General Constraints on the Architectures of Functionally Complex Learning Systems: Implications for Understanding Human Memory

L. Andrew Coward
School of Information Technology, Murdoch University
South Street, Murdoch, WA 6150, Australia

Abstract

The human brain is a functionally complex system. It can be demonstrated both on theoretical grounds and from design experience with functionally complex electronic systems that practical considerations such as the needs to limit information recording and processing resources and to be able to make functional changes without excessive undesirable side effects place severe constraints on how information is stored and processed in any such system. A system which must heuristically define its own functionality (i.e. learn) is constrained within a set of architectural bounds called the recommendation architecture. The processes by which information from the environment is recorded and used to generate behaviour in a system with the recommendation architecture are shown to provide an explanation for a range of observations of relatively simple human memory and skill learning capabilities. It is concluded that the recommendation architecture is a potentially valuable approach to understanding more complex human memory phenomena.

1 Introduction

There are a number of approaches to understanding human memory, including psychological experiment, investigation of neuron processes, and electronic simulation of memory like phenomena using artificial neural networks. This paper offers an approach based upon the understanding of general architectural constraints on functionally complex systems and the consequences of these constraints for information handling within such systems.

Examples of functionally complex electronic systems include the systems which control extensive physical plant such as telecommunications networks or chemical processing facilities in real time with minimal human intervention. Such systems can include tens of millions of lines of software and require many thousands of closely coordinated man years of design engineer efforts to implement. The design of such systems represents a radically different problem from the design of information technology systems, and knowledge of the required design processes is much less widely available.

The architectural form of such systems is very severely constrained by practical considerations such as the need to limit information recording and processing resources and the need for ongoing modification of some functionality without side effects on other functionality. The brain has some analogous constraints, including natural selection pressures in favour of forms which can perform equivalent functionality with fewer resources and the need to learn without interference with prior learning.

Although the brain has minimal direct resemblance with functionally complex electronic systems, the way in which practical constraints affect the architectures of such systems can provide guidance on how analogous constraints affect the architecture of the brain. For functionally complex electronic systems in which functionality is defined under external intellectual control, these practical constraints result in architectural forms including modular hierarchies with specific criteria for how modules must be defined, and a separation between memory and processing. For functionally complex systems in which functionality is defined heuristically, the analogous considerations lead to specific architectural forms with some similarities to and some major differences from those in electronic systems.

This paper describes the architectural forms into which systems that heuristically define their own complex functionality are constrained, and summarizes the reasons for their emergence. The paper then goes on to demonstrate how the ways in which information must be recorded and processed in systems with such forms generate the phenomenology of human memory.

2 Functional Complexity

A functional system is one which acts upon its environment and itself in order to achieve objectives or target states for itself and the environment. A functionally complex system is one in which firstly there are large numbers of possible behaviours, secondly there are multiple potentially conflicting objectives, thirdly considerable interaction is required within a large body of information derived from input sensors to determine appropriate behaviour at each point in time, and fourthly the time available in which to determine behaviour is short.

An example of a functionally complex system is a telecommunications central office switch [Nortel Networks, 2001]. One such switch can provide telecommunications services to over 100 thousand telephones and computers. Behaviours include assigning subsets of a very large number of

connectivity resources to establish links between users, performing diagnostic tests and failure recovery actions, measuring current traffic volume to determine the need for additional resources, modifying the services provided, and collecting billing information. Objectives include rapid establishment of connections with the appropriate quality of service, keeping system downtime well below one minute per year, and accurate billing. The need to complete calls quickly may during busy periods come into conflict with the need to perform regular diagnostic tests to detect problems before they cause system outage. The actual resources used for the link established between two users depends upon an interaction between information on the location of the users , the type of service required, the connectivity resources already in use for other calls, and the diagnostic status of currently unused resources. Finally, dial tone must be provided within less than a second, and a connection established within a few seconds of dialing, even if thousands of other calls are also in progress. The central office switch thus meets the multiple behaviours, multiple conflicting objectives, large internal interaction to determine behaviour, and limited processing time criteria for a functionally complex system. It is important to note that with the given definition of functional complexity, most information technology systems are functionally relatively simple (although still difficult to design and implement).

The human brain is another example of a functionally complex system. Consider for example an encounter between two people called Rick and Elaine in which Rick's brain must process visual inputs to respond appropriately to Elaine. The activation of Rick's retina is constantly changing as both Rick and Elaine move, and as Rick's attention shifts between a range of people and objects in his current environment. If Elaine is in an environment very familiar to Rick, the appropriate response might be to say "Hi Elaine, I didn't know you came here". If Elaine is with people known to Rick, the response might be "Hi Elaine, I didn't know you knew these people". If Elaine appears unwell, the response might be "Hi Elaine, are you feeling okay", but the appropriateness of this response could be affected by the identities of the other people present. Rick's response will depend on his own priorities (to improve his image with Elaine or with other people present, or to continue with what he is already doing without interruption). However, Rick's priorities may be affected by Elaine's expression, gestures etc. Hugging, handshaking or kissing may be appropriate in addition to words. Finally, Rick has one or two seconds in which to determine appropriate behaviour. Thus the human brain also meets all the criteria for a functionally complex system. However, in the case of the brain, the system must heuristically define or modify its own behaviour (i.e. learn).

3 Constraints on Functionally Complex Systems

There are a number of constraints on functional systems which tend to become more severe as functional complexity increases. One constraint is the need to keep information recording, information processing and connectivity resources used within limits. A second constraint is that it must generally be possible to modify the system functionality in some appropriate ways without introducing undesirable side effects on other functionality. In electronic systems such modifications are made under external intellectual control, and undesirable side effects must be identified by extensive regression testing of all system functions and corrected under external intellectual control. In artificial neural networks (ANNs), a major problem is interference between early and later learning [e.g. French 1999]. This interference can be catastrophic in the sense that it results in the complete destruction of earlier learning. Such catastrophic forgetting limits the complexity of the tasks which ANNs can learn to perform.

Modular Functional Hierarchies

For reasons discussed in detail in Coward [2001], the combination of resource constraints and need for modifiability without side effects forces functional systems into modular hierarchies. A module is a component in a system which is optimized for a particular group of system operations. In a modular hierarchy, overall system functionality is separated into modules, those modules into submodules and so on all the way down to the most detailed operations of the system. In order to make modifications to functionality possible without excessive side effects, modules must be defined in such a way that the information exchange between modules required to coordinate their separate operations is minimized as far as possible. Because of the strong interaction between functions in a functionally complex system, modules which corresponded exactly with system features would be unlikely to satisfy the need to minimize information exchange without using excessive resources. As a result, modules in such a system do not correspond with system features or categories on any level. Rather, the effect of the definition process is that on average a system feature utilizes a relatively small number of modules, and on average a module is utilized by a relatively small number of features.

In the relatively functionally simple set of applications provided by a personal computer, it is possible to define modules (or applications) which correspond closely with major system features.

These applications handle features like word processing, accounting, web access etc. and the information exchange between two applications can be limited to file exchange between the applications when neither application is actively processing information. In contrast, in the central office switch described earlier, even though the need for modifiability forces the software of such systems into modular hierarchies [Kamel 1987], no individual modules on any level correspond with features like conferencing or call forwarding, or with packages of such features for particular types of customers like call centre services, or even with major separations like call processing and diagnostics. It is plausible that functional complexity also accounts for the difficulty in associating physiological or other structures in the cortex with cognitive functions or features [e.g. Kingsley, 2000].

To understand the modifiability issue better, consider the information exchange between modules in more detail. There are two equivalent ways of understanding a specific exchange between two modules. One is that the source module has detected a specific condition in the environment and/or in the internal state of the system. The other is that for the recipient module, the set of behaviours within which the currently appropriate behaviour is located has been limited to a subset of the full range influenced by that module. The modifiability issue can then be understood as deriving from the fact that, in a functionally complex system, a condition detected and communicated by one module may have different functional meanings in each of the modules receiving the indication that it has been detected. A change to the definition of a condition for one functional purpose could therefore introduce undesirable side effects in all the other modules using the indication of the original condition for other functional purposes.

Although not all changes will necessarily introduce undesirable side effects, any such side effects must be corrected. If a functional change to a module on average introduced side effects in two other modules, and the correction of those side effects in each of those two modules on average introduced side effects in two more modules, the exponential growth in side effects would make any functional change impractical. The modular hierarchy with minimized information exchange can thus be seen as being required to limit the proliferation of side effects. It could of course be argued that condition detection must be duplicated until the detection of a condition only held one functional meaning, but this approach conflicts with the need to limit resources, and in a practical system resource limitations will always compel multifunctional use of condition detections. However, information exchange can always be reduced by some independent detection of shared conditions, and the modular hierarchy will be the best compromise between resource usage and information exchange which can be identified.

Unambiguous and Ambiguous Information Exchange

One further issue is the degree of functional confidence assigned to information by recipient modules. In other words, the detection of a condition by some source module could indicate to the recipient that currently appropriate behaviour was within a defined subset with 100% confidence. Alternatively, the indication could be probabilistic, in other words the appropriate behaviour was more likely to be within the defined subset than elsewhere. If information exchanges carry 100% confidence (or are unambiguous), such exchanges can be interpreted as commands or instructions, and the need to ensure that such a level of meaning is maintained forces the system into the familiar memory/processing form of the von Neumann architecture ubiquitous in commercial electronic systems [Coward 2001]. However, the unambiguity makes heuristic modification of functionality (or learning) impractical. To understand this point, imagine how learning could occur in a modular hierarchy. Such learning implies that at least one module must change, resulting in a change to its outputs. Such changes will have side effects on many other behaviours. The information available to guide changes is limited to supervised learning (i.e. teaching) for some limited initial period and otherwise to only feedback of positive or negative consequences. A process which depended upon consequence feedback to compensate for the proliferation of side effects under conditions in which every information exchange was functionally unambiguous would be very unlikely to converge. This impracticality is confirmed by the failure to implement learning in any non-trivial von Neumann based system.

If information exchanges are probabilistic, in other words if the detection of a specific condition is partially ambiguous in functional terms, such exchanges can only be interpreted as recommendations. The implication of the use of ambiguous information is that in order to achieve high integrity behaviour, a much wider range of conditions must be detected, and the summation of the recommendation strengths for all recommended behaviours will indicate the most appropriate. An implementation using ambiguous information will therefore require many more resources than one using unambiguous information. It will not in general be possible to mix ambiguous and unambiguous information in the same modular hierarchy, since with a complex pattern of information exchange the presence of some ambiguous information will render all information ambiguous.

However, when information exchange is partially ambiguous, any accepted behaviour will in general have multiple recommendations generated by multiple modules. Changes to conditions detected by one of these modules to implement a functional change to some other behaviour will therefore have less effect on existing functionality than when information exchange is unambiguous and every module output is a command. Learning is therefore more feasible. However, even though partial ambiguity is tolerated, an adequate level of meaning must be maintained.

To illustrate the issue of maintaining adequate levels of meaning, consider a simple human example. Suppose that I have an understanding with my wife that when I leave a telephone message asking her to call me at the office, an emergency has occurred. Suppose also that this understanding is completely unambiguous: I will never leave such a message in the absence of an emergency. In this situation, if I leave a message she will stop whatever she is doing and act as if there is an emergency. If one day I just wanted to chat, leaving the message will result in an inappropriate response. In other words there is no scope for learning. However, if leaving a message was partially ambiguous, then the tone of my voice, the exact wording, my mood when I left that morning, and her knowledge of the probability of an emergency at that particular time could all contribute to selection of a different response. In other words, learning is possible. There are nevertheless limits to the degree of change. I cannot leave a message in a foreign language and expect a call, and I cannot leave the usual message and expect to be called at the new number I have just been assigned. So the critical issues for context maintenance are how much the form of the message generated under the same conditions can be changed, and how much the conditions under which the same message is generated can be changed without destroying the functional usefulness of the message to recipients.

As functional complexity increases, the average degree of change which can be tolerated without unacceptable loss of meaning will decrease. For example, at the most detailed level in the modular hierarchy there must be devices which can select, record, and detect combinations of their inputs. Suppose that one such device recorded its first combination and indicated its presence by producing an output. Devices in many other modules might then select that output to be part of their own combinations. In general the source device has no "knowledge" of the functional meanings derived from its outputs by its targets. If that source device subsequently changed its programmed combination, the meaning assigned to the output by its recipients would be less valid. Degree of change to a device must therefore be "slight" and located on as "few" devices as possible.

Degree of Change

The minimum change situation would be if a device could only be programmed with one combination, and only produce an output in response to an exact repetition of that combination. Such an approach would be extremely expensive in terms of device resources. A slightly greater degree of change would be if a device could be adjusted to produce an output in response to any large subset of its original combination. A further degree of change would be if a small proportion of inputs could be added to an existing combination, for example, additional inputs occurring at the same time as a high proportion of the combination. A greater degree of change would be if the device could be programmed with multiple semi-independent combinations with inputs drawn from a population of possible inputs which tended to occur frequently at the same time. Note that for all these change algorithms, an exact repetition of any combination which has previously generated an output will always again generate an output. Higher level modules are made up of groups of devices, so in general an exact repetition of an input to a module which produced an output before will produce an output again. In other words, the portfolio of conditions with which modules are programmed can be expanded but not contracted.

Change algorithms based on adjustment to relative weights must have some criterion for weight change. Typically that criterion is some kind of feedback on the appropriateness of the device output under current conditions. The problem with this approach is that "appropriateness" tends to be based upon one type of expectation at a time (e.g. consequence feedback for one behaviour). If the device output can influence multiple behaviours, each change for one behaviour means that the device may not produce an output in response to the correct condition for another behaviour. The undesirable side effects of changes will for this reason increase rapidly as functional complexity increases. Coward [2001] has argued that this is the primary reason for the catastrophic forgetting observed in neural networks in which learning is based on adjustment of relative device weights.

The conclusion is that given the need to learn a sufficiently complex set of functions, devices will be forced to adopt learning algorithms which assure that in general any exact repetition of an input which generated an output in the past results in an output. Any expansion of the portfolio of inputs which can generate outputs from a device will need to be carefully managed to avoid excessive dilution of functional meaning.

Change Management

There are two possible indicators of a need for change. Some recommendations must be generated in response to every input state, even if the recommendations are interpreted as "do nothing". If no conditions are being detected by the modular hierarchy, portfolios must be expanded until some conditions are detected. A second indication is that if over some period of time detection of a specific group of conditions has been interpreted into behaviour, but consequences have sometimes been good and sometimes bad, the implication is that the group does not have the ability to discriminate between similar but functionally different input states. The solution is to add new conditions in the same input space to the group, with the objective of providing additional discrimination. Existing conditions are not changed, and other behaviours recommended by those existing conditions will not be affected.

Determining the most appropriate location for change in the modular hierarchy is itself a functionally complex problem which must be managed by the modular hierarchy. Detection of conditions by some modules must be interpreted as change recommendations by other modules. At the device level there will be change management inputs. These change management inputs will indicate the general level of activity in some source module rather than the presence of specific information conditions, and will excite or inhibit changes to conditions but will not themselves form part of conditions.

Although the need for and location of change can be managed, the selection of the exact change will always have some random element. For example, definition of a condition as one set of active inputs rather than another at the device level will depend upon the physical inputs which the device happens to possess. The selection of these inputs cannot be fully deterministic, because the system does not have the knowledge to guide fully deterministic selections. However, there are ways to bias the selection of inputs to improve the probability of functionally useful selections [Coward 1990]. This non deterministic definition of conditions is another way of understanding information ambiguity.

The Form of the Modular Hierarchy

In order to manage both external behaviour and change, detection of combinations with many different orders of magnitude of complexity will be required. In this context, the complexity of a combination is defined as the number of raw sensory inputs which ultimately contribute to the combination. To illustrate this point, consider the management of behaviour in response to input from a retina. Although combinations will not correlate unambiguously with behaviourally relevant conditions like features, objects, or groups of objects, combinations with the same order of complexity as such conditions are likely to be most useful in recommending behaviour. The need to detect combinations on different orders of complexity plus the need to ensure that combinations of the same order detected at the same time correspond with one input state at one time will result in devices being organized into layers [Coward 2001]. The level of module above device will be a small area on a layer, and the level above that will be a sequence of such areas across several layers (i.e. a column). The column detects a portfolio of functionally ambiguous conditions in an input space, and different layers within the column excite or inhibit change in the column or in other columns. Column outputs indicate the presence of conditions at some behaviourally useful order of complexity. Arrays of columns are organized so that one or more columns within the array generates an output in response to every input state, adding new columns or expanding the portfolios of existing columns as necessary. Modules made up of sequences of arrays detect the presence of conditions on a number of different behaviourally useful orders of complexity, with the condition portfolios optimized for some broad type of behaviour. Examples could be aggressive, fearful and food-seeking regions. This modular structure has some resemblances with the cortex [Calvin, 1995], and an implemented electronic version of such a modular hierarchy exhibits processes with similarities to human cognition [Coward 2000; 2001].

Selection of Behaviour

The modular hierarchy detects information conditions within input states. Each such condition can be interpreted as a recommendation for many different behaviours. The system must also select one specific behaviour in response to each input state, and must therefore learn to associate different combinations of conditions with different behaviours. Given the ambiguity of all information within the modular hierarchy, some separate subsystem must make the selections, and the selections must be based on some information over and above that contained in the input state. The three possible sources for such information are a priori (e.g. genetically programmed), supervision (e.g. teaching or imitation) and consequence feedback. However, the use of such information means that it would not be possible to support an internal pattern of information exchange with complex functional meanings. The subsystem must therefore be organized into modules which each correspond with individual behaviours or types of behaviour, and the functional meaning of an input to a module from another module within

the subsystem or from the modular hierarchy is either to encourage or to discourage the behaviour corresponding with the target module. At the device level, inputs will be excitatory or inhibitory corresponding with encouraging or discouraging the corresponding behaviour. The relative weights of such inputs will be continuously adjusted by, for example, consequence feedback. In order to reduce the overall requirement for connectivity, the subsystem will tend to be organized into separate subsystems managing selection of the input space to the modular hierarchy (i.e. the attention function), selection of general type of behaviour, and selection of specific behaviour within an already selected general type [Coward 2001].

4 The Recommendation Architecture

The modular hierarchy clusters experience into portfolios of similarity conditions which can be interpreted as behavioural recommendations and is therefore called clustering. The separate subsystem uses consequence feedback etc. to manage competitions between alternative recommendations and is therefore called competition. The separation between clustering and competition is called the recommendation architecture. Coward [1990; 2000] has pointed out the resemblances between clustering and the layers, columns and areas of the cortex, and between attention management, general behaviour selection and specific behaviour selection and the thalamus, basal ganglia and cerebellum respectively.

Information Recording and Activation Mechanisms in the Recommendation Architecture

In clustering the identity of specific conditions within input states is recorded permanently and changed relatively slightly. This information is functionally ambiguous, individual elements of information do not correlate exactly with cognitively useful features or categories. Such information will be activated in response to a repetition of similar information in any future input state. Any behaviours implemented in the past in response to an input state which contained an element of recorded information will be interpreted by competition as a recommendation for or against those behaviours depending on the consequences of those behaviours at that time. The behaviour with the strongest total recommendation will be implemented.

Recording of information in clustering is organized by similarity, with new information being recorded in the module which contains the most similar information. As a result, it is possible for information not actually present in an input state but similar to such information to be activated. Furthermore, it would be possible to activate information which was often active in the past at the same time as currently present information, provided that appropriate connectivity were created at the past time. Such secondary activations would expand the spectrum of recommendations available for competition to consider. However, the decision to generate such secondary activations is itself a behaviour which would in general need to be adequately recommended by the existing activation in clustering and accepted by competition.

Competition records the relative weights assigned to different outputs from clustering in favour of different behaviours. These weights change continuously and no record of past weights is preserved.

Information Recording and Activation Processes in the Recommendation Architecture

Given the recording and activation mechanisms described in the previous section, there are a number of major processes using these mechanisms to support generation of behavior. All these processes have been implemented in an electronic version of the recommendation architecture [Coward 2001].

The first process is finding and/or defining conditions within an input state. In this process, arrays of columns at every level are constrained to generate an output in response to an input state. Column portfolios are expanded or new columns defined until such an output results. Activity levels within specific layer submodules of columns define where changes will be made.

The second process is activating conditions by similarity. In response to detection of conditions that recommend a particular general type of behaviour, a signal is generated which lowers the thresholds of all devices in the region module corresponding with the type. This lowering increases the strength of recommendations of the type. Coward [1990] has suggested that this type of process models generation of flight behaviour by fear, or generation of food seeking behaviour by hunger etc.

The third process is activation of conditions on the basis of past temporal coincidence. From a functional point of view this is equivalent to supplementing behavioural recommendations based on conditions in the current input state with behavioural recommendations based on conditions which have often been present in the past in the same input states as the currently present conditions, or in different past input states which were close in time. There are two ways in which a temporal association could be defined. One way is permanently during some (probably supervised) learning

period. The second way allows constant adjustment of the associations to reflect some average over the past, with a bias in favour of the recent past. Coward [1990] suggested that word recognition could be an example of the first way, and associative recollection an example of the second, and support for this suggestion has been gained from electronic simulations [Coward 2001]. On this model, in word recognition the portfolio of conditions detected in a verbal input are associated permanently via a competitive subsystem with conditions at some levels of complexity detected in visual inputs which occurred at the same time as the verbal input. Future hearing of the word can therefore result in activation of visual conditions active in the past when the word was learned, and these visual conditions can recommend behaviours appropriate to the visual object. Note that the secondary activation will not include visual conditions with complexities close to raw visual input, in other words a visual hallucination does not occur. The model for associative recollection is that if two objects were perceived at similar times, the conditions detected in the perception of one object would acquire weights in a competitive subsystem to activate the conditions detected in the other object and vice versa, but these weights would decay with time.

A fourth process is using supervised learning and/or consequence feedback to associate conditions with behaviours. In supervised learning the system is given an indication of the currently appropriate behaviour. The weights in competition of the currently present conditions detected by clustering can therefore be adjusted in the corresponding module to favour that behaviour. In consequence learning the only indication given the system is whether the selected behaviour was good or bad. Only the weights into the module corresponding with that selected behaviour can be adjusted.

A sixth process is managing resources within clustering. There are two aspects to this process. One is that because module portfolios are defined heuristically, the system cannot know a priori the level of resources required to record the information needed to define portfolios at, for example, the device or column level. Some means must exist to manage the assignment of overall clustering resources to modules requiring them. Coward [1990] suggested that a simple mechanism would be a map of clustering resources which could be assigned as required over time, and pointed out that a number of amnesia phenomena were consistent with the view that the hippocampus plays that role for the cortex. The second aspect of resource management is assignment of appropriate connectivity to new and existing clustering resources in such a way that overall minimization of information exchange is maintained. Such minimization implies that connectivity between modules only be established if there is high probability of it being functionally relevant. Coward [1990] suggested that the only available indicator of probable relevance is past simultaneous activity of source and target modules, and that such information was best generated periodically by a partial rerun of past experience. This led to the suggestion that one role of REM sleep is global information exchange minimization.

5 Understanding the Phenomenology of Memory

Although the recommendation architecture approach can be applied to understanding sophisticated human memory processes, for the purposes of this paper its capabilities will be demonstrated with relatively simple phenomena. Experimental work in the areas of memory and skill acquisition has often been interpreted in terms of distinctions between implicit and explicit mental processes [Kirsner et al. 1998]. Although there has been criticism of these distinctions, they remain a useful categorization of some different ways in which human beings are observed to detect and store information and will be used as the memory process examples. The distinction had one of its origins in the observation of the kinds of memory tasks amnesiacs can and cannot perform [Dunn 1998]. In amnesiacs such as patients with Korsakoff's syndrome [Butters 1984], the most obvious symptoms are the inability to create new event memories. However, an ability to learn skills exists, although no memory of the skill learning experience is created. For example, performance in solving the Tower of Hanoi problem steadily improves over a number of sessions, even though at the beginning of each session the patient has no memory of seeing the problem before [Cohen et al. 1985].

Implicit and Explicit Memory

Laboratory testing of explicit memory is of two types: free recall and recognition. In free recall, subjects are presented with a list of words or other objects and in a subsequent test phase are asked to recall the items on the list. Recognition testing is similar except that in the test phase subjects are supplied with items and asked if they occurred on the original list. In a typical test of implicit memory, subjects are presented with letter strings which may or may not be English words. The subject is asked to classify each letter string as a genuine word or meaningless string, and the time taken to reach a decision measured. In the first phase of the experiment a series of strings are presented. In the second phase another series containing both strings from the first series and new strings are presented. It is found that the reaction time to classify repeated strings is shorter than for new strings.

There are a number of dissociations between explicit and implicit memory which indicate separate underlying mechanisms. In explicit memory, accuracy declines systematically as a function of time [Scarborough et al. 1977; Tulving et al. 1982] but decline for implicit memory is much less [Kirsner 1998]. Explicit memory is associated with the ability to describe the process verbally, while implicit memory is not [Schacter and Graf 1986; Shimamura 1986]. Performance in explicit memory testing is strongly affected by organization of the material to be remembered, but such organization has minimal effect on implicit memory [Schacter et al. 1989]. Implicit memory is more strongly reduced by changes to the lexical form of words than explicit memory [Forbach et al. 1974].

In the recommendation architecture model for implicit memory, the visual perception of a letter string generates the activation of a population of previously recorded information conditions which exist within the current visual image of the string. These information conditions might be on levels of complexity roughly equivalent to letters, letter pairs or triplets etc. In the case of a genuine word, this activated population has existing weights into competition subsystems recommending activation of visual conditions (at relatively high complexity) which were originally present within visual inputs from the actual object corresponding with the word. The activation of such a secondary population is itself a condition which can be detected by clustering. The presence or absence of such a condition is interpreted by a competitive subsystem as a recommendation to say "yes" (a genuine word) or "no" (not a genuine word). The accepted recommendation results in consequence feedback which increases the weights of active conditions into the simple verbal behaviour. However, some of the original conditions present in the visual string could also acquire weights into the selected behaviour. As a result, a repetition of these conditions could generate the correct behaviour without the need to wait for the presence or absence of the secondary activation. In other words, response time will be reduced. Because the conditions acquiring the weights will be derived from the specific string, the population derived from a lexically different string may not have enough weight to generate the behaviour and the faster response will not be present. Since the weights are specific to the one selected behaviour, there is no reason why the weights should change with time.

In the explicit memory experiments, there are two types of memory tested: recognition and recall. In the recommendation architecture, the more novel an experience, the more additional conditions will need to be recorded in order to generate an adequate spectrum of recommendations from clustering. The degree of recording required is therefore a good and readily available indicator for degree of novelty. This is consistent with observations that human beings have an extremely high capability to determine whether an object is novel. For example, subjects given a few seconds to examine each photograph in a set of 2500 could pick the familiar photograph from pairs in which only one came from the set at an accuracy level of 90% [Standing et al 1970].

The process for recall in the recommendation architecture is generation of a secondary activation which contains information activated during the past experience to be recalled. Conditions within this secondary activation could be interpreted as recommendations for behaviours appropriate in response to the past experience, including verbal descriptions of the experience. As described earlier, such secondary activations are generated via weights into a competitive subsystem which are given to conditions in any current activation. Such weights encourage secondary activation of information frequently active in the past at the same time as the currently active conditions.

It is important to emphasize at this point that each element of information is an abstract from some experience, but elements do not correspond unambiguously with cognitive features or objects. One element could easily contain sensory input information derived from a number of such features. A behaviour will only be activated in response to the sum of the weights into that behaviour from many activated elements.

Unlike the Standing work, the explicit memory experiments described earlier are not simply testing for the ability to determine the familiarity of individual objects, since all the words on the lists would have been familiar. The accounting for these experiments in the recommendation architecture depends rather on two factors in the experience of presentation of the lists. The first factor is that information derived from the words, the physical form of the lists, and the test environment would all be active at the same time. The second factor is that there may be some recording of information, but new elements will tend to abstract novelty in the experience, such as juxtaposition of words on the list and the environment in which the lists were presented.

In recognition memory testing, a word from an earlier list is presented, and information derived from the word is activated. This activated information is accepted as a recommendation to activate secondary information recently active at the same time in the past. If this secondary information contains a significant level of information recommending, for example, verbal description of the physical form of the list or of the environment where the original list was presented, this activation is accepted as indicating that the word was on the list. Absence of such recommendations indicates that

the word was not on the list. Recall memory depends upon a series of secondary activations. When a test subject is asked to remember the words on the list, the verbal input might activate a limited population of information elements present in past test environments. Some of this information might activate a tertiary activation which contained some information present in the physical form of the list in the recent presentation. This activation might in turn activate a population containing some information present in the perception of words. If a population contains enough information to generate an accepted recommendation to speak a word, that word is identified as being on the list. Because the strength of the weights generating the secondary activations depends upon how recently the simultaneous activation occurred, recognition and recall memory will decline with time.

The ability to generate secondary activations relevant to both the recognition and recall tasks is enhanced if there is significant recording of additional information during the experience of the lists, provided that the information elements include cross information between words or between words and the environment. For example, lexical organization means that information might be recorded from multiple words. These additional elements will tend to increase the strength of secondary activations for all the words involved.

The recommendation architecture approach thus provides a qualitative account for the observed differences between implicit and explicit memory.

Implicit and Explicit Skill Learning

Two concepts used to describe cognitive skill acquisition are declarative and procedural knowledge. Declarative knowledge is defined as knowledge of objects and events, while procedural knowledge is defined as knowledge of how to perform a cognitive skill. Declarative and procedural knowledge have also been distinguished on the basis that procedural knowledge can be used without "consciousness" in many cases [e.g. Anderson 1993]. The definition of consciousness is highly controversial, but for the purposes of the current discussion will be interpreted as the ability to make a verbal report of the relevant processing occurring in the brain. This interpretation is generally consistent with the accessibility distinction between conscious and unconscious processes [e.g. Clark 1992]. Although cognitive skills can be acquired without acquisition of new declarative knowledge, as demonstrated by the earlier discussion of Korsakoff's syndrome patients learning to solve the Tower of Hanoi problem, declarative knowledge is advantageous in many situations. Declarative knowledge can speed up the learning process when constructed on-line during skill learning [Mathews et al. 1989]. However, there can be inconsistencies between procedural and declarative knowledge of a skill, as revealed by studies of specialist expertise. "In some areas of expertise, there is a dissociation between what experts say they do and what they do ..... [An expert verbal] description typically bears only a superficial relationship to the expertise" [Speelman 1998]. Verbal descriptions by experts describing their expertise often correspond with beginner methods rather than actual methods, and requiring a verbal description can even result in the expert reverting to the less effective beginner method [Bainbridge 1977; Berry 1987].

In the recommendation architecture model for learning a skill, conditions activated within clustering in environments where the skill is relevant must acquire weights in competition associated with skilled behaviours. These conditions will generally include both new information elements resulting from novelty in the environments and information elements recorded in prior experiences. Although the new elements may be particularly useful for recommending the new behaviours, some learning would be possible using only previously recorded elements which happen to occur in the new environments. Thus Korsakoff's patients lose the ability to define new elements but can still learn new skills.

In practicing a skill, an activated population generates an action, and the sensory input from the experience of the action and its results generates a new action and so on. At each stage the weights assigned to the elements in the population into the module corresponding with the accepted action in competition will be adjusted by consequence feedback. In verbally describing a skill, an activated population generates a verbal output plus an activated population corresponding with the next step, which in turn generates the next verbal output and the next activated population and so on. The populations activated in the course of generating verbal descriptions may not be exactly the same as those generated in the course of skilled behaviour, and the weights of the information elements into verbal behaviour are not directly influenced by consequence feedback in response to skilled behaviour. Verbal descriptions can therefore be misaligned with the skill. However, when the skill is first learned, activated populations generated by verbal inputs may be a useful starting point from which to learn the populations to generate skilled behaviour. Information on behavioural weights is within competition, and the information cannot be directly activated in clustering to generate verbal descriptions. Such verbal descriptions can only be updated by self observation during practice of the skill. No record exists of past values of weights, and past states of skill are therefore generally inaccessible.

The recommendation architecture approach thus also provides a qualitative account for the observed phenomena of skill acquisition.

6 Conclusions

Any system required to learn a sufficiently complex combination of functions will tend to be forced into an architectural form called the recommendation architecture. Major structures of the human brain resemble those of the recommendation architecture in both form and function. The type of information recording and activation mechanisms required in a system with the recommendation architecture result in processes for generation of behaviour which strongly resemble the phenomenology of relatively simple memory and skill learning in human subjects. It is concluded that the recommendation architecture approach has potential value for understanding more complex human memory phenomena.

7 References

Anderson, J. R. (1993). Rules of the Mind. Lawrence Erlbaum Associates. Hillsdale, NJ.

Bainbridge, L. (1977). Verbal reports as evidence of the process operator's knowledge. International Journal of Man-Machine Studies, 11, 411-436.

Berry, D.C. (1987). The problem of implicit knowledge. Expert Systems, 4, 144-151.

Butters, N. (1984). Alcoholic Korsakoff's Syndrome: An Update. *Seminars in Neurology* 4, 2, 226 - 244.

Calvin, W.H., (1995). Cortical Columns, modules, and Hebbian cell assemblies. In M.A. Arbib, Ed., The Handbook of Brain Theory and Neural Networks. Cambridge, MA: Bradford Books/MIT Press.

Clark, A. (1992). The presence of a symbol. Connection Science, 4, 193-205.

Cohen, N.J., Eichenbaum, H., Deacedo, B.S., and Corkin, S. (1985). Different memory systems underlying acquisition of procedural and declarative knowledge. In D.S. Olton, E. Gamzu, and S. Corkin (Eds.), Memory dysfunction: an integration of animal and human research from preclinical and clinical perspectives , 54-71. New York: New York Academy of Sciences.

Coward, L. A. (1990). *Pattern Thinking*. New York: Praeger.

Coward, L. A. (1999). A physiologically based approach to consciousness, *New Ideas in Psychology*, 17 (3), 271-290.

Coward, L. A. (2000). A Functional Architecture Approach to Neural Systems. *International Journal of Systems Research and Information Systems*, 9, 69 - 120.

Coward, L. A. (2001). The Recommendation Architecture: lessons from the design of large scale electronic systems for cognitive science. *Journal of Cognitive Systems Research* 2(2), 111-156.

Dunn, J. (1998). Implicit Memory and Amnesia, in Kirsner, K., Speelman, C., Maybery, M., O'Brien-Malone, A., Andersen, M., and MacLeod, C. (Eds). Implicit and Explicit Mental Processes, 99-117. New Jersey: Erlbaum.

Forbach, G.B., Stanners, R.F., and Hochhaus, L. (1974). Repetition and practice effects in a lexical decision task. Memory and Cognition, 2, 337-339.

French, R.M. (1999). Catastrophic Forgetting in Connectionist Networks. *Trends in Cognitive Science* 3(4), 128-135.

Kamel, R. (1987). Effect of Modularity on System Evolution. *IEEE Software*, January 1987, 48 - 54

Kingsley, R. E. (2000). *Concise text of neuroscience*. Baltimore MA: Lippincott, Williams and Wilkins.

Kirsner, K. (1998). Implicit Memory, in Kirsner, K., Speelman, C., Maybery, M., O'Brien-Malone, A., Andersen, M., and MacLeod, C. (eds), Implicit and Explicit Mental Processes, 13-36, New Jersey: Erlbaum.

Mathews, R. Buss, R. Stanley, W. Blanchard-Fields, F. Cho, J. and Druhan, B. (1989). Role of implicit and explicit processes in learning from examples: a synergistic effect. Journal of Experimental Psychology: Learning, Memory and Cognition, 15, 1083-1100.

Nortel Networks (2001). *DMS-100/500 Feature Planning Guide* http://www.nortelnetworks.com/products/01/dms100w/doclib.html

Scarborough, D.L., Cortese, C., and Scarborough, H.S. (1977). Frequency and repetition effects in lexical memory. Journal of Experimental Psychology: Human Perception and Performance, 3, 1-17.

Schacter, D.L., Bowers, J., and Booker, J. (1989). Intention, awareness and implicit memory: The retrieval intentionality criterion. In S. Lewandowsky, K. Kirsner, and J. Dunn (Eds.). Implicit Memory: Theoretical Issues. Hillsdale NJ: Erlbaum.

Schacter, J.C. and Graf , P. (1986). Preserved learning in amnesic patients: Perspectives from research on direct priming. Journal of Clinical and Experimental Neuropsychology, 8, 727-743.

Shimamura, A. (1986). Priming effects in amnesia: Evidence for a dissociable memory function. Quarterly Journal of Experimental Psychology, 38A, 619-644.

Speelman, C. (1998). Implicit Expertise: Do We Expect Too Much from Our Experts. In Kirsner, K., Speelman, C., Maybery, M., O'Brien-Malone, A., Andersen, M., and MacLeod, C. (Eds.), Implicit and Explicit Mental Processes. New Jersey: Erlbaum.

Standing, L., Conexio, J., and Haber, R.N. (1970). Perception and Memory for Pictures: Single-trial learning of 2500 visual stimuli. *Psychonomic Science* 19(2), 73 - 74.

Tulving, E., Schacter, D.L., and Stark, H.A. (1982). Priming effects in word-fragment completion are independent of recognition memory. Journal of Experimental Psychology: Learning, Memory, and Cognition, 8, 4, 336-342.