Some Learning Tasks from a Control Perspective

Andrew G. Barto
Computer and Information Science Department
University of Massachusetts

COINS Technical Report 90-122

December 1990

ACKNOWLEDGEMENTS: The author gratefully acknowledges Charles Anderson, Michael Jordan, Harry Klopf, and Richard Sutton for the many hours of discussion which have contributed to the material outlined here; Vijaykumar Gullipalli, James Houk, and Richard Yee for helpful comments on drafts of this chapter; and the Air Force Office of Scientific Research, Bolling AFB, for support through grant AFOSR-89-0526. Some of the observations made here were also made by Barto and Sutton [6].

This paper will appear as a chapter in the *1990 Lectures in Complex Systems*, Santa Fe Institute Studies in the Sciences of Complexity, Lynn Nadel and Daniel Stein (eds.), Lect. Vol. III. Redwood City, CA: Addison-Wesley.

Some Learning Tasks from a Control Perspective

Andrew G. Barto
Department of Computer and Information Science
University of Massachusetts, Amherst MA 01003

1 Introduction

Progress in understanding complex systems strongly depends on choices made in defining the problems to be studied. This process of abstraction requires both simplification sufficient to make solutions feasible and the retention of enough complexity to make solutions relevant to the real systems being studied. These aspects of the modeling methodology have been discussed extensively, but in specific cases it is easy to overlook the advice of philosophers and system theorists by forgetting that the solution to a problem abstraction may not be the ultimate goal. There is a strong tendency to commit what Alfred North Whitehead called the "Fallacy of Misplaced Concreteness" [40]: we tend to mistake abstractions for reality, especially if these abstractions are successful. This admonition is relevant to the study of learning because so many researchers are currently studying such a small fraction of the issues relavant to learning. The purpose of this chapter is to describe a collection of learning tasks in order to place the most commonly studied learning tasks into a broad context and to suggest a range of tasks that are important despite the fact that they are receiving relatively little attention.

Artificial intelligence researchers often distinguish between a system's "environment," "performance element," and "learning element" [15]. Dietterich in ref. [15] referred to these elements to make a useful statement about learning:

> The task of the learning element can be viewed as the task of bridging the gap between the level at which the information is provided by the environment and the level at which the performance element can use the information to carry out its function. (p. 328)

One can consider a spectrum of possibilities for the extent of this gap. At one extreme, the environment provides reliable, explicit, and detailed specification of desired actions. In this case, the learning element simply has to remember what it is told and employ an appropriate indexing scheme for accessing this information. A somewhat wider gap occurs if the environment can provide desired actions for only a subset of the situations that might occur in the future. Learning in this case requires storing information and forming an indexing scheme that can generalize beyond the specific instances used in training. This is usually called a supervised learning task and is being widely studied in the context of artificial neural networks. It is possible to consider much wider gaps in

which the environment provides less information, and the learning element has to perform more sophisticated processing. For example, the environment may provide unreliable and infrequent assessments of the consequences of the performance element's actions. In this case, the learning element must somehow use a problem-solving method to discover what is "worth" remembering and combine it with a storage mechanism and an appropriate indexing scheme so that this knowledge can be used to improve future performance.

Widely recognized in the field of artificial intelligence is the need to consider learning in the context of problem solving. For example, there is a trend toward developing comprehensive "learning architectures" (e.g., refs. [25, 36]). In the field of artificial neural networks, or connectionist modeling, there is also recognition of the need to study comprehensive learning architectures, but most neural network research addresses only a few subproblems of more general learning tasks. Although restricting attention to subproblems is necessary for making initial progress in understanding more complete systems, and solutions to subproblems can be of practical use in their own right, it is useful to reassess progress in light of more realistic frameworks. In this chapter, instead of the problem-solving framework of artificial intelligence, we adopt the framework of *control*. In addition to its closer ties to traditional engineering problems, a control framework more realistically emphasizes real-time interaction with reactive environments.

This chapter only addresses learning *tasks*; methods for solving these tasks are not addressed at all. Learning tasks are defined in terms of the task's objectives, the nature of the learning system's environment, and the nature of the information available to the learning system. Although the methods applicable to a particular task depend on the task's characteristics, there is not a one-to-one correspondence between tasks and methods, despite the fact that tasks and methods are often given the same names. Most methods can be applied to several classes of tasks, and most tasks can be solved by several different methods. By discussing only tasks, we hope to provide a sound basis for understanding the methods applicable to learning, while avoiding the confusion invariably produced by confounding tasks and methods.[1] Hinton [19] provides a good overview of learning methods developed for artificial neural networks, and Barto [2] discusses some of these methods in the context of control.

We describe several classes of learning tasks within a control framework designed to highlight the limitations of each class and the relationships between them. Because each class of tasks is defined by specializing this framework in specific ways, it is possible to see what additional complexities are encompassed by the more elaborate tasks, and what simplifying assumptions reduce the more elaborate tasks to the simpler ones. Some of the tasks are rather degenerate examples of control tasks, but viewing them as such serves our purpose of exposing issues they *do not address*. Our goal in using this framework is not to provide an exhaustive categorization or a "unified theory" of learning tasks. The subject of learning is so broad, with so many domain-specific features, that a useful com-

[1]The distinction between tasks and methods is similar to the distinction between Marr's [28] computational and algorithmic levels. Jordan and Rumelhart [22] provide additional insight into this distinction as it applies to issues in learning.

prehensive theory probably does not exist. Our framework does nothing to illuminate some interesting and significant classes of learning tasks, and it avoids many subleties. Further, for the most part we avoid mathematical formalism even though some of the discussion could benefit from the additional precision that exists in extensive mathematical literature pertinent to each class of tasks. The reader may be able to suggest improvements to this framework and how we embed tasks within it, but we believe it is adequate for showing that there is more to learning than is encompassed by the tasks most frequently studied.

2 Learning Control Tasks

The philosopher Daniel Dennett [16] expresses the root idea of control in common language as follows:

> A *controls* B if and only if the relation between A and B is such that A can *drive* B into whichever of B's normal range of states A *wants* B to be in. (p. 52)

For our purposes, an even broader definition is better: A controls B if and only if the relation between A and B is such that A can cause B to behave as A wants B to behave. Some of the behavior in which we are interested is not readily expressible in terms of a range of states. *Learning control* tasks involve system A learning through experience how to make a system B, and systems similar to B, do what A wants them to do.

Figure 1 shows systems A and B in a basic control loop. System A is called the controller, system B is called the controlled system, and the signals that the controller sends to the controlled system are called control signals. We will refer to these signals simply as A's actions. Other inputs to the controlled system are classified as "disturbances." These are usually regarded as unpredictable signals whose influence on the controlled system is to be counteracted by the controller. The input signals to the controller, on the other hand, typically provide information about the current state of the controlled system as well as specification of commmands to the controller, which one can regard as the "context" of the control task. Commands may indicate, for example, a desired output of the controlled system (e.g., the set point of a thermostat), a desired time course of system variables (e.g., the desired trajectory of robot manipulator positions), or more general commmands (e.g., "make the controlled system do X").

The range of learning tasks described in this chapter requires elaborating Figure 1 by adding an additional component and the interconnections as shown in Figure 2. We focus on the controller, A, as both the learning element and the performance element,[2] and we assume that its input can be divided into three categories. First, A receives

[2]We find it less useful to distinguish between a learning element and a performance element than do researchers following a symbolic approach to machine learning. In connectionist networks—and in biological systems—learning mechanisms are distributed throughout performance systems.

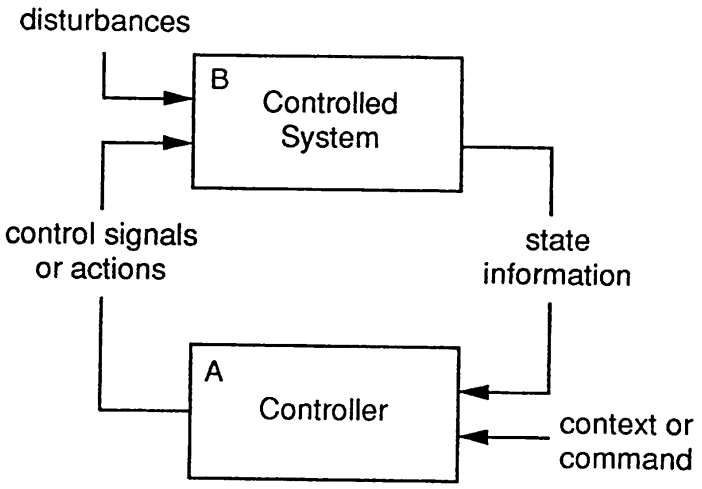

Figure 1: A Basic Control Loop. The controller, A, receives information about the state of the controlled system, B, as well as context signals or commands. The behavior of B is influenced by control signals from A and unpredictable disturbances.

information describing the state of B. This information is usually incomplete: B only provides "clues" to A about B's current state. The second category of information is provided to A from a component, labeled C, which we call a "critic" or a "teacher," depending on the type of task being considered. The critic provides information to A that guides learning. C has access to clues about the state of B and also has access to the actions of A. It implements some kind of evaluative procedure that assesses A's actions in light of B's current state according to a measure of "correctness," "goodness," "utility," etc. Sometimes the critic provides "payoffs" to A. The gap referred to above between the information provided by a learning system's environment and the information needed to accomplish some task is largely determined by the nature and quality of the information provided to A by C. Significant differences between learning tasks depend on the nature of this information.[3] The third category of information component A receives "sets the context" for the learning task. This information corresponds to the context or command information A receives in Figure 1, but in the expanded schema of Figure 2 it includes the training patterns required for some learning tasks. This information is available to all components via the line labeled "context, commands, or training patterns," but the source of the signals on this line is assumed to be outside of the system and "beyond the control" of the components A, B, and C.

In general, components A, B, and C are dynamic systems. This means that the current output of any of these components can depend on an internal state, which changes over

[3]We do not consider the case of multiple critics because the single-critic case is sufficiently complex for our purposes; many significant issues not discussed in this chapter come into play in the multi-critic case.

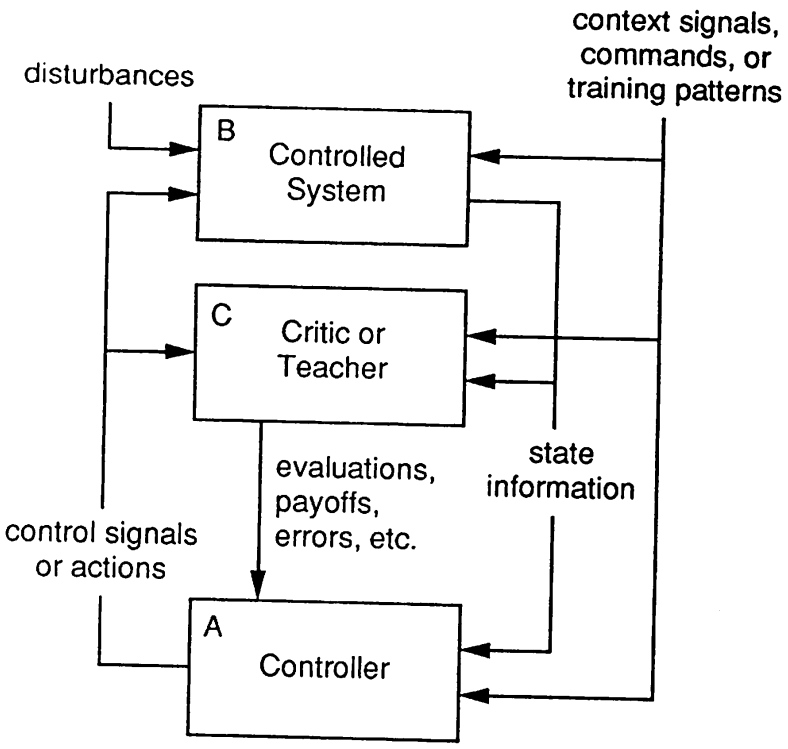

Figure 2: A General Schema for Describing a Range of Learning Tasks. The basic control loop of Figure 1 involving the controller, A, and the controlled system, B, is augmented with a critic, C, and an input to all components labeled "context, command, or training pattern" which sets the context for the learning task.

time, in addition to a current input. Consequently, the current output of a component can depend on the past history of its inputs in addition to its current input. Component A *must* be a dynamic system if it is to be capable of learning from its past experience, but for the learning tasks we consider here, some or all of the other components are assumed to be "memoryless," i.e., their current output only depends on their current input. We assume throughout that C is deterministic and memoryless, an assumption whose consequences we discuss in Section 3.

In attempting to place particular tasks into the schema shown in Figure 2, what constitutes the components A, B, and C, and what corresponds to the context signals, commands, or training patterns is not an intrinsic property of the mechanisms and phenomena under consideration. Some divisions of reality into such boxes, with certain communication channels designated as input and output channels, may have obvious advantages over others in terms of explanatory power, but these divisions are best seen as the result of the process of constructing models for specific purposes. Many different ways of matching specific cases to this schema can be useful. In particular, we point out two aspects of the schema shown in Figure 2 that are misleading if taken too literally. First, the controller, A, receives three categories of information via three distinct input channels. This is a useful abstraction that may appear differently in real situations. For example, there may be no identifiable separation of A's input channels into three categories because all of this information might be represented in a distributed code involving overlapping sets of input channels. A second misleading conclusion is that the controller, A, corresponds to something like an entire adaptive organism. This identification might make sense in some circumstances (e.g., for very simple adaptive organisms), but it is better to avoid this interpretation because parts of some or all of the components shown in Figure 2 may also reside inside the organism (e.g., B might include the organism's respiratory system, and C might represent the organism's highly evolved preference structure).

Because we frequently refer to the components of Figure 2 only by their lables A, B, and C, the following summary should be helpful in remembering their meaning:

A Controller: the learning system; combines learning and performance elements.

B Controlled System: behavior can be influenced by the controller and is evaluated by the critic; not present in all tasks.

C Critic or Teacher: implements evaluative process that assesses A's actions in light of B's current state and the current context, command, or training pattern according to a measure of "correctness," "goodness," "utility," etc.

3 Performance Measures and Gradients

Although it is notoriously difficult to give a definition of learning that is both adequate and precise, most attempts to do so involve the idea of a system improving performance

over time according to some performance measure. If one thinks of the performance measure as a function defined over the set of possible external behaviors of the learning controller (being, for the moment, purposefully vague about what the controller's external behavior is), and if one visualizes this function as a surface, then any given behavior of the controller corresponds to a point on this surface. For a controller to improve performance over time, the point corresponding to its behavior has to move to successively higher points on this performance surface; or, assuming continuous changes in behavior, the point has to move uphill across this surface.

One critical aspect of most interesting learning tasks is the problem of measuring the learning system's performance according to a performance measure that reflects the true objectives of the task. The "true" performance measure might evaluate the overall performance of the learning system over its lifetime, but it is impossible to determine this *during* the system's lifetime when learning has to occur. In practice one has to devise measures that are closely correlated with the true performance and yet can be measured more easily and more quickly. The problem of specifying accessible performance measures that can act as surrogates for basic but hard-to-measure criteria is a central issue in most realistic learning tasks.

Given this preamble, we can provide the rationale for our view of the critic as a memoryless system. Referring to Figure 2, the critic, C, provides the controller, A, with information pertinent to the *immediate consequences* of the action just taken by A on the controlled system, B, as assessed within the current context. The performance measure embodying the task's true objective, however, is not necessarily revealed in the immediate consequences of the controller's actions. The true performance measure usually evaluates behavior over longer periods of time, and although C's signals *over time* provide information relevant to A's true performance, C does not directly implement this measure. Hence, restricting discussion to memoryless critics emphasizes the problem of improving performance according to a measure that is not directly accessible. One might think of C as the "primary critic" which is part of a task's specification and provides the most basic evaluative information. Some learning methods involve "secondary critics" which provide performance information that is more accurate and more timely than the information provided by C. Research exists on how useful secondary critics can be contructed by means of learning processes (e.g., Barto, Sutton, and Anderson [7], Sutton [34, 35], Watkins [37], Werbos [39]), but within the framework presented here, these "adaptive critic" methods are considered to be implemented within the controller A.[4]

In some tasks there is no conflict between short-term performance, as revealed by the critic's instantaneous signals, and long-term performance, as determined by the task's true performance measure. In these tasks, performing each action to produce the best immediate response from the critic coincides with the task's true objective. This occurs, for example, when the training experiences to which A is exposed at each time step are

[4]Alternatively, if one focused only on the learning task faced by an adaptive critic, the task would appear as a special kind of adaptive prediction task within the class of supervised learning tasks as discussed in Section 5.

statistically independent and the true performance measure is the time average of the critic's outputs. In these cases, acting to optimize each immediate signal of the critic coincides with the objective of optimizing the critic's average signal over time. In other tasks, however, dependencies over time may make it necessary to sacrifice short-term performance in order to produce improvement with respect to the true performance measure. This occurs when A's actions not only influence the critic's immediate evaluation but also influence a dynamic process which plays a part in determining future evaluations. Section 6.3 discusses tasks with this property, called *sequential decision tasks*. All the other tasks we discuss are defined to avoid conflict between short-term and long-term performance.

Two basic types of learning tasks can be distinguished on the basis of the local characteristics of the performance surface about which the critic's instantaneous signals provide information. We call a task a *supervised learning task* if each instantaneous signal of the critic provides information pertinent to the *gradient* of the performance surface at the point corresponding to the controller's current external behavior. Recalling that the gradient of a surface at a point is a vector pointing in the direction of steepest ascent, a supervised learning task is characterized by the availability from C of *directed* information about how the controller should change its behavior in order to improve performance (at least locally). This gradient information often (but not always) takes the form of an error vector giving the difference between the controller's action and some desired, or target, action. Although each error vector (based on a single training pattern) is not the gradient of the true performance measure (a measure of the error over all training patterns) it provides useful information about the true gradient. In other cases, the critic's signals may lack the magnitude information present in error vectors and only provide directional information; for example, only the signs of errors may be provided. Although less informative than other kinds of gradient information, this is still gradient information according to our view, and such tasks are supervised learning tasks.[5] The key point is that in a supervised learning task, the learning system is provided with information about (1) whether or not a local improvement is possible (is the gradient zero?), and (2) if improvement is possible, *how* (what direction) behavior should be changed to achieve improvement.

In other learning tasks, which we call *reinforcement learning tasks*, instead of directly providing gradient information during learning, the instantaneous output of C gives the controller information about the current *value* of the performance measure. In contrast to gradient information, information about the performance measure's value does not itself indicate how the learning system should change its behavior to improve performance it is not *directed* information. A system facing a reinforcement learning task has to concern itself with estimating gradients based on information about the performance measure values provided by C over time. Although C's outputs are usually scalars in a reinforcement learning task, whereas they are vectors in supervised tasks, the distinction between

[5]The signs of the components of a gradient vector instruct a direction of local improvement but not the direction of most rapid local improvement.

scalar and vector-valued information from C is not central. For example, although we do not discuss them in this chapter, in multi-criterion reinforcement learning tasks the critic signals are vectors consisting of the values of multiple performance measures. The key point is that each output of the critic in a reinforcement learning task evaluates behavior but does not in itself indicate if improvement is possible or *direct* changes in behavior.

The distinction between supervised and reinforcement learning tasks is only one of many distinctions between learning tasks that one can make. We use this distinction as an organizing principle in our view of learning tasks because it corresponds to a significant division of existing theoretical traditions and their literatures, and it separates tasks according to issues that have been treated as almost orthogonal: *generalization* is central in most supervised learning tasks, whereas *exploration* is central in reinforcement learning tasks. Addressing these issues separately provides a good basis for understanding the more complex tasks discussed below which combine them. However, in practice it may be more difficult than we have suggested to distinguish between these classes of tasks even in relatively simple cases. For example, it may have occurred to the reader that whether a task's objective is a global or local extremum of the performance measure is a factor that complicates the situation. Just because directional information is immediately available in a task does not mean that the task's objective is best served by always following it. Doing so would result in a local extremum, whereas a global extremum may be the objective. A task such as this would match our definition of a supervised learning task in a local sense, but it would be more like a reinforcement learning task when viewed globally because exploration is a key issue for global optimization.

4 Unsupervised Learning Tasks

Unsupervised learning tasks are concerned with recoding information into forms that are more compact, better organized, or otherwise better suited for understanding or further processing. Usually, the objective is to recode while retaining as much of the original signal's information as possible. The framework used in this chapter has not been designed to do justice to unsupervised learning tasks. However, if one were to place these tasks within our control framework, they would appear as tasks in which there is no controlled system, B, and the signals that A must learn to recode are the context, command, or training patterns. We might regard the critic, C, as embodying the principle by which the recoding is to take place. Unlike all the other tasks discussed below, however, this critic would be *fixed* and completely *known* for each type of unsupervised learning task.[6] Consequently, a better view might omit the separate critic altogether and regard its function as being built into the learning system, A, from the start. In this case, there is no closed-loop interaction between any of the components, and the term "controller" for A

[6]For the other types of tasks we discuss, the critic is fixed, and possibly completely known, for each task *instance*, e.g., a specific supervised pattern recognition task. In contrast, the critic is fixed and known for each *entire class* of unsupervised learning tasks, e.g., all tasks in which principal components are to be extracted.

is highly inappropriate. For example, in one type of unsupervised learning task known as a clustering task, the objective might be to separate the input data into disjoint classes (clusters) so as to maximize the ratio of measures of the between-cluster and within-cluster scatters. All the details of this performance measure are known in advance and can be incorporated into the clustering algorithm.

In contrast, the learning tasks discussed below have the property that important aspects of the performance measure are unknown and can vary with each task instance. For example, in a supervised learning task, the desired pairings of training patterns and target actions depend on the specific task instance (Section 5). In a reinforcement learning task, even less is known about the performance measure (Section 6). Despite the rather degenerate form they take within the framework being used here, the category of unsupervised learning tasks is appropriate for concisely addressing issues of information representation that are essential in all the more elaborate learning control tasks that one can define. Although we consider them important, these issues are orthogonal to the ones we have chosen to address in this chapter. Barlow [1] provides a good philosophical overview of unsupervised learning tasks.

5 Supervised Learning Tasks

A critic capable of providing gradient information about a performance measure is usually called a "teacher." Following Jordan and Rumelhart [22], we distinguish between supervised learning tasks with "proximal" and "distal" teachers. We discuss the case of the proximal teacher first because it is the prototypical supervised learning task.

Proximal teacher—In a supervised learning task with a proximal teacher, the performance measure is defined in terms of a set of input vectors to A, called *training patterns*, and the actions that A should produce in response to them. These correct actions are sometimes called *target actions*. In practice, training patterns and target actions are represented as bit vectors or vectors of real numbers. Within the framework of Figure 2, the controlled system, B, plays no role, and the training patterns arrive from outside the system as context signals. Based on each action of A and knowledge of the target action for each training pattern, C provides an error vector to A for each training pattern, where the error vector is the vector difference between A's actual action and the target action for the current training pattern. In an alternative view of these tasks, C simply provides A with the target action for each training pattern instead of an error vector. In this case, A itself has to compute the error vector (and the term critic is not very descriptive of C). But whether C provides A with error vectors or target actions, each instantaneous output of C tells A something about how it should change the action it produces in response to each training pattern. In what follows, we regard C as providing error vectors. Figure 3 is the variant of Figure 2 appropriate for the supervised learning tasks most commonly studied. It differs from Figure 2 in that the controlled system, B, is absent.

The objective of a supervised learning task with a proximal teacher is for A to learn to produce the correct action (i.e., the target action) in response to each training pattern and

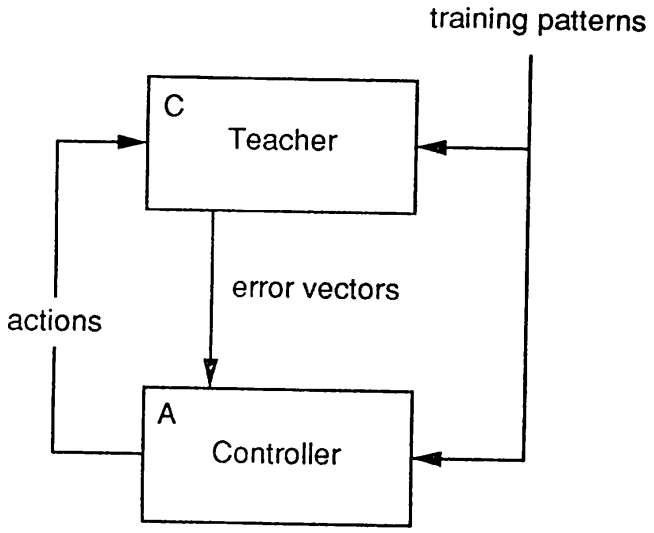

Figure 3: The Basic Components of a Supervised Learning Task with a Proximal Teacher. It differs from Figure 2 in that there is no role for the controlled system, B.

to generalize correctly to patterns not presented during training. Stating this somewhat more precisely, the objective is for A to form a mapping from the set of possible input patterns to actions that would produce error vectors of small magnitude on all possible sets of training patterns. Exactly what constitutes the magnitude of an error vector can be defined in a variety of ways which involve issues that we do not address here. Supervised learning tasks, as described above, provide the simplest framework permitting the study of generalization, and applicable methods can be useful in a wide range of practical problems. In artificial intelligence, learning in this type of task is called "learning from examples" [15].

By assuming that the training patterns arrive from outside of the system, we incorporate a key feature of supervised learning tasks as they are usually studied: the selection of training patterns is not influenced by the behavior of the controller, A. There is no closed-loop interaction between A and the source of training patterns. The teacher, C, and controller, A, interact in a closed-loop because the error vectors supplied by C depend on A's actions, but the generator of training patterns is not in this loop. The only real learning control that occurs is that A is learning how to control C in the context of each training vector: it wants to make C produce error vectors equal to zero.

A number of special cases of supervised learning tasks with a proximal teacher have been studied separately. Rote memory storage is the variant of this task in which the issue of generalization is omitted. For example, a standard computer random access memory solves the version of this task in which each training pattern is an address and each target action is the bit vector to be stored at the given address. Extending rote memory to include some form of generalization leads to more general associative memory systems,

which have been extensively studied in the form of artificial neural networks (e.g., Hinton and Anderson [20] and Kohonen [23, 24]). If the target actions are elements of a finite set, then supervised learning tasks are sometimes called adaptive pattern classification tasks where the target actions are the class labels (e.g., Duda and Hart [17]). If the target actions are real numbers, then these tasks are function approximation problems in which generalization involves interpolation and extrapolation.

Open-loop system identification (e.g., refs. [18, 27]) is also a supervised learning task with a proximal teacher. Open-loop system identification is the task of modeling an incompletely known system by observing how it responds to a set of inputs. It is open-loop because there is no simultaneous attempt to control the system being identified. The inputs to the system being identified and its corresponding outputs play the roles, respectively, of training patterns and target actions. This fits into the schema of Figure 3 by letting C contain the system being identified. Training patterns are the inputs to the system being identified and also inputs to A. The error vectors provided to A during learning are computed by comparing A's actions with the outputs of the system being identified. As the magnitudes of the error vectors are reduced, A's input/output behavior more closely matches that of the system being identified. According to this formulation, the controller, A, is not engaged in controlling the system being identified: it is attempting to control C, in the sense described above, and the task is arranged so that it has to identify the unknown system in order to do this.

This view of the open-loop system identification task does not do justice to many of its features, especially the use of delay-coordinate representations of system input and output, but it does correctly characterize the nature of the learning task according to the distinctions we are making. Similarly, open-loop adaptive prediction tasks appear as tasks in which the role of the critic is played by the system whose behavior is to be predicted. Storage buffers must be introduced for storing training patterns until the target actions become available. Barto [2] discusses these tasks in somewhat more detail; Widrow and Stearns [41] and Goodwin and Sin [18] provide views of these tasks from the perspectives, respectively, of adaptive signal processing and adaptive control. Adaptive critic methods (e.g., Sutton [35]) are methods for solving open-loop adaptive prediction tasks in which the objective is to predict some statistic about the critic's signals over the future (such as the expected sum over the future) instead of its signal at a prespecified time in the future.

Some tasks that might be called supervised learning tasks do not conform to this open-loop view. For these tasks, the generation of training patterns is not "beyond the control" of component A as we have specified. The training process may incorporate pedagogic principles which alter the order, frequency, etc. of training patterns based on A's behavior. For example, to accelerate the learning process, training may concentrate on the patterns on which large errors have been made, or training "queries" can be generated to test hypotheses [12]. These techniques, as well as others such as "fading" and "shaping" in which the training information changes over time depending on A's

behavior,[7] require that both A and C influence the source of training patterns. These techniques are not often studied from a theoretical perspective and do not fit into Figure 3, but they can be important for improving the rate and/or quality of learning. Similarly, the closed-loop system identification task requires identification to proceed at the same time that the system being identified is being controlled. Such tasks are more properly viewed as examples of adaptive sequential decision tasks discussed in Section 6.3.

Distal Teacher—The basic supervised learning task described above can be generalized to what Jordan and Rumelhart [22] call "supervised learning with a distal teacher." The gradient information provided by a distal teacher is not expressed in the same coordinate system as are A's actions. Jordan and Rumelhart [22] use the example of learning to move a robot arm to specified locations in the arm's workspace. For this problem, A's actions specify joint angles, but C provides error vectors that are differences between desired spatial positions of the end effector and the positions actually attained. Here, A is working in joint space, but C is providing Cartesian training information. Learning methods must incorporate some means for deducing from the distal teacher's training information what a proximal teacher would specify if one were provided. Jordan and Rumelhart [22] discuss methods that effectively construct an appropriate proximal teacher based on the distal teacher's training information.

Figure 4 is an elaboration of Figure 3 illustrating the components and connections required for a supervised learning task with a distal teacher. The controlled system, B, is placed in the pathway from the controller, A, to the teacher, C. Component B transforms the space of A's actions to the space in which C is able to provide error vectors. As in other supervised learning tasks, training patterns, assumed to be generated outside of the system, are available to both A and C. If B is the identity transformation, then the schema shown in Figure 4 reduces to that shown in Figure 3. For the arm movement example, B is the forward kinematic transformation of the arm, i.e, the transformation from joint angles (the actions of A) to the corresponding spatial coordinates of the end effector (the outputs of B). Training patterns are coded descriptions of the target spatial positions which play the role of commands to move the end effector to these positions. During training, a series of such commands is given to A—so that A can learn how to generate appropriate actions for each commanded position, and to C—so that C can properly assess the arm's actual movement. Although the arm, B, does not require access to these commands in the example of Jordan and Rumelhart [22], in general these signals may include context information that influences B's behavior; for example, movement may have to occur in the context of different loads.

The distal teacher, C, provides error vectors to A based on discrepancies between B's outputs and known target outputs for B for each contex signal, training pattern, or command. In the arm movement task, for each training pattern, C knows how to

[7]These terms describe techniques used to train animals [21]. Fading refers to the alteration of stimulus patterns over time as a function of the animal's behavior, whereas shaping refers to the alteration over time of the criteria by which the delivery of reinforcement is determined. Although these techniques more properly relate to reinforcement learning than to supervised learning, both might be thought of as alterations of the pattern/target pairs as a function of the animal's behavior.

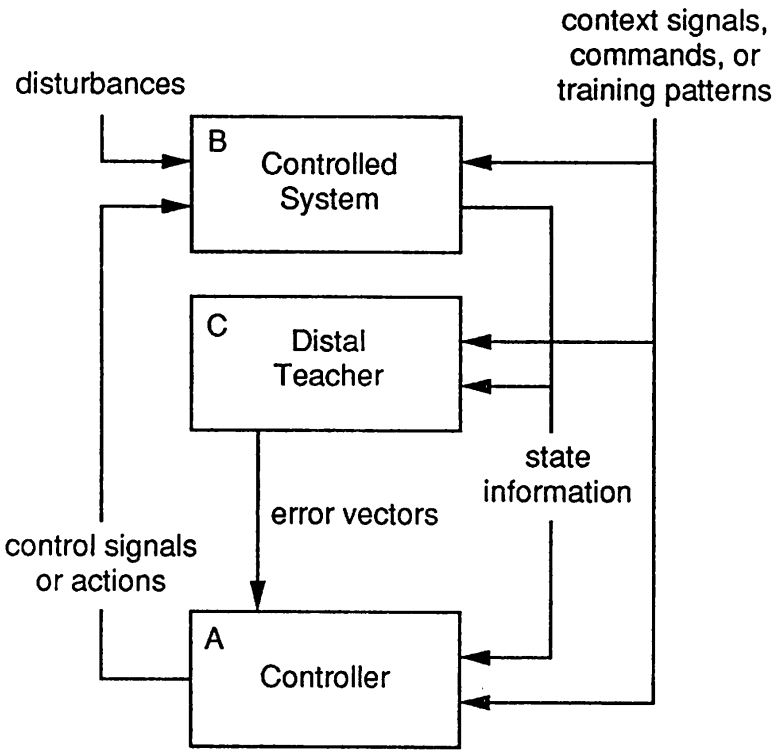

Figure 4: The Basic Components of a Supervised Learning Task with a Distal Teacher. A controlled system, B, is placed in the pathway connecting the controller, A, with the critic, C.

determine an error vector in spatial coordinates. Component A somehow has to figure out how to alter its actions in response to each training pattern to reduce the magnitude of these error vectors. This task can be nontrivial, and a variety of methods can be used. The availability of B's outputs to A, as provided by the link from B to A in Figure 4, may be useful to A depending on what learning strategy it employs. In tasks such as this, there is a closed loop of influence passing from A, through B and C, and back to A. We can say that A is trying to control C, and the task is arranged so that in the process A must also learn how to control B. One of the issues that arises in these tasks is that there may be many actions of A that can achieve a distally specified target (for example, there may be many arm configurations that place the end effector at a target spatial location). In these cases, the objective may be to achieve any one of the possible solutions, or to achieve a solution that satisfies additional constraints.

The generalization of supervised learning tasks to include distal teachers, i.e., to cases in which B is not the identity transformation, is much more widely applicable than suggested by the arm control example. In fact, when details of the internal structure of the learning component are taken into account, supervised learning tasks *always* involve a distal teacher. To understand this, recall that in Section 3 we characterized supervised learning tasks by the availability of gradient information pertaining to the performance surface defined over the *external* behavior of the controller A. Because we are discussing learning tasks and not learning algorithms, we have said nothing about how gradient information pertinent to external behavior can be translated into gradient information pertinent to the *internal* parameters of the learning component that are to be adjusted during learning. The controller, A, is being treated as a black box.

But if one focuses on an internal adjustable component of A (such as a weight of a neural network), and shifts perspective so that this internal component is now the learning component A of Figure 4, then this A faces a task with a distal teacher. This task can be viewed in terms of Figure 4 by regarding component B as the process that translates the internal component's actions into the external actions of the overall controller. How the learning task of such an internal component can be solved depends on how much knowledge is available about the transformation implemented by B. For example, in the case of a layered artificial neural network, if A is a hidden unit, component B represents the transformation implemented by the part of the network between A and the network's output units. Because the details of this transformation are known, the appropriate translation of gradient information from distal to proximal coordinate systems can be accomplished, in this case, by error back-propagation [33] (which evaluates the Jacobian transpose of B at the current operating point). If the transformation performed by B is not known in detail sufficient to permit the required translation of gradient information, B can be identified via supervised learning (as discussed above) so that the required translation can be approximated using the resulting model [22].

In the examples of supervised learning with a distal teacher given above, the controlled system B was treated as being memoryless (arm dynamics were ignored in the version of the arm movement task discussed). If B is a dynamic system, i.e., if its output depends on an internal state in addition to input, then the tasks we have classified as supervised

learning tasks with a distal teacher are the same as adaptive control tasks in which the objective is to cause B's behavior to track a desired reference trajectory, where the trajectory is specified at all time instants or only at selected time instants. If the reference trajectory is specified at all time instants, the context or command signals directly provide the target trajectory, so that the distal teacher can provide the controller with a tracking error at each time instant. If the reference trajectory is not specified at all time instants, then the task's objective is to achieve the specified behavior at the designated times while perhaps satisfying other constraints in the intervening time intervals. An example of the latter type of task occurs when the context or command signals specify a target output to be produced by B at a later time but do not specify a target trajectory for B's states or outputs over the intervening time interval. Jordan and Rumelhart [22] refer to the teacher in such tasks as being "distal in time." Although in such tasks the process intervening between A's actions and the relevant output of B extends over time, the logic is the same as for other tasks with distal teachers: somehow, knowledge about B must be used to deduce what a proximal teacher would specify at each time instant if one were present.

Supervised learning tasks with teachers that are distal in time are related to the adaptive sequential decision tasks we discuss in Section 6.3 as examples of reinforcement learning tasks. In fact, it may not be very useful to draw sharp distinctions between these tasks because they can both require learning to produce sequences of actions without the immediate availability of training information. However, within the framework we are using in this chapter, no matter how infrequent the training information, if it occurs in the form of target outputs of B, then the task is supervised because C can provide gradient information by comparing B's actual outputs with the targets. In a reinforcement learning task, on the other hand, the objective is to make B's behavior improve according to a performance measure that does not necessarily involve target outputs. But the situation is more complicated than this because learning with a teacher distal in time usually involves satisfying trajectory constraints that also are not specified in terms of target outputs. We leave these subleties for a more discriminating framework and turn to a discussion of reinforcement learning tasks.

6 Reinforcement Learning Tasks

In a supervised learning task the critic, or teacher, C, supplies information to A in the form of error vectors. Each error vector provides information about the gradient of an underlying performance measure at the current operating point. The gradient vector points in the direction in which the controller's behavior should be changed to to yield the best local improvement. In a reinforcement learning task, in contrast, C provides A with information pertinent to just the value of the performance measure for the current behavior. Accordingly, for reinforcement learning tasks we call C's outputs "evaluations." An evaluation does not tell A how it should change its behavior, or even whether improvement is possible or not. We distinguish between reinforcement learning

tasks according to what information is available to A in addition to C's evaluations. In *nonassociative* tasks, A receives only C's evaluation, whereas in *associative* tasks, A has access to information that can influence its actions via associative relationships.

6.1 Nonassociative Reinforcement Learning Tasks

In a nonassociative reinforcement learning task the only input A receives are the evaluations provided by C. Each of A's actions is separately evaluated by C, and the objective is for A to find the best action (or one of possibly many best actions) as evaluated by C. This basic picture is complicated by the possibility that C's evaluations may not depend in a simple way on A's actions. Evaluations may depend on a complex, and possibly stochastic, dynamic system intervening between A's actions and C, as well on other factors that A's actions cannot influence. To accomodate these possibilities within the general schema of Figure 2, we use all of the components and delineate special cases by assigning specific properties to them.

Figure 5 shows the basic arrangement of these components for nonassociative reinforcement learning tasks. Because neither the context signals nor the outputs of B are available to it, A cannot select actions conditionally on this information even though the evaluations it receives may depend on this information as well as on its actions. We use the term nonassociative because A cannot form an associative mapping from this information to actions. Because C evaluates the consequences of A's actions on B, C may not need access to the actions themselves. However, in some cases C's evaluation depends on the actions themselves as well as on their consequences (for example, the actions may have different "costs"), and thus we provide the direct connection from A to C in Figure 5 (which is present for the same reasons in all remaining figures depicting reinforcement learning tasks).

If the output of B is a deterministic function of A's actions (that is, if there are no disturbances and B is memoryless and insensitive to any changes in context signals) and the performance measure concerns only the instantaneous behavior of A, then the task is a function optimization task. If C implements a real-valued function of B's output, as is usually assumed, then B and C together assign a real number to each action of A. The objective is for A to determine which of its actions maximizes this number, or, since this is generally impossible, to find an action that is a local maximum. Within the framework being used in this chapter, this is a relatively simple *type* of task, although specific instances can be extremely difficult to solve due to the nature of the functions implemented by B and C. Function optimization has been studied very extensively. Central issues concern the search efficiency, local versus global optimization, and the justification and use of specific classes of models of the function to be optimized.

If B's output is not a deterministic or memoryless function of A's actions, then the evaluation A receives from C in response to an action may not be the same each time that action is performed. This can also happen if B and/or C are sensitive to multiple context signals. In these cases, the performance measure has to take this variability into account. It often makes sense to model the overall evaluation process in terms of

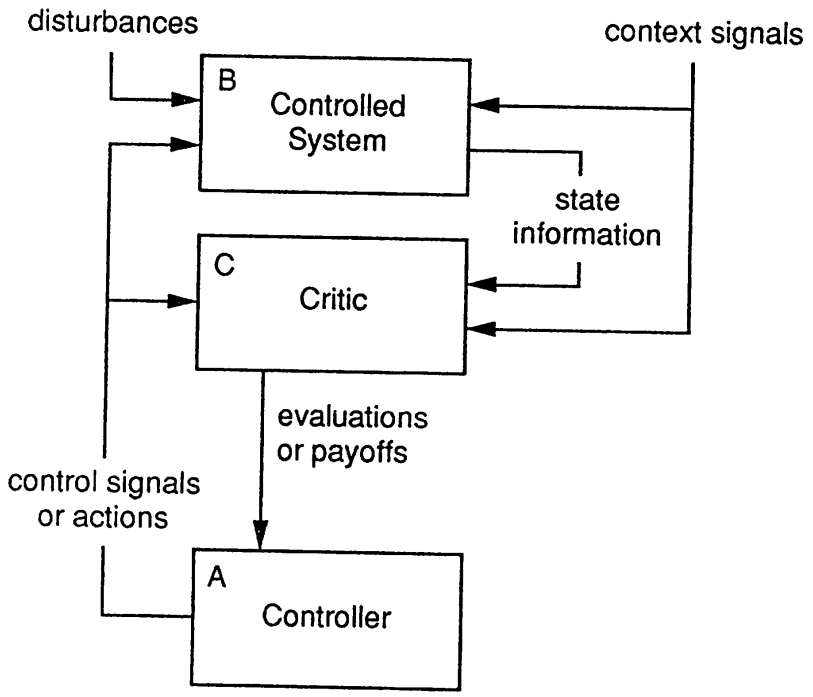

Figure 5: The Basic Components of a Nonassociative Reinforcement Learning Task. The only input A receives is the evaluative input from C.

probability distribitions, and to define the performance measure as the average over time of the evaluations produced by C. Two cases are distinguished depending on whether the probability distributions are stationary or nonstationary over time. A stationary probabilistic model is appropriate when B is memoryless and B and C are insensitive to multiple contexts. In this case, the action that yields the optimal average evaluation stays the same over time. A nonstationary model is appropriate when B is not memoryless and/or there is sensitivity to multiple context signals. Here, an action that is best at one time may not be best at another. Because A is blind both to B's state and to any context signal, to A it appears that C is changing its evaluation criteria over time.

Stationary tasks of this kind have been extensively studied as "bandit problems" (e.g., ref. [13]) or addressed by means of the theory of learning automata (e.g., ref. [29]). For example, learning automata theorists have studied the following problem. The controller, A, has n actions a_1, a_2, ..., a_n. Immediately after A generates an action, C sends to A a signal indicating either 'success' or 'failure.' This signal is determined from A's action by some complex nondeterministic process implemented by components B and C of Figure 5. Whatever this process may be, it is modeled simply as a collection of success probabilities d_1, d_2, ..., d_n, where d_i is the probability of returning 'success' given that A has produced action a_i. Each d_i can be any number between 0 and 1 (the d_i's do not have to sum to one), and one assumes that A has no initial knowledge of these values. The objective is for A to eventually maximize the probability of receiving 'success,' which is accomplished when A always performs the action a_j such that $d_j = \max\{d_i | i = 1, n\}$. In a variant of this task, the success probabilities are known, but it is not known which corresponds to which action.

In the nonstationary version of this task, the success probabilities, d_i, change over time. A reasonable objective under these circumstances is for A to maintain a high level of performance over time by continuing to change its action in an attempt to track the maximum of the nonstationary evaluation process. It can do this by attempting to construct a predictive model of the evaluation process, but this is difficult to do only on the basis of the information provided by C, or it may be impossible due to unpredictable shifts of context. More realistically, if A can adjust its behavior quickly enough to track the nonstationary evaluation process, then it can maintain good performance over time without constructing a predictive model. The point, however, is that even though evaluations of A's actions may depend on the context and on the state of B, it is not possible for A to select actions conditionally on this information because A is completely blind to this information.

To prevent too narrow an interpretation of nonassociative reinforcement learning tasks, note that the following qualifies as an example. Suppose each of A's actions is an entire control rule which can be applied to a complex stochastic dynamic system. Let the transformation B assign to each such action (now an entire control rule) some description of the dynamic system's behavior under the control of the corresponding control rule. For example, B might produce a description of the state the dynamic system reaches after some time interval. Based on this state description, C provides to A an

evaluation of the choice of control rule.[8] The solution sought in this task is a control rule optimal in a sense determined by the critic. This formulation of learning control does not have much to recommend it due to the size and complexity of the sets and systems involved, but it is an example of a nonassociative reinforcement learning task.

Deeper aspects of reinforcement learning tasks exist when the objective is not just to discover *eventually* which action is optimal, but also to perform the optimal action as frequently as possible during the discovery process, or, more generally, to maintain performance throughout the discovery process at as high a level as possible. The issue arises of how to balance the requirement for maintaining a high level of performance over time with the requirement for estimating the relative worth of the actions. Two factors must influence each action selection: (1) the desire to use what is already known about the relative merits of the actions, and (2) the desire to acquire more knowledge about actions' consequences to make better selections in the future. These two factors ordinarily conflict: the best decision according to one is not best according to the other. This is called the conflict between control and identification. It is present in its simplest form in the stochastic success/failure task described above, which is one of the simplest adaptive optimal control tasks [31].

Despite the limitations imposed by restricting the information to which component A has access, nonassociative reinforcement learning tasks clearly illustrate differences between reinforcement learning tasks and supervised learning tasks. Obviously, when C directly provides A with gradient information, as it does in a supervised learning task, the learning algorithm implemented by A does not have to estimate the gradient of the performance measure from a set of evaluations. It is effectively told by C how to change its behavior. In a reinforcement learning task, however, the learning component has to *do something* to estimate the gradient. What it does depends on the specifics of the learning algorithm, but in all cases it must perform some form of exploratory behavior and must manipulate the resulting set of evaluations to determine how to change the rule for generating behavior. In a supervised learning task, as formulated here, all the responsibility for exploratory behavior is relegated to whatever process generates training patterns, and this process is usually fixed and simple (e.g., it repeatedly cycles through a finite set of training patterns). Consequently, the conflict between control and identification—the issue that emerges in its most stark form in nonassociative reinforcement learning tasks—is not present in supervised learning tasks unless the generation of training examples can be influenced by A's actions.

6.2 Associative Reinforcement Learning Tasks

Associative reinforcement learning tasks have all of the properties of nonassociative reinforcement learning tasks except that A has access to information other than C's evaluations. Consequently, A can take advantage of information that can help it perform

[8] Biological evolution is sometimes viewed this way: each of A's actions is the genetic material of an organism, and C provides a measure of the reproductive success of that organism over its lifetime.

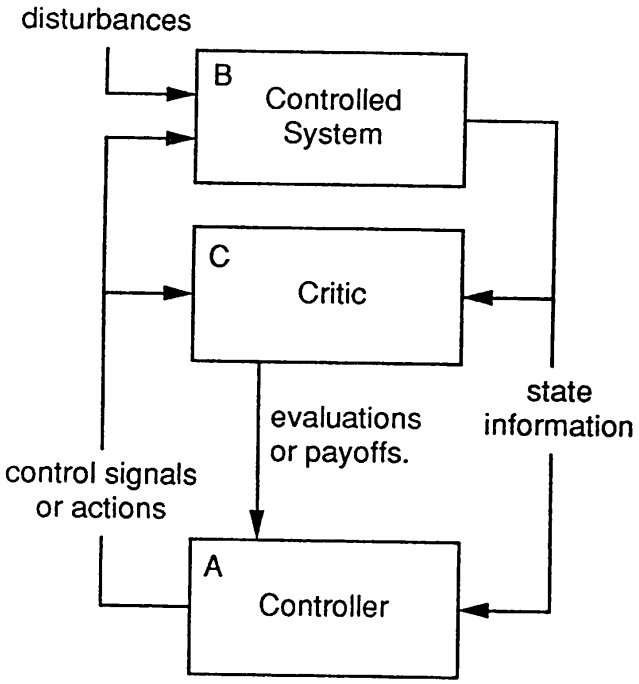

Figure 6: Basic Components of an Associative Reinforcement Learning Task. The critic provides evaluative information to A, which also has access to information about the state of B and the context signal or command.

better than can its blind counterpart facing a nonassociative version of the same task. In Figure 6, the controller A has access to information about the state of B and, if present, context or command information.

One of the simplest associative reinforcement learning tasks is a generalization of the 'success/failure' nonassociative task described in Section 6.1. Suppose that at any time, there can be one of several context signals and that the success probabilities depend on the context signal as well as the action of A. Specifically, suppose d_{ix} is the probability of C sending 'success' to A given that A produced action a_i while the context signal was x. To achieve the best rate of success, whenever A receives context signal x, it should select the action a_j such that $d_{jx} = \max\{d_{ix} | i = 1, n\}$. Clearly, if A selects actions independently of the context signals—as it would be forced to do if facing a nonassociative task—then in general it could, at best, achieve a lower success rate than if its actions could depend on the context signals. A's sensitivity to context signals eliminates nonstationarity by allowing the nonstationary task to be solved as multiple stationary tasks. Tasks of this kind are discussed by Barto, Sutton, and Brouwer [8], Barto and Anandan [5], Barto [3, 4], and Williams [42, 43].

In other associative reinforcement learning tasks, the controller, A, can take advan-

tage of information about B's current state if this state influences C's evaluation of the current action. For example, B's current state may act as context for the evaluation process in the same way that the context signals do in the example just described, except that in this case, A's actions can influence how these context signals change over time. Consequently, A's actions not only can influence the immediate evaluation signal produced by C, but they can also influence future evaluations by influencing B's future behavior. However, by an associative reinforcement learning task we mean a task in which the underlying performance measure is such that the the objective is to maximize only the *immediate* evaluation at each step. More complex tasks, called adaptive sequential decision tasks, require learning how to perfom actions so as to maximize a measure of long-term performance by manipulating B's long-term behavior. In these tasks, it can make sense to forego a favorable immediate evaluation in order to achieve a better evaluation in the future. We discuss adaptive sequential decision tasks in the next section.

Associative reinforcement learning tasks combine the important aspects of nonassociative reinforcement learning tasks with aspects of supervised learning tasks, while avoiding the additional complexities present in sequential tasks. For example, the context signals in the associative version of the 'success/failure' task just described correspond to pattern vectors in a supervised learning task. The objective is for A to respond to each pattern with the action that is optimal for that pattern, but it must learn how to do this on the basis of success/failure feedback instead of error vectors. All the issues concerning generalization which are important in supervised learning are important here, but there is a conflict between control and identification as well.

6.3 Adaptive Sequential Decision Tasks

Adaptive sequential decision tasks are characterized by performance measures that evaluate the controller's behavior over extended periods of time, but unlike some of the tasks discussed above—which are special cases—the full framework for sequential tasks takes into account the possibility of the controller influencing the long-term behavior of the controlled system, B. Whereas in associative reinforcement learning tasks, the controller, A, has to discover what actions produce the most favorable immediate evaluation from C, in sequential tasks A has to learn how to select the actions that maximize some cumulative measure of the evaluations it receives over time. This is more difficult than merely trying to achieve the best immediate evaluation. Some actions may be useful in producing a high immediate evaluation, but these same actions may cause B to enter states from which *later* high evaluations are unlikely or impossible. Hence, performing these actions would result in worse performance over the long-term than might be possible otherwise. Conversely, some actions may produce low evaluations in the short-term but are necessary to set the stage for better evaluations in the future. The controller's decision-making method must somehow account for both the short- and the long-term consequences of actions. By a task's horizon, we mean the duration of the time interval into the future over which the consequences of current actions are significant.

Many tasks of practical significance can be modeled as sequential decision tasks. Finding the least-cost route from one place to another is perhaps the most generic example. Choice points along a route correspond to the states of B, A's actions determine the path taken and the next place reached, and the evaluation from C in response to an action is related to the cost of traveling the path. In this example, it is clear that an action influences the immediate evaluation (the cost of the path immediately taken) as well as the evaluations possible over the future (determined by the place reached). More complex tasks involving resource allocation, investment, gambling, and foraging for food are also examples of sequential decision tasks. Many of the planning and problem-solving tasks studied by artificial intelligence researchers are sequential decision tasks. Unlike most of the tasks described above, solving sequential decision tasks can be difficult even if one knows all the details about the sytems and performance measures involved. Bertsekas [14] provides a good account of sequential decision tasks and their solution using computational methods known as dynamic programming, which are applicable when there is an accurate model of the systems and performance measures involved.

By an *adaptive* sequential decision task we mean a sequential decision task in which one does not have the accurate models required for applying these solution methods. These tasks are obviously very difficult to solve: solving them requires learning, and they can simultaneously involve the difficulties present in all of the other learning tasks we have discussed. The associative reinforcement learning tasks discussed in Section 6.2 are special cases of adaptive sequential decision tasks whose horizons are reduced so that only the immediate consequences of actions are significant. Some approaches to solving adaptive sequential decision tasks are discussed by Barto, Sutton, and Watkins [9, 10] and Barto and Singh [11].

One type of sequential decision task receiving attention is known as a Markov decision task (e.g., ref. [32]). In this kind of task, the controller, A, interacts with a controlled system, B, which is a discrete-time, finite-state, stochastic dynamic system. At each time step t, A observes the system's current state, x_t, and selects an action, a_t. After the action is performed, A receives a certain amount of payoff, r_t, from the critic, C, that depends on a_t and x_t, and B makes a transition from state $x_t = x$ to state $x_{t+1} = y$ with probability $P_{xy}(a_t)$. Upon observing state x_{t+1}, the controller A chooses another action, a_{t+1}, and continues in this manner for a sequence of time steps. The objective of the task is for A to form a rule to use in selecting actions, called a decision policy, that maximizes a measure of the total amount of payoff accumulated over time.

One commonly studied measure of cumulative payoff is the expected infinite horizon discounted return. This is defined by using a discount factor, γ, $0 \leq \gamma < 1$, to weight future payoffs less than immediate payoffs. Specifically, the expected infinite horizon discounted return for a given policy and state, x, is

$$E\left[\sum_{t=0}^{\infty} \gamma^t r_t | x_0 = x\right], \tag{1}$$

where x_0 is the initial system state, and E is the expectation assuming that A uses the given policy. The objective of the decision task is to form a policy (there may be many) that maximizes the expected infinite horizon discounted return defined by Equation 1

for each system state x. This sequential decision task reduces to the 'success/failure' associative reinforcement learning task described in Section 6.2 if there are only two possible payoff values and the discount factor, γ, equals zero. In this case, one can identify the larger payoff value with the 'success' signal of the associative reinforcement learning task, and the smaller value with the 'failure' signal of that task. When γ equals zero, Equation 1 becomes simply $E[r_0|x_0 = x]$, which is maximzed when the probability of immediate success is maximized for each state x. If in addition, B has only one state, this task further reduces to the 'success/failure' nonassociative reinforcement learning task described in Section 6.1.

The most general adaptive sequential decision tasks require all the components and connections shown in Figure 2, with the exception of the pathway for context signals, commands, or training patterns. If such signals were present, they could indicate which of several different adaptive sequential decision tasks was present at any particular time. For example, in the context of "thirst," a foraging animal's task would be different than in the context of "hunger." The critic's evaluation criteria would depend on the context, and the animal's foraging strategy, implemented by A, may come to depend on the context also. We might call such tasks "associative adaptive sequential decision tasks" in analogy with the associative reinforcement learning tasks discussed above. Solving them would involve learning associative relationships between context signals and control rules effective in different contexts, and generalization among different sequential decision tasks could be important. This observation brings us in a full-circle back to the issues addressed by supervised learning tasks, except that here the gap between what must be learned and the kind of training information available is wide indeed.

7 Discussion

In this chapter we described a collection of learning tasks in order to place the tasks most commonly studied into a broad context and to describe other types of learning tasks that are receiving relatively little attention. We restricted discussion to learning *tasks*—defined in terms of the task's objectives, the nature of the learning system's environment, and the nature of the information available to the learning system—and did not discuss methods for solving these tasks. By placing all the tasks within a control framework we attempted to illuminate the limitations of each type of task and the relationships between them, even though some of the tasks are rather degenerate kinds of control tasks. We reiterate that our aim has not been to provide an exhaustive categorization or a unified theory of learning tasks.

We distinguished two broad classes of tasks on the basis of the type of training information available during learning. In supervised learning tasks information about the gradient of the performance measure is directly available to guide learning. By gradient information we mean information that directly tells the system if local improvement in behavior is possible and, if so, specifies how the behavior should be changed, i.e, in what direction a change should be made. The central issue in supervised learning tasks is

generalization: how can a complete mapping be inferred from a sample of examples? In contrast, the training signals in reinforcement learning tasks do not directly contain directional information. These signals evaluate behavior but do not directly indicate if local improvement is possible or how the behavior should be changed for improvement. Directional information must be obtained from a collection of signals evaluating different behaviors, but the learning system itself has to perform this integrative process. It has to probe the environment—perform some form of *exploration*—to obtain information about how to change its behavior. In so doing, a system in a reinforcement learning task can encounter a conflict between how it has to change behavior in order to obtain directional information, and how the resulting directional information tells it to change its behavior for improvement. This is known as the conflict between control and identification and is absent in supervised learning tasks. Although many tasks involve aspects of both supervised learning and reinforcement learning tasks—so that the distinction between these classes of tasks may not be as sharply drawn in practice as we have suggested—this distinction has significant consequences for the design of learning methods.

Following Jordan and Rumelhart [22], we discussed supervised learning involving proximal and distal teachers. Whereas a proximal teacher provides gradient information in the coordinate frame of the learning system's actions, a distal teacher provides this information in a different coordinate frame. Solving an example of the latter task requires transforming distal training information into the information a proximal teacher would provide if one were present. Tasks with distal teachers involve control in more substantive ways than do other supervised learning tasks. To learn to achieve distal targets implies that the learning system must learn to control aspects of the system that transforms its actions into the distal coordinate frame (component B in Figure 4). If this is a dynamic system, these tasks correspond to adaptive control tasks in which the objective is to control a system so that its output tracks a desired reference trajectory.

We also emphasized a distinction between nonassociative and associative learning tasks. In a nonassociative task, the learning system tries to find a single optimum action, whereas in an associative task, it tries to construct an associative mapping from inputs providing state and/or context information to optimal actions. We discussed this distinction only in terms of reinforcement learning tasks. Nonassociative reinforcement learning tasks are the simplest tasks involving the issues of exploration and the conflict between control and identification. Supervised learning tasks, on the other hand, must be associative in order to be interesting (except, perhaps, in cases involving distal training information). A key feature of supervised learning is the possibility for forming associative mappings that generalize appropriately to novel inputs, and in nonassociative cases there are no novel inputs.

For any kind of learning task, access to state and/or context information is important because it may allow nonstationary nonassociative tasks to be transformed into stationary associative tasks. In the nonstationary case, the critic's evaluation criteria may appear to vary over time because information relevant to performance is unavailable to the learning system; in the stationary case, context signals allow the learning system to alter its behavior according to which of a collection of stationary tasks is being faced at any

time. Given the ability to make its behavior conditional on relevant state and/or context information, a learning system has the potential for dramatically improved performance over what it could achieve if it were blind to this information. It is obviously important, therefore, for a learning system to have access to information that it can use to reduce or eliminate nonstationarity by means of associative processes.

In associative learning tasks, state and context signals play similar roles in supplying information that can be used to reduce nonstationarity. In the framework used in this chapter, the difference between state and context signals is that the former have the potential for being influenced by the learning system, whereas we assumed that the latter were beyond the learning system's control. We let the context signals play the role of the training patterns in supervised learning tasks because in most studies of supervised learning applied to pattern classification or function approximation, the sequence of training patterns is not influenced by the learning system's behavior. But it should be clear that signals regarded as context in one task formulation may become state signals in another formulation that extends the learning system's influence.

Finally, we discussed sequential decision tasks in which it can make sense to forgo short-term performance in order to achieve better performance over the long-term. Solving a task having this property requires extensive planning that can be difficult even if all the details of the task are known in advance. When these details are not all known in advance, performance over the long-term can be improved by learning, and we used the term adaptive sequential decision task for these cases. The other reinforcement learning tasks discussed in this chapter are special cases of adaptive sequential decision tasks. Reducing the horizon so that only the immediate consequences of actions influence the performance measure yields associative reinforcement learning tasks. If in addition the learning system is not permitted access to information other than the critic's evaluations, the task becomes a nonassociative reinforcement learning task. Consequently, all the issues we have discussed in this chapter must be considered in solving adaptive sequential decision tasks.

Although the framework adopted in this chapter for comparing and contrasting learning tasks is one of many that could have been created, the exercise of embedding a range of tasks within it demonstrates that there are many factors relevant to learning beyond those addressed in the most commonly studied learning tasks. Conducting this exercise while avoiding discussion of learning methods eliminates the confusion arising when aspects of tasks and methods are confounded. When studying learning methods against this background, a clearer view of their capabilities and limitations is possible. For example, one can consider separately two different categories of limitations on the capabilities of a learning method.[9] A method may be limited in its ability to solve tasks *of a given type* that are more complex than a certain level, and a method may be limited because it can only solve a *certain type of task*. For example, an artificial neural network consisting of a single layer of linear threshold units using the perceptron learning rule can learn to implement only linear discriminate functions. A very different limitation of such

[9]This point was made by Barto and Sutton in ref. [6].

a network is that even if it could learn to implement arbitrarily complex discriminant functions (for example, by using the error backpropagation method [26, 30, 33, 38]), it would still require an environment capable of providing a specific type of detailed training information.

Although solution methods for difficult nonlinear pattern classification, function approximation, and function optimization tasks will have important roles in sophisticated learning systems, it seems to us that sophisticated learning behavior can result from a system designed to solve many interrelated learning tasks of different types, where each task is a relatively simple example of its type given the available prior knowledge. One key to designing useful learning systems may be to design systems applicable to realistic models of the tasks actually faced by animals than to abstract tasks isolating only a few features of realistic learning tasks.

References

[1] H. B. Barlow. Unsupervised learning. *Neural Computation*, 1:295–311, 1989.

[2] A. G. Barto. Connectionist learning for control: An overview. In T. Miller, R. S. Sutton, and P. J. Werbos, editors, *Neural Networks for Control*. MIT Press, Cambridge, MA. To appear.

[3] A. G. Barto. Learning by statistical cooperation of self-interested neuron-like computing elements. *Human Neurobiology*, 4:229–256, 1985.

[4] A. G. Barto. From chemotaxis to cooperativity: Abstract exercises in neuronal learning strategies. In R. Durbin, R. Maill, and G. Mitchison, editors, *The Computing Neuron*, pages 73–98. Addison-Wesley, Reading, MA, 1989.

[5] A. G. Barto and P. Anandan. Pattern recognizing stochastic learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, 15:360–375, 1985.

[6] A. G. Barto and R. S. Sutton. Goal seeking components for adaptive intelligence: An initial assessment. Technical Report AFWAL-TR-81-1070, Air Force Wright Aeronautical Laboratories/Avionics Laboratory, Wright-Patterson AFB, OH, 1981.

[7] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:835–846, 1983. Reprinted in J. A. Anderson and E. Rosenfeld, *Neurocomputing: Foundations of Research*, MIT Press, Cambridge, MA, 1988.

[8] A. G. Barto, R. S. Sutton, and P. S. Brouwer. Associative search network: A reinforcement learning associative memory. *IEEE Transactions on Systems, Man, and Cybernetics*, 40:201–211, 1981.

[9] A. G. Barto, R. S. Sutton, and C. Watkins. Learning and sequential decision making. In M. Gabriel and J. W. Moore, editors, *Learning and Computational Neuroscience.* MIT Press, Cambridge, MA. To appear.

[10] A. G. Barto, R. S. Sutton, and C. Watkins. Sequential decision problems and neural networks. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 686–693, San Mateo, CA, 1990. Morgan Kaufmann.

[11] A.G. Barto and S.P. Singh. On the computational economics of reinforcement learning. In *Proceedings of the 1990 Connectionist Models Summer School.*

[12] E. B. Baum. Neural net algorithms that learn in polynomial time from examples and queries. *IEEE Transactions on Neural Networks.* To appear.

[13] D. A. Berry and B. Fristedt. *Bandit Problems.* Chapman and Hall, London, 1985.

[14] D. P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models.* Prentice-Hall, Englewood Cliffs, NJ, 1987.

[15] P. R. Cohen and E. A. Feigenbaum, editors. *The Handbook of Artificial Intelligence*, volume 3. HeurisTech Press, Stanford, California, 1982.

[16] D. C. Dennett. *Elbow Room. The Varieties of Free Will Worth Wanting.* MIT Press, Cambridge, MA, 1985.

[17] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.

[18] G. C. Goodwin and K. S. Sin. *Adaptive Filtering Prediction and Control.* Prentice-Hall, Englewood Cliffs, N.J., 1984.

[19] G. E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40:185–234, 1989.

[20] G. E. Hinton and J. A. Anderson, editors. *Parallel Models of Associative Memory.* Erlbaum, Hillsdale, NJ, 1981.

[21] W. K. Honig and J. E. R. Staddon. *Handbook of Operant Behavior.* Prentice Hall, Englewood Cliffs, NJ, 1977.

[22] M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. Submitted for publication.

[23] T. Kohonen. *Content-Addressable Memories.* Springer-Verlag, Berlin, 1980.

[24] T. Kohonen. *Self-Organization and Associative Memory.* Springer-Verlag, Berlin, 1984.

[25] J. E. Laird, P. S. Rosenbloom, and A. Newell. Chunking in SOAR: The anatomy of a general learning mechanism. *Machine Learning*, 1:11–46.

[26] Y. le Cun. Une procedure d'apprentissage pour reseau a sequil assymetrique [A learning procedure for asymmetric threshold network]. *Proceedings of Cognitiva*, 85:599–604, 1985.

[27] L. Ljung and T. Söderstrom. *Theory and Practice of Recursive Identification.* MIT Press, Cambridge, MA, 1983.

[28] D. Marr. *Vision.* W. H. Freeman, San Francisco, 1982.

[29] K. Narendra and M. A. L. Thathachar. *Learning Automata: An Introduction.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[30] D. B. Parker. Learning logic. Technical Report TR-47, Massachusetts Institute of Technology, 1985.

[31] J. W. Polderman. *Adaptive Control and Identification: Conflict or Conflux?* Centrum voor Wiskinde en Informatica, P.O. Box 4079, 1009 AB Amsterdam, The Netherlands, 1987.

[32] S. Ross. *Introduction to Stochastic Dynamic Programming.* Academic Press, New York, 1983.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol.1: Foundations.* Bradford Books/MIT Press, Cambridge, MA, 1986.

[34] R. S. Sutton. *Temporal Credit Assignment in Reinforcement Learning.* PhD thesis, University of Massachusetts, Amherst, MA, 1984.

[35] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

[36] R. S. Sutton. Integrating architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 216–224, San Mateo, CA, 1990. Morgan Kaufmann.

[37] C. J. C. H. Watkins. *Learning from Delayed Rewards.* PhD thesis, Cambridge University, Cambridge, England, 1989.

[38] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* PhD thesis, Harvard University, 1974.

[39] P. J. Werbos. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man, and Cybernetics*, 1987.

[40] A. N. Whitehead. *Science and the Modern World (Lowell Lectures, 1925).* The Macmillan Company, New York, 1925.

[41] B. Widrow and S. D. Stearns. *Adaptive Signal Processing.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1985.

[42] R. J. Williams. Reinforcement learning in connectionist networks: A mathematical analysis. Technical Report ICS 8605, Institute for Cognitive Science, University of California at San Diego, La Jolla, CA, 1986.

[43] R. J. Williams. Reinforcement-learning connectionist systems. Technical Report NU-CCS-87-3, College of Computer Science, Northeastern University, 360 Huntington Avenue, Boston, MA, 1987.