Institution of Electrical Engineers, Conference on Pattern Recognition, National Physical Laboratory, July 1968

The convergence of adaptive pattern classifiers in control systems

B.R.Gaines, Department of Electrical Engineering Science, University of Essex
D.J.Quarmby, Department of Physics, University College, London

Synopsis Whilst the convergence of adaptive-threshold-logic elements may be dealt with quite simply when they are used for pattern recognition, the behaviour of such devices is far more complex and less amenable to analysis when they form the variable element of an adaptive control system. This is because they have the dual control problem of identifying their environment by classification whilst controlling it, and performance feedback is both relative and global. This paper considers five problem areas in the utilization of pattern classifying control systems: the derivation of performance feedback, appropriate coding of input and response patterns, the state-space of adaption and training, linguistic communication with the controller, and hardware realization of the controller.

Introduction

Although the main virtue of adaptive control systems is usually taken to be their relative insensitivity to changes in the nature, or parameters, of their environment, there are other characteristics of equal, if not greater, importance. In particular it may be possible to devise controllers which are *simple to fabricate* because of their homogeneity of structure, of *low cost* because of their universality and concomitant mass-production, and which are *self-synthesizing* through their adaptive capability. Classical techniques of controller synthesis become very difficult and uneconomic when faced with complex, nonlinear, multi-variable environments, and the optimum approach to the synthesis and fabrication of specialized controllers may well become one of making best use of the self-synthesizing capabilities of a few general-purpose adaptive control systems. In such a case the environment and control policy may never change during the working life of the controller, and adaption becomes a means of synthesis rather than a dynamic, sensitivity-reducing, feature of the controller.

One of the most attractive forms of 'learning module' for batch-fabrication at low cost is the *adaptive-threshold-logic element* studied by many workers as a pattern-classifier. This device is simple enough to allow a detailed analysis of convergence under certain conditions, but it is also powerful enough to be utilized as-the basic building block of complex, hierarchical control systems. This paper considers the problems encountered in attempts to utilize simple adaptive-threshold-logic pattern-classifiers as the variable elements of adaptive control systems. Although the behaviour of these devices is simple to analyze in conventional 'stimulus presentation with reward/punishment' situations, it becomes far more complex and less amenable to analysis when they form part of an adaptive controller. This is because the sequence of incoming patterns is a function of previous outputs of the threshold logic unit, since these determine the control actions which influence the environment generating the patterns, and also because the immediate, decision by decision, performance-feedback of pattern recognition situations is lacking, and the performance feedback is generally averaged over many decisions and delayed.

Rather than treat in depth one particular aspect of pattern-classifying adaptive control, this paper presents an over-view of a number of problems, each the subject of current research, and places them in the context of a simple, single-level configuration in which a basic adaptive-threshold-logic element is used to generate and implement a control policy.

The Controller and Associated Problems

The complete control system considered is shown in Figure 1. The 'environment' may be virtually any system with inputs and outputs, but for most of the experiments described it is a continuous system, consisting of two or three integrators in cascade with interns disturbances, feedback loops and output limiting. The controller is constructed from an adaptive-threshold-logic element, whose input and response patterns are binary vectors, and which requires performance feedback in the form of a binary signal. Hence analogue/binary coders and decoders are placed in the input and output channels, and some form of performance evaluation must be provided.

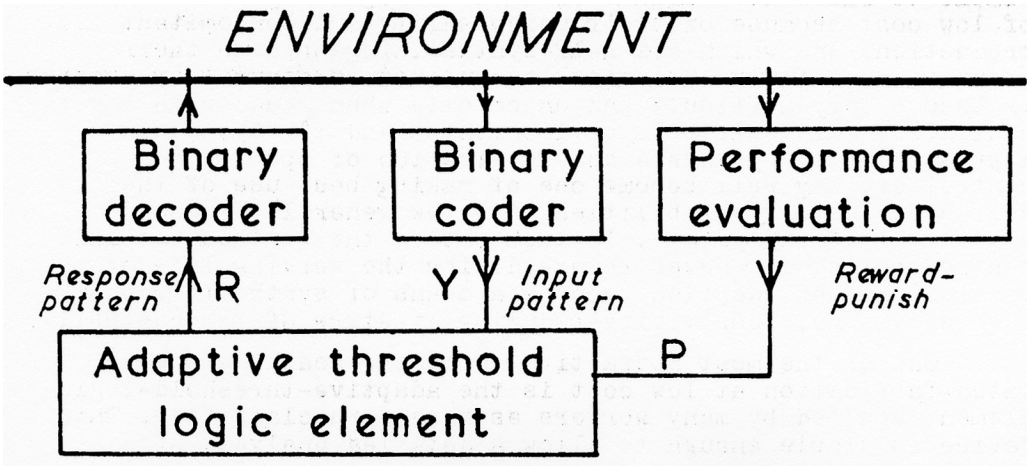

Figure 1: Pattern Classifying Adaptive Controller

The major problems concerned with the realization, utilization and performance, of pattern classifying adaptive Controllers, discussed in this paper, are:

(i) Performance feedback In pattern recognition each input is associated with a response, and reward/punishment signals may be given for each decision of the pattern classifier. Such immediate and definite performance evaluation is not generally available in control problems, and not only is the performance measure averaged over many decisions (e.g., rms error) but also the optimum value is unknown. Various techniques have been proposed for providing performance feedback, but none is particularly satisfactory in that they involve additional storage, a pre-existent solution to the problem, or specialized knowledge about the plant parameters and structure. This paper describes in addition some novel experiments in which a second, competitive adaptive controller is used as a (time-varying) standard against which to evaluate the performance of the first, and vice versa.

(ii) Coding of input and response patterns Adaptive-threshold logic elements cannot realize all possible input/response transformations, and some are more difficult and less stable than others. In so far as the required control policy corresponds to a natural input/response transformation for the threshold-logic element, convergence to a satisfactory policy will take place most readily and rapidly. This is a typical example of the approach to synthesis engendered by the utilization of adaptive systems. Partial knowledge of the environment, e.g. some features of the topology of its state-space, may be totally insufficient to determine an optimum control policy, but may be quite adequate to suggest input and response codings which aid the adaptive controller in converging to an optimal policy.

(iii) The state-space of adaption and training The convergence of adaptive-threshold-logic elements in pattern recognition is a simple phenomenon, in which if a solution giving the required input/response transformation exists it is bound to be attained. In control systems attainment of an optimal policy becomes a function of the initial state of the controller and the environments in which it learns (there is no necessity for learning to take place only in the final environment), and an optimal policy, once attained, may not persist (i.e., may not be stable) since learning continues and maladaption is possible. These phenomena may be analyzed by considering the effect of performing in a given environment for a prescribed time, a 'task', upon the parameters of the threshold-logic element, the 'state' of the adaptive controller. The selection of a sequence of tasks which forces the controller to a state where it has a persistent, optimal policy for the final environment ia in itself a control problem—the training problem. This paper proposes a system for solving the training problem automatically, and gives experimental results on its application. This is another example of synthesis through adaption, by using partial knowledge of the environment to derive similar environments which are easier to learn in.

(iv) Linguistic communication with the controller Whilst certain prior information about the environment or control problem may be used to suggest reasonable input/output codings and training sequences, there is other information which seems best used to set up internal parameters of the controller. For example, a simple control policy may be known which, whilst sub-optimal, is at least stable, and would form a good initial policy from which to adapt. Internal parameters of the controller which might be made available for adjustment, however, need bear no obvious relationship to control policies, or even individual input/response connectives. A requirement to increase the gain of the controller of Figure 1 demands some change in the weights of the adaptive threshold logic element, but much further information is required about the controller before this change may be made directly. The human controller overcomes this problem indirectly through the use of language, and it is attractive to consider the possibility of setting up channels for linguistic communication between the control engineer and the adaptive controller. The present paper serves to indicate that such channels may be implemented, reasonably and simply, and demonstrates some pitfalls in the use of language.

(v) Hardware realization of the controller One advantage of adaptive threshold logic is that the convergence is steep descent in form, with feedback from the weights through the control policy and performance measure to the weights themselves. The changes in the weights made through adaption thus need not be very precise provided they are in the intended direction and not excessively large or small. This opens up the possibility of almost any physical process exhibiting 'memory' being used as the basis of an adaptive element. The most attractive basis at present, however, is the use of conventional digital silicon integrated circuits, amenable to large-scale integration at very low cost. Threshold logic elements using these circuits have the peculiarity that their weights are discrete and bounded, and it may be shown that convergence to a solution, even of a normal pattern-recognition problem, need not occur even though the solution is within the range of weights. This effect is demonstrated and a means of overcoming it outlined, which involves the deliberate introduction of random processes in weight changing.

The only existing adaptive controller which has been utilized in the manner suggested in this paper is the human operator, and, without implying that the structure of the human adaptive process is that of Figure 1, it is useful to compare the results of experiments on coding, competition, training, communication, and so on, for both human operators and pattern-classifying automatic controllers. The results are, in fact, striking in their similarity, which might be expected since an intensely adaptive controller is one whose behaviour is moulded largely by its environment.

The following sections develop and exemplify the problems and possible solutions outlined in these five categories.

Performance Feedback

A very simple form of performance evaluation is possible, if a second controller is available whose policy is satisfactory. It may be placed in control of the environment, with the adaptive controller receiving the same input but feeding its output only to a comparator which determines whether it is the same as that of the standard controller and rewards or punishes it accordingly (Figure 2). This reduces the control situation to a pattern-recognition one, and the adaptive controller is bound to come to mimic the standard one perfectly in so far as this is possible. One aspect of the behaviour which is of interest is the response of the adaptive controller to inputs it has not received whilst learning. Its modelling need be neither strictly in the frequency domain nor in the time domain, and its generalization will have peculiarities dependent on the input/output coding.

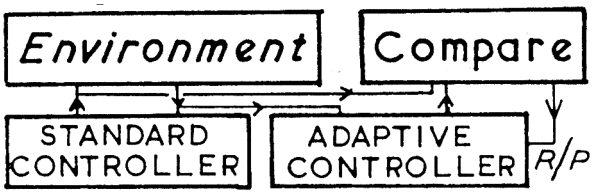

Figure 2: Learning by Example

The configuration of Figure 2 enables the adaptive controller to copy the policy of an existing controller, but not to form a novel and optimal policy. It is of practical interest only when the existing controller is itself a black-box requiring replacement, e.g. a skilled human operator (Widrow and Smith, 1968). The prime objective of current research on performance feedback in pattern-classifying adaptive controllers is to remove the need for a standard controller, or 'teacher' as it is sometimes (misleadingly) called. One ingenuous proposal has been to generate a Lyapunov function of the state variables and reward the machine after each action if this has decreased, punishing it otherwise (Mirabelli, Connelly and Worthen, 1964). This entails too great a knowledge about the environment to be useful in practice, and is in some ways a variation on the use of a standard controller since the Lyapunov function effectively defines one; the main difference is that it is the adaptive controller, rather than this standard one, which controls the environment during learning.

The type of performance directive one would like to utilize in practice is to minimize some error functional over an interval. This creates two sources of difficulty in that no absolute standard of performance is defined, and also the performance measure is global to a sequence of actions rather than local to each single one. Widrow (1966) has proposed a technique, called

'bootstrapping', for overcoming this second difficulty by assuming that the results of a global evaluation can be applied locally, and has justified this theoretically under certain conditions. In practice this entails storing all input/response pattern-pairs during the interval over which the performance is being evaluated, and this removes some of the simplicity and robustness of adaptive-threshold-logic hardware. An alternative to this, not previously proposed, is to triplicate the threshold-logic element, reward one continuously, punish another continuously, and use the third to make the actual control decisions during the interval over which the performance is evaluated. At the end of this interval the weight values in either the rewarded or punished element are transferred to the decision-making one, according as the performance is globally good or bad.

Widrow has applied bootstrapping successfully to game-playing environments where, even though the performance evaluation is global, it does at least take the form of a definite WIN/LOSE result. If the performance evaluation is the value of an error-functional, then further information is required before this can be fully evaluated as a GOOD/BAD result. Comparison with a fixed standard is not attractive because the optimum value is not known, and any fixed comparison is likely to lead to an overwhelming preponderance of reward or punishment initially such that constructive learning does not take place. It is possible to create a variable standard by turning the problem into an adaptive game in which two matched controllers each interact with their separate, but identical, environments for an interval, at the end of which their performances are evaluated and compared to determine the winner and loser.

The effects of this technique for obtaining performance feedback have been investigated in a simple control problem. A boat is required to cross a river 1000 ft.wide between two opposite terminals in minimum time. It moves at a constant speed of 5ft/sec and is controlled incrementally in angle every 125ft across the river. The river velocity is uniform at 1ft/sec, and the input to the threshold-logic element is a 20-bit binary pattern representing the distance of the boat from the bank and its angle to the river. Two pattern classifying adaptive controllers, identical in structure and parameters, cross the river together and their crossing times, μ, are compared. All the decisions of that which takes the least time are rewarded, whilst those of the other controller are punished.

Figure 3 shows typical performance curves for different controllers, in terms of the mean crossing time (over 10 consecutive crossings) against the number of crossings. In the first graph

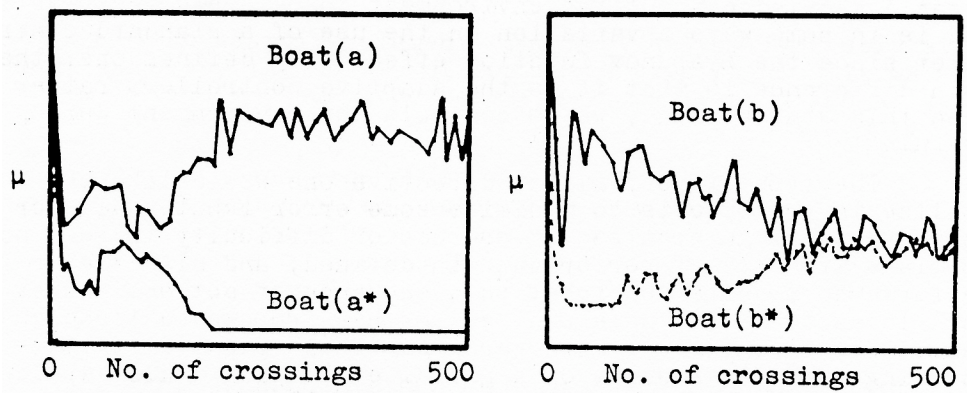

Figure 3: Performance of Competing Controllers

Boat (a*) converges to a near optimal solution whilst its competitor diverges. In the second graph both boats converge to a fairly sub-optimal solution. These results might reasonably be compared with similar phenomena which occur in the relative learning of individuals in small groups of schoolchildren. The stability of the competitive situation is now the subject of investigation, and these results are given to demonstrate that it is not, in itself, clearly stable or unstable.

Coding of Input and Response Patterns

A threshold-logic element with binary output divides the space of input patterns into two parts which are said to be linearly separable. The proportion of possible dichotomies which are linearly separable is asymptotic to zero as the dimension of the pattern-space increases. The mean number of patterns selected at random which can be arbitrarily dichotomized is about twice the dimension of the pattern-space, whereas the total number of patterns is 2 to the power of this dimension. These considerations demonstrate that the capability of an adaptive-threshold-logic element to realize a control policy with arbitrary codings at input and output is severely limited. In the same way that a human operator has pre-conceptions about the nature of the world which strongly influence his reaction to a control task, even when it is completely novel, the pattern-classifying adaptive controller has inherent assumptions about the topology of its environments and to the extent that they are satisfied convergence will be possible and rapid.

The topological properties of threshold-logic have been extensively documented (Dertouzos, 1965; Hu, 1965; Lewis and Coates, 1967), but matching them to those of the environment must take into account both its natural topology and that induced by the performance criterion. One source of guidance is that threshold-logic is closely related to maximum-likelihood prediction based on Bayes inversion of conditional probabilities (Gaines, 1967a), and linear separability is equivalent to the assumption that the components of the input-pattern vector are independent evidence for the hypothesis that the best output is a 0 or a 1. The most complete study of coding problems for pattern classifying adaptive controllers has been made by Smith for the situation where the required output is a binary nonlinear function of continuous inputs, not involving cross-product terms (Widrow and Smith, 1968; Smith, 1966).

The State-space of Adaption and Training

The expected behaviour of an adaptive controller when coupled to an environment is that, if its initial control policy is not satisfactory, then it will eventually become so (Zadeh, 1963). Thus it is possible to segment the interaction of the controller with its environment into at least two parts, in the first of which it is not satisfactory and in the last of which it has become so. This segmentation, which is inherent in the basic concept of adaption, may be extended to form a description of the full range of adaptive behaviour (Gaines, 1967b). A *task* is defined to be a segment of the interaction for which it is possible to say whether the controller has been satisfactory. At the beginning of a task the controller will be in some *state* which causes it to implement a particular control policy. At the end of a task it will be in some other state and implement another control policy. With a reasonable definition of the set of tasks, the final state and satisfactoriness of performance will be a function of the initial state and task, and the adaptive behaviour of the controller may be ascribed to an automaton whose inputs are tasks, whose states are controller states, and whose outputs are on the one hand control policies, and on the other the satisfactoriness of the control policy for a given task.

Being coupled to a fixed environment is equivalent to the adaption-automaton of the controller having a sequence of inputs consisting of the same task repeated, and this interaction is defined to be *acceptable* if it is eventually always satisfactory. An acceptable interaction is one which attains a stable condition of satisfactoriness, and in this stable condition the controller is said to be *adapted* to the input task. A controller in such a state that it will have an acceptable interaction with any one of a number of tasks is defined to be *potentially adaptive* to that set of tasks; for any of these tasks it will adapt to satisfactory performance through experience and without aid. Figure 4 illustrates how the state-space of an adaption-automaton is divided into regions by consideration of the trajectories generated in adaption to a task, t. The states for which the controller is adapted to t form a sub-set of those for which it is satisfactory, and also for those in which it is potentially adaptive to t. A trajectory through the state-space generated by giving the task t many times in succession, t^n, started outside the potentially adaptive region, at Y, may enter the region of satisfactory interaction but must eventually leave it, whereas started within the potentially adaptive region, at X, it will remain within that region, eventually entering the adapted region and never leaving it.

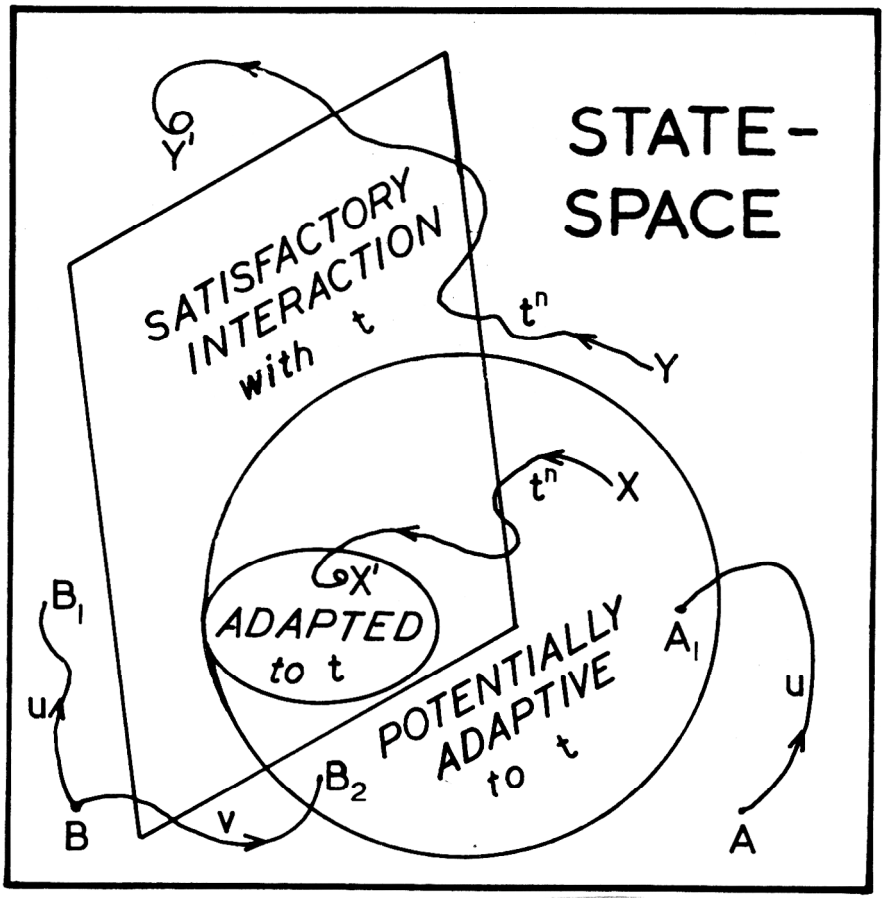

Figure 4 State-space of Adaption

When the controller's initial state is outside the region of potential adaption for a task, then learning will not take place given that task alone. Given some other sequence of tasks, however, the controller will adapt to them and in so doing may become potentially adaptive to the original task—the sequence of tasks has *trained* it for the original one. In Figure 4, the state A is outside

the region of potential adaption, but from it the trajectory generated by the sequence u terminates at A_1 within that region. From X, however, u does not generate a trajectory into the desired region, although there is another sequence v which does so. The selection of tasks to form a training sequence is itself a control problem in the state-space of the adaption-automaton, but the theory of adaptive behaviour does not, in itself, indicate how this problem may be solved.

The causes of mal-adaption lie in the basic epistemological problem of attempting to control an environment whilst at the same time learning about it in order to improve the control policy, the *dual control problem* (Feldbaum, 1966). A given control policy restricts the states and transitions of the environment to some sub-set of the total possible behaviour. The desired sub-environment generated by an optimal policy may be entirely different from that generated by the initial policy of a naive controller, and adaption which takes place in it may be irrelevant or even deleterious. Hence the aim in selecting a training sequence should be to maintain the desired sub-environment regardless of the control policy of the trainee. To do this requires feedback about the state of the trainee, and, if the environment is a linear dynamical system with a single output, maintaining the mean-error at a constant level may be shown to provide a feedback criterion having the required effect (Gaines, 1968a).

Figure 5 illustrates a training system for a third-order continuous environment (Gaines, 1966)—the additional control loop necessary to maintain the desired sub-environment is closed by a *training controller*, selected by a *trainer* having feedback from the performance of the trainee. The training controller damps some of the integrators to make the control problem simpler, and the amount of damping is varied automatically by the trainer. This consists in practice of a single integrator which integrates the modulus of the error minus a fixed tolerance. If the mean error is above this tolerance the output of the integrator increases and drives a servo which increases the damping, making the control problem simpler, whereas otherwise it is made more difficult. For controllers of fixed performance, therefore, the control problem is automatically adjusted in difficulty to an asymptotic value at which the controller's mean error is equal to the tolerance.

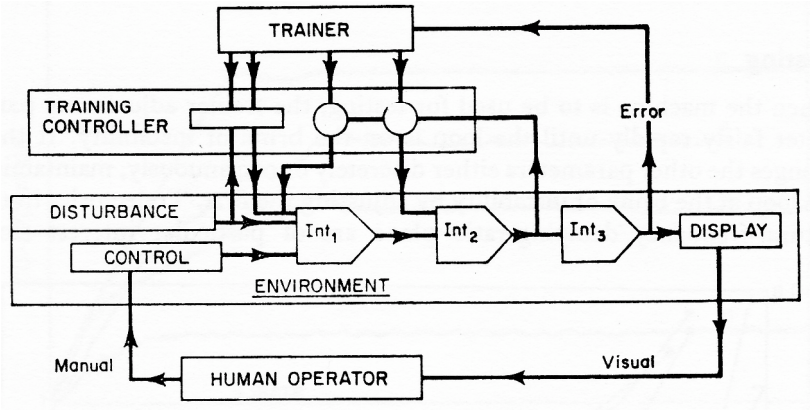

Figure 5: Automatic Training system

With adaptive controllers as trainees the performance will again be held constant, but the difficulty of the control task should increase as they learn. Figure 6 shows the variation in difficulty with time for three pattern-classifying adaptive controllers having the structure of Figure 1, controlling the three integrators of Figure 5. Machines A and B learn to high and medium levels respectively, whereas machine C shows unstable initial learning (cf the trajectory

from Y in Figure 4) eventually settling to a medium level. With regard to the efficacy of the training technique it may be noted that none of the machines learn a control policy which will enable them to perform at the tolerated level, even at zero difficulty, when they are trained in a fixed environment with a difficulty of 0.3 or greater. Similar results have been obtained with human operators training on the system of Figure 5 (Gaines, 1967b, 1968b).

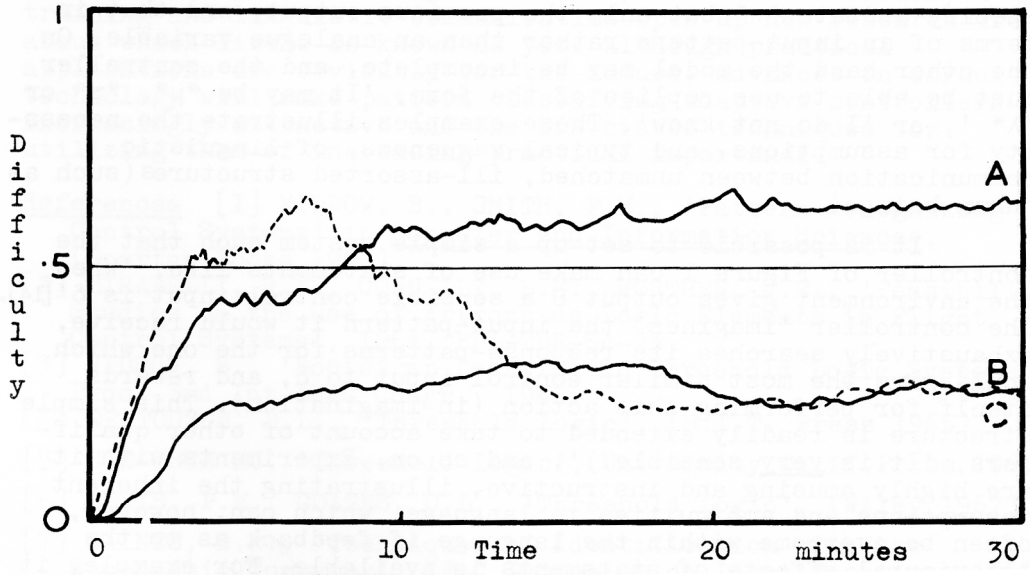

Figure 6: Automatic Training of Adaptive Controllers

Linguistic Communication with the Controller

Human language is a complex, multi-purpose structure which defies complete formal analysis. One approach to language which seems most fruitful in control situations is to consider the linguistic structure that must be set up to replace an existing, or hypothetical, physical link. For example, consider a controller which uses a 'fast model' of its environment to search for an optimum control sequence, such that the normal link between model and controller is broken and replaced by communication in a 'natural language'. The controller must be able to make statements of the form, 'If the environment is in state *s* and I use control action *a*, what will the next state be', and the model must be able to reply, 'The next state will be *n* '. This is language at a simple and apparently trivial level, but consider the similar, but more realistic, situation in which a controller exists, some form of model exists, and it is desired to have the one make good use of the other. Assuming that some mechanism may be set up in the controller to enable it to make use of knowledge about the effects of control actions, a typical problem will be that the representation of the environment in the controller is very different from that in the model and that it is unable to specify a state *s* but only the previous output, and that in terms of an input-pattern rather than an analogue variable. On the other hand the model may be incomplete, and the controller must be able to use replies of the form, 'It may be *n*, *n* or *p*', or 'I do not know'. These examples illustrate the necessity for assumptions, and typical vagueness, of linguistic communication between unmatched, ill-assorted structures (such as human beings).

It is possible to set up a simple system such that the controller of Figure 1 can make use of statements like, 'When the environment gives output *o* a sensible control input is *i* (Gaines,

1967c). The controller 'imagines the input-pattern it would receive, exhaustively searches its response-patterns for the one which would give the most similar control input to d, and rewards itself for performing this action (in imagination). This simple structure is readily extended to take account of other qualifiers, 'It is *very* sensible..', and so on. Experiments with it are highly amusing and instructive, illustrating the inherent assumptions and ambiguities in language, which can, however, often be overcome within the language if feedback as to the behavioural effects of statements is available. For example, it was found that a naive controller generalized a statement of the form, when the error is positive perform action-minus', into a control policy consisting of action-minus whatever the input, which retarded learning. Coupled with the dual statement about negative errors and action-plus, however, this greatly reduced the learning transient. A simple result, but one which was unexpected when the statement was first made.

Hardware Realization of the Controller

The most promising means of fabricating the adaptive-threshold-logic element in the controller of Figure 1 is all-digital, large-scale integrated circuits, using digital counters to store the variable weights. The standard convergence proof for adaptive-threshold-logic in a pattern-recognition situation (Novikoff, 1962), assumes a far greater potential range of weights than is necessary to ensure that a solution exists. If this range is essential to convergence then the threshold-logic element is forced to have redundant states in so far as its ability to realize a particular control policy is concerned. A simple example has been given elsewhere (Gaines, 1967a) to demonstrate that this redundancy is essential if a deterministic adaption algorithm is utilized, and an analytic proof is also given that no redundancy is required if a suitable stochastic algorithm is implemented. In fact convergence is guaranteed under the usual conditions whenever a solution exists within the range of weights, provided the normal, individual weight changes are made, independently, with a probability bounded away from both zero and unity. Effectively, random noise must be injected into the magnitude of the weight change, and this has been done, with the stated effect, in at least one hardware adaptive threshold-logic pattern classifier (Clapper, 1967).

Conclusions

This paper has ranged far and wide through the problems involved in using adaptive pattern classifiers as the variable elements of adaptive control systems, and it has raised far more problems than it has solved. It illustrates the new dimensions added to the normal pattern-recognition situation when there is feedback from the classifier's decisions to the sequence of input patterns, and when performance feedback is relative and global. The problems of performance-feedback, training and linguistic communication open up new research areas about which little is known as yet. Although practical applications are obviously for the future, advances in circuit technology will make pattern classifying adaptive controllers increasingly attractive and the associated techniques for utilizing them of increasing practical importance.

References

Clapper, G.L. (1967) 'Machine Looks, Listens, Learns', Electronics, October 30th, 1967.

Dertouzos, M.L. (1965) 'Threshold Logic', (M.I.T. Press).

Feldbaum, A.A. (1966) 'Dual Control Theory Problems', Proc.3rd IFAC Congr.

Gaines, B.R. (1966) 'Teaching Machines for Perceptual-Motor Skills', in Aspects of Educational Technology (Methuen).

Gaines, B.R. (1967a) 'Techniques of Identification with the Stochastic Computer', IFAC Symposium on Identification.

Gaines, B.R. (1967b) 'Adaptive Control Theory', Encyclopaedia of Information, Linguistics and Control, (Pergamon).

Gaines, B.R. (1967c) 'Automated Feedback Trainers for Perceptual Motor Skills', Final Report to M.o.D., 1967.

Gaines, B.R. (1968a) 'Training the Human Adaptive Controller', Proc. I.E.E., to be published.

Gaines, B.R. (1968b) 'The Control of Human Learning', IEEE Int. Conv.Digest.

Hu, Sze-Tsen (1965) 'Threshold Logic', (University Calif. Press).

Lewis, P.M. and Coates, C.L. (1967) 'Threshold Logic', (Wiley).

Mirabelli, R.E., Connelly, E.M. and Worthen, J.E. (1964) 'Feasibility Studies on the use of Artrons as Logic Elements in Flight Control Systems', FDL-TDR-64-23.

Novikoff, A. (1962) 'Convergence Proofs for Perceptrons', in The Mathematical Theory of Automata, (Wiley).

Smith, F.W. (1966) 'Design of Quasi-optimal Minimum Time Controllers', IEEE Trans.Aut.Contr. AC-11, 71-77.

Widrow, B. (1966) 'Bootstrap Learning in Threshold Logic Systems', Proc. 3rd Int. IFAC Congr.

Widrow, B. and Smith, F.W. (1964) 'Pattern Recognizing Control Systems' in Computer and Information Sciences, (Spartan Books).

Zadeh, L.A. (1963) 'On the Definition of Adaptivity', Proc.IEEE, 469-470.