jordan, M. I., & Barto, A. G. Task Decomposition Through Competition in a Modular Connectionist Architecture: The What and Where Vision Tasks. Tech. Rep. 90-27. Amherst, Massachusetts: Department of Computer and Information Science, University of Massachusetts.

Koza, J. (1992). Genetic Programming. Boston, Massachusetts: MIT Press.

Laird, J. E., Rosenbloom, P. S. & Newell (1986). Chunking in SOAR: The Anatomy of a General Learning Mechanism. *Machine Learning 1*, 11-46.

Levenshtein, A. (1966). Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics - Doklady 10*, 703-710.

Michalski, R. S. (1993). Toward a Unified Theory of Learning: Multi-Strategy Task-Adaptive Learning. In: *Readings in Knowledge Acquisition and Learning.* Buchanan, B. G., & Wilkins, D. C. San Mateo, California: Morgan Kaufmann.

Miclet, L. (1986). Structural Methods in Pattern Recognition. New York: Springer-Verlag.

Minsky, M. (1975). A Framework for Representing Knowledge. In: *The Psychology of Computer Vision.* Winston, P. H. (Ed). New York: McGraw-Hill.

Parekh, R. G., & Honavar, V. (1993). Grammar Induction for Machine Learning. In preparation.

Pavlidis, T. (1977). Structural Pattern Recognition. New York: Springer.

Rosenfeld, A. (1981). Picture Languages. New York: Academic Press.

Rosenfeld, A. (1984). (Ed). Multi-Resolution Image Processing and Analysis. New York: Springer-Verlag.

Rumelhart, D. E., & McClelland, J. L. (1986) (Ed). *Parallel Distributed Processing, Vol. I-II*, Cambridge, Massachusetts: MIT Press.

Shavlik, J. W., & Dietterich, T. G. (1990). (Ed). Readings in Machine Learning. San Mateo, California: Morgan Kaufmann.

Shavlik, J. W. (1993). A Framework for Combining Symbolic and Neural Learning. In: *Symbol Processors and Connectionist Networks in Artificial Intelligence and Cognitive Modelling.* Honavar, V. & Uhr, L. (ed). New York: Academic Press. To appear.

Sowa, J. F. (1984). Conceptual Structures: Information Processing in Mind and Machine. Reading, Massachusetts: Addison-Wesley.

Tanimoto, S. L., & Klinger, A. (1980). Structured Computer Vision: Machine Perception Through Hierarchical Computation Structures. New York: Academic Press.

Uhr, L., & Vossler, C. (1963). A Pattern Recognition Program that Generates, Evaluates, and Adjusts its Own Pperators. In: *Computers and Thought.* Feigenbaum, E., & Feldman, J. (Ed). New York: McGraw-Hill.

Uhr, L. (1973). Pattern Recognition, Learning, and Thought. New York: Prentice-Hall.

Uhr, L. (1987) (Ed). Parallel Computer Vision. New York: Academic Press.

Winston, P. H. (1975) (Ed). The Psychology of Computer Vision. New York: McGraw-Hill.

Yang, J. & Honavar, V. (1993). A Hierarchical Representation Scheme for Three-Dimensional Object Recognition and Description. In: *Proceedings of FLAIRS-93*, Fort Lauderdale, Florida.