

The Data Learning Problem Arises also in Cognitive Architectures other than Soar

Richard M Young (r.m.young@herts.ac.uk)
Department of Psychology, University of Hertfordshire
Hatfield, Herts AL10 9AB, UK

Abstract

The *data learning problem* is a phenomenon that arises when an agent employing a cognitive architecture faces the task of acquiring declarative information from an external source, such as the “answer” to a “question”. Because the agent has to pay attention to both question and answer in order to learn the association between them, it is problematic for the agent to learn to produce the answer in response to the question alone. This observation helps shape the basic characteristics of human memory. The problem was first reported with the Soar architecture, but it arises also in ACT-R, and this paper argues that it will occur in any cognitive architecture, connectionist as well as symbolic, which is specified in a sufficiently explicit manner to avoid having the theorist act as an implicit homunculus for the agent.

The Data Learning Problem

The *data learning problem* is a phenomenon that arises when an agent faces the task of acquiring declarative information provided by an external source. Paradigmatic tasks are, in the laboratory, paired associates, where in response to a stimulus word such as *shelf*, the agent has to learn to say the digit *eight*; or in everyday life, learning the correct answer to a question, such as that *the capital of France* is *Paris*. In such situations, both the question and the answer (or the “stimulus” and the “response”) must be present and attended to by the agent, in order to learn the connection between them. Thus the agent must process both the question “capital of France?” and the answer “Paris” if it is to acquire the relevant fact. It is therefore problematic for the agent to acquire a directed association from the question to the answer, such that it will be able to retrieve the answer in response to the question alone, without the answer also being present.

The data learning problem was first identified in work with the Soar cognitive architecture. However, this paper shows that the problem occurs with ACT-R as well, and argues that it arises in any architecture which is sufficiently theoretically explicit.

The Data Learning Problem in Soar

The data learning problem was first noticed and discussed (and termed the *data chunking problem*) by Rosenbloom, Laird & Newell (1987; Newell, 1990) in early work with Soar, and later substantially re-analysed by others (Vera, Lewis & Lerch, 1993; Young & Lewis, 1999). All these authors trace the problem back to the learning mechanism employed by Soar.

In Soar, all knowledge — both declarative and procedural — is encoded as *if* \Rightarrow *then* associations in production rules.

These production rules implement a search in a state space, by repeatedly identifying an operator to apply to the current state, thereby transforming it to a new state. Learning can occur only when Soar encounters a situation, an *impasse*, where it cannot proceed with its normal process of choosing operators and applying them to the current state. In the impasse, Soar sets up a new state, called a *substate*, and starts a new search whose purpose is find further information that resolves the impasse and allows the original search to continue. Soar learns a new production rule when it finds information that helps the resolution of the impasse and which it associates with the original state. For that new production rule, the ‘then’ part simply consists of the new information. The ‘if’ part consists of the information that existed before the impasse was encountered and to which the new information is connected by the sequence of production rule firings.

That description sounds a bit complicated, but the process is quite simple in concrete cases. It also has some surprising consequences. Consider a Soar agent learning about the capital of France. As analysts outside of the process, we can see that the production rule the agent needs to acquire is of the form:

Q: capital of France? \Rightarrow A: Paris (R1)

However, in the actual learning situation, the agent needs to be told both the question and the answer, otherwise it will have no way of knowing what the answer is. So the agent encounters an impasse because it doesn’t know the answer to the question *What is the capital of France?*, and that impasse is resolved when it is told that the answer is *Paris* because it can then respond “Paris”. But because both the question and the answer were consulted in giving that response, the production rule actually learned takes the form not of R1 but of:

Q: capital of France? & A: Paris \Rightarrow A: Paris (R2)

Rule R2 encodes the information “If you are given the question *What is the capital of France?* and the answer *Paris*, then respond ‘Paris’”. This is obviously not equivalent to the desired rule R1, because it is applicable only when it is given both the question and the answer. It therefore will not fire when given only the question, which is what R1 does and what the agent needs to be able to do if it is to know how to answer the question. How, then, can the agent ever acquire rule R1?

The solution to this problem (Rosenbloom *et al*, 1987) requires us to note that although rule R2 cannot serve to *recall* the answer to the question, it can serve as an *episodic recognition* rule encoding the fact that the question and

answer have been encountered together. Rosenbloom *et al* point out that if the agent is able to generate plausible candidates for the answer, then the recognition rule R2 can fire to pick out which one was previously presented as the answer. In this case the agent asked *What is the capital of France?* might generate and consider well-known cities in France, such as *Calais*, or *Lyon*, or *Marseilles*, or *Paris*. For the first three nothing would happen, but for the fourth, rule R2 would fire, identifying Paris as the answer and enabling the agent to respond. Because this time the answer is generated internally, the impasse occurs when only the question is present, not the answer, and so when *Paris* is given as the response the new rule learned is R1. Thereafter the agent can answer the question by recalling the answer, i.e. by firing rule R1, and the generation process does not have to be repeated.

Young & Lewis (1999) re-cast the process as a choice between four methods of answering the question:

1. If there is a production rule that provides the answer to the question, then it fires to deliver the answer.
2. If the answer already exists in working memory, then it gets picked up and delivered as the answer.
3. If possible candidates for the answer can be generated, and if a production rule exists that will recognise the answer, then candidates are generated until the answer is recognised.
4. The answer is sought from external sources.

These methods are listed in order of increasing effort. It is therefore rational for the agent to attempt the methods in that order. On the first pass, when the agent is first asked *What is the capital of France?*, none of the first three methods is applicable: the agent does not yet have a rule like R1; the answer is not available in working memory; and although the agent can try generating plausible candidates, there is no basis for recognising the correct answer. So the agent needs to be told (method 4) that the answer is *Paris*. It then has the answer in working memory, and so can respond (using method 2). The result is the acquisition of rule R2, the recognition rule.

On the second pass, when the agent is asked the question again, once more neither method 1 nor method 2 is applicable. Method 3 however does now work. As just described, the agent proceeds to generate plausible candidates, and when *Paris* is generated rule R2 fires, identifying it as the correct answer. In consequence, the recall rule R1 is acquired.

On the third and subsequent passes, once the agent is asked the question rule R1 fires (method 1) to give the answer. The agent has now clearly learned the geographical fact.

In their discussion, Young & Lewis (1999) stress several characteristics of this process:

- Learning the fact, i.e. acquiring the recall rule R1, requires the agent at one stage to generate the answer from its own internal, cognitive resources (Newell, 1990). This observation connects with wider

discussion of the “reconstructive” nature of human memory (e.g., Bahrck, 1970).

- The sequence of learning emerges as a by-product of the agent’s rational attempts to answer the question. At no point is there a deliberate attempt to learn, and there is no appeal to any mechanism for “self-programming”, i.e. deliberate learning.
- The learning process is inherently multi-pass, not because of any kind of gradual strengthening but because of the inherent informational dependencies in the process. The first pass is needed to acquire the recognition rule, which is used in the second pass to acquire the recall rule. Only in the third and later passes can recall occur.

The learning process just sketched arises as a consequence of the architecture, and is lean in its theoretical assumptions. Young & Lewis see these as theoretically desirable properties of any explanation of learning. In particular, the analysis argues against the need for any homunculus-like mechanism for deciding what should and what should not get learned.

Generality of the Data Learning Problem

Although the data learning problem has to date been documented and analysed only for Soar, our account of its origins suggests that the problem is potentially of wide generality and should arise also for other, or perhaps even all, cognitive architectures. After all, the only assumption needed to carry the argument is that learned memory associations incorporate the information presented — and that would seem to be a reasonable tenet of most architectures and associative theories.

So, is the data learning problem just a quirk of Soar, or is it instead pointing us to a deep truth about cognitive architectures in general? If we strip away the particularities of Soar, the bare bones of the argument look something like this:

- We assume that memory (or at least, part of memory) is encoded in the form of directed associations, *cue* \Rightarrow *response*.
- When learning a new association, the *response* derives from some response previously produced by the agent, and the *cue* reflects the information presented to the agent.
- When learning a new fact, such as a question-answer pair, the agent has to be presented with both the question and the answer, since by supposition this is a new fact that the agent does not already know.
- Therefore, the agent is likely to acquire an association

$$\text{cue \& response} \Rightarrow \text{response}$$
which does not serve to recall the *response* given just the *cue*.

If we apply this argument to the case of an agent learning that the capital of France is Paris, it follows that the agent will be presented with both the question *capital of France?*

and the answer *Paris*, and will learn a new association based on

Q: capital of France? & A: Paris ⇒ A: Paris.

This is the data learning problem. The solution to the problem appears to be that:

- By some means or other, the agent has to generate the *response* from its own cognitive resources. (The learned association just above may be of help in this process.)
- The effect of generating the response with only the cue presented will be to learn a new association based on just the cue, e.g.,

Q: capital of France? ⇒ A: Paris

which is exactly the association wanted.

Stated that way, the argument makes no appeal to features or mechanisms specific to Soar. Indeed, the assumptions would seem to be widely shared by most if not all associative theories and architectures, which means that they too should exhibit the data learning problem.

If the logic of this argument is correct, it raises a number of puzzling questions. Why has the problem been reported only for Soar? Does the problem not in fact arise for other theories and architectures, or has it simply not been noticed? If it does arise, why has it not been identified and discussed before now?

Such questions are perhaps more for historians and philosophers of science to answer than for working scientists, but we can go some way towards resolving the puzzle by considering the operation of the many theories that posit a direct learning of cue-response associations. It appears that the authors of such theories have no hesitation in assigning to the cue and the response different roles in the postulated learning mechanism, even when both are of the same kind (e.g., written words) and are simultaneously presented to the agent. For example, in network models (for more on which, see below), the theorist might treat the cue as the input to a network, while the response is treated as the target output for supervised training of the network. Even as careful a theorist as Estes (e.g., 1994), for example, seems implicitly to assume that in verbal paired-associate learning, subjects have enough deliberate control over their cognitive mechanism that they can treat the *cue* part of a paired associate in one way and the *response* part in another.

The plausibility of this assumption would intuitively seem to depend upon the nature of the items being associated. In the case of classical conditioning, say, the ‘response’ is definitely a *response*, i.e. something that the subject actually does (such as a dog’s salivating: Pavlov, 1927), and is quite distinct in kind from the perceptual stimulus used as the cue (in that case, the ringing of a bell). In such cases, it seems reasonable to suppose that the cue is treated by the cognitive machinery as the input to an associative network, with the response being the output. But at the other end of the spectrum, with verbal materials, the situation is quite different. There, in many studies, the cue and response are exactly the same kind of thing: written words or nonsense

syllables, where the desired ‘response’ is as much an input (presented to the subject) as an output. Indeed, the same item can appear as both a cue and a response, in the same or different lists. How plausible is it then that just because the experimenter tells the subject “When you see *this*, I want you to say *that*”, the subject’s cognitive system will treat the two items in such contrasting ways?

To take just one concrete illustration from the many possible, when presenting their “stimulus sampling theory”, Atkinson & Estes (1963) state that

“... most contemporary learning theories utilize a common conceptualization of the learning situation in terms of *stimulus*, *response*, and *reinforcement*. The stimulus term of this triumvirate refers to the environmental situation with respect to which behavior is being observed, the response term to the class of observable behaviors whose measurable properties change in some orderly fashion during learning, ...” (p.123)

However, in describing a paired-associate learning experiment, they give an example of a pair

| S | R |
|------------|--------------|
| <i>xvk</i> | <i>house</i> |

and say “On a reinforced trial the stimulus and response members were exposed together, as shown.” (p.126). So despite its earlier definition in terms of “observable behaviors”, the desired ‘response’ is presented as a perceptual input to the subject just as much as the ‘stimulus’ is.

The (implicit) assumption that the cue and response can be treated in contrasting ways by the agent’s cognitive mechanism makes sense in terms of task analysis, because they do indeed play asymmetric roles in the task: given the cue, the agent is supposed to recall the response. However, the assumption is much less plausible from the viewpoint of the cognitive architecture. It seems unlikely that two structurally similar perceptual stimuli can be treated by the cognitive mechanism in such structurally contrasting ways. From the perspective of a more complete computational account, such as is given by Soar or other integrated cognitive architectures, it becomes clear that the theorist — albeit unintentionally — is playing the role of a *deus ex machina* in relation to the mechanism postulated by the associative theories, making sure that the cue goes *here* and does *this*, and the response goes *there* and does *that* within the mechanism.

According to the viewpoint adopted in this present paper, this role being taken by the theorist serves to mask the data learning problem, leaving the theorist unaware of it. Thus, the answer to the puzzle is that, in principle, the data learning problem does indeed arise for other learning theories, and becomes evident once they are formulated sufficiently carefully and explicitly. However, in their usual formulations, the theorist plays a homunculus-like role by arranging for the cue to be handled by the theory in one way and the response in a different way. We should expect that the data learning problem will become apparent once the theories are reformulated with sufficient computational explicitness.

The rest of the present paper begins to explore the generality of the data learning problem. It first shows that the problem arises also in ACT-R, and then goes on to argue that it occurs in connectionist architectures too. The paper concludes by considering the implications for cognitive theory.

The Data Learning Problem in ACT-R

ACT-R (Anderson & Lebiere, 1998; Anderson, Bothell, Byrne, Douglass, Lebiere & Qin, in press) stores long-term knowledge in two different forms. Declarative Memory (DM) holds information in the form of simple relational units called *chunks*, while Production Memory holds knowledge in procedural form as production rules. By default, facts are stored as chunks in DM. However, retrieval of chunks from DM is a potentially slow and error-prone process. Consequently, when a task which includes the retrieval of information from DM has been much practised, the information becomes incorporated into production rules which then no longer have to access DM during fluent performance. This process is referred to as *knowledge compilation*.

New chunks in DM are acquired by, among other means, perceptual input from the environment. The acquisition of new production rules is a little more complicated (Taatgen & Lee, 2003). Basically, ACT-R examines each pair of production rules that fire consecutively, and tries to compose a new production by combining them. If the first production is of the form *condition1* \Rightarrow *action1* and the second one is *condition2* \Rightarrow *action2*, then the new composed production is roughly

{condition1 \cup condition2} – action1 \Rightarrow action2

where the ‘ \cup ’ indicates the union of the two conditions. In the case where *action1* requests an item to be retrieved from DM and *condition2* makes use of that item, the actual item retrieved is compiled into the new production rule, so that in future it avoids the need for the retrieval.

Interestingly, this mechanism implies that ACT-R is unable directly to acquire a production rule encoding information presented externally. In other words, it exhibits the data learning problem. Consider an ACT-R model given the information that Paris is the capital of France, and trying to acquire a rule like R1 encoding that fact. It cannot do so directly, because the composition mechanism just described gives a newly created production an action (*action2* as written above) which comes from an existing production, and initially the model will have no production whose action side is *Paris*.

The solution is for the model first to learn the information as a fact in DM. Then, on a later occasion, it can retrieve the fact and compile it into a production rule. The whole process of acquisition consists of three stages:

In the first stage, the ACT-R model is given the question *capital of France?* and the answer *Paris*. It attends to them and acquires a chunk in DM encoding the question-answer relation between them.

In the second stage, the model is given the question again (but not the answer), and it attempts to answer the question itself by using a pair of very general production rules representing a method along the lines of

IF you want to answer a question (Pa)
THEN try to retrieve from DM a fact about that question

IF you want to answer a question (Pb)
and you have retrieved from DM a fact relating that question to an answer

THEN give that answer

In this case the question is *capital of France?* and the answer retrieved is *Paris*. The consequence of firing production Pa followed by Pb is that the model acquires a new production rule corresponding to R1.

In the third stage, on later occasions when the model is presented with the question, the new production rule R1 fires and yields the answer directly.

This process has some noteworthy characteristics:

- Acquiring the recall rule R1 requires the agent to generate the answer from its own internal, cognitive resources, in this case by recall of the relevant chunk from DM.
- The sequence of learning emerges as a by-product of the agent’s attempts to answer the question. The agent’s behaviour is rational, in that the firing of rule R1 is a faster, less effortful, and more reliable method than continued reliance on the recall route. At no point is there a deliberate attempt to learn, and there is no appeal to any mechanism for “self-programming”, i.e. deliberate learning.
- The learning process is inherently multi-pass, because of the implicit informational dependencies in the process. The first pass is needed to acquire the declarative chunk, which is used in the second pass to acquire the direct production. Only in the third and later passes can the production be used.

These characteristics can fruitfully be compared with the corresponding properties of the process for Soar. (The astute reader will notice that the data learning problem relates mainly to the *condition* or *stimulus* side of the learned association in Soar, but to the *action* or *response* side in ACT-R.)

The Data Learning Problem in Connectionist Architectures

At first sight it seems implausible that connectionist models would suffer from the data learning problem, since such models are generally regarded as epitomising the mechanism needed for associating a given input with a specific output. However, closer examination raises doubts as to whether the usual story about associative learning and connectionism is indeed sufficient to enable networks to learn associations directly without encountering the data learning problem.

We consider first the connectionist architectures for supervised learning, such as simple pattern associators or multi-layer feedforward networks (see, e.g., McLeod, Plunkett & Rolls, 1998), trained by back-propagation or similar regimes. Such networks cannot be considered as providing counter-examples to the claimed generality of the data learning problem, for two reasons. First, as conventionally formulated, they require the cue and the response to be treated by the mechanism in quite different ways, with the cue being used as the input to a network and the response being used as the implicit target for training. We have already seen, as in the discussion of stimulus sampling theory above, that such different treatment of cue and response indicates that the theory is not fully computationally explicit and that the theorist is implicitly playing a homunculus-like role in the application of the theory. The second reason is the difficulty of mapping the supervised learning paradigm onto the cognitive reality of the kind of learning situations we are considering. If the supervisor is regarded as being external to the agent, then it simply is not the case that a source outside the agent is providing the information and feedback the numerous times needed by this kind of connectionist learning. If, on the other hand, the supervisor is regarded as being somehow internal to the agent, then it is even more apparent that the theory is postulating a homunculus inside the agent able to configure the input and output of the network appropriately and provide a suitable regime of reward and punishment.

More relevant to the concerns of the present paper are the various kinds of auto-associative and recurrent networks. Since the overt task for these networks is to learn to reproduce (or anticipate) their input, a knowledgeable supervisor is not needed. The natural way to apply such networks to the task of associating a response with a given cue is to treat the cue and the response together as a compound input and learn to map that input onto itself. Doing this forces the network to develop attractors, one for each cue-response pair, and we can then rely on the ability of such networks to “complete” partial inputs in order to reconstruct the pair given just the cue. (It should be noted that the ability of the network to auto-complete is closely analogous to ACT-R’s ability to retrieve a complete chunk of information from declarative memory, given just one of its components.)

Whether such auto-completion is feasible and effective for a particular set of cues and responses will depend in complicated ways on the encoding of the input, on the pattern of similarities among the cues and responses, and on the regularity or otherwise of the mapping between them. If we assume that the completion can be learned, then the network will have learned to retrieve the pair (e.g., {*capital of France?*, *Paris*}) given just the cue (*capital of France?*). If we really want a connectionist network that, given the cue, produces just the response, then the auto-associative network could be used as an internal supervisory mechanism to train a separate feedforward network — as is done, for example, in network models of how a fast-learning

hippocampus may function to internally train a slower-learning neocortical system (McClelland, McNaughton & O’Reilly, 1995).

If we try to summarise the situation for connectionist architectures in the same style as we have for Soar and Act-R, we get:

- A computationally explicit connectionist model is unable to learn directly the mapping from cue to response. (This is the data learning problem again.)
- An appropriate kind of auto-associative network can learn the pairs {cue, response}.
- The ability of the network to auto-complete enables it to retrieve a pair given just the cue.
- If necessary, the network with the learned pairs can be employed as the basis for an internal process of learning the direct mapping from cue to response.

The parallels with the case of the production rule architectures will be apparent to the reader.

Concluding Discussion

This paper has defined and explored the data learning problem and the way the problem manifests itself in a symbolic cognitive architecture, Soar; in a hybrid architecture, ACT-R; and in connectionist architectures. Despite their marked differences at the implementation level, the problem arises in a strikingly similar way in all three families of architecture. The paper further argues that the problem should in principle occur for all cognitive architectures.

The core of the phenomenon is that, when an agent has to learn the answer to a question — or in a laboratory setting, has to learn the *response* to a *cue* — then because both the question and the answer have to be presented to the agent in order for it to learn the pairing, the association that is acquired links both the question and the answer to what gets retrieved. In order for the agent to learn a ‘direct’ association, so that given (just) the question it can recall (just) the answer, the agent has to use the acquired pairing in order to generate the answer for itself as a basis for learning the direct association.

Although there has not been space in this paper to pursue the matter, the data learning problem exerts a deep-seated influence over the nature of human memory (Young & Lewis, 1999). It bears directly on the generative aspects of memory (e.g., Bahrick, 1970); it likely underlies the need for the agent’s active involvement in learning; and it perhaps extends to effects such as the importance of self-explanation in conceptual understanding (e.g., Chi, Deleew, Chiu & Lavancher, 1994).

If the phenomenon is as ubiquitous as is being claimed here, some account is needed of why it has not been noticed before. The explanation was traced back to the observation that in less computationally explicit theories, such as the many versions of associative learning theory on offer in psychology, the theorist implicitly plays the role of homunculus for the theory, deciding what aspects of the

perceived environment are to be regarded as cue and what as response, and the different ways in which they are handled. This unintentional involvement of the theorist in the working of the theory has served to mask the data learning problem until now. It is only when cognitive architectures are formulated explicitly in computational terms that the problem comes to light.

One of the advantages always claimed for computational cognitive modelling is the increased explicitness it forces on the theorist, and the way that it consequently banishes the homunculus from cognitive theorising. The present discussion suggests that the emergence of the data learning problem is a case in point.

References

- Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. & Qin, Y. (in press). An integrated theory of the mind. [Available from <http://ACT-R.psy.cmu.edu/papers/403/IntegratedTheory.pdf>]
- Anderson, J. R. & Lebiere, C. (1998) *The Atomic Components of Thought*. Erlbaum.
- Atkinson, R. C. & Estes, W. K. (1963) Stimulus sampling theory. In R. D. Luce, R. R. Bush & E. Galanter (Eds.), *Handbook of Mathematical Psychology*, vol 2, 121-268. New York: Wiley.
- Bahrick, H. P. (1970) Two-phase model for prompted recall. *Psychological Review*, 77, 215-222.
- Chi, M., Deleew, N., Chiu, M. & Lavancher, C. (1994) Eliciting self-explanations improves understanding. *Cognitive Science*, 18, 439-477.
- Estes, W. K. (1994) *Classification and Cognition*. Oxford University Press.
- McClelland, J. L., McNaughton, B. L. & O'Reilly, R. C. (1995) Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102 (3), 419-437.
- McLeod, P., Plunkett, K. & Rolls, E. T. (1998) *Introduction to Connectionist Modelling of Cognitive Processes*. Oxford University Press.
- Newell, A. (1990) *Unified Theories of Cognition*. Harvard University Press.
- Pavlov, I. P. (1927) *Conditioned Reflexes*. London: Oxford University Press.
- Rosenbloom, P. S., Laird, J. E. & Newell, A. (1987) Knowledge-level learning in Soar. *Proceedings of AAAI'87*, 499-504. Los Altos, CA: Morgan Kaufmann. [Reprinted in P. S. Rosenbloom, J. E. Laird & A. Newell (Eds), *The Soar Papers: Research on Integrated Intelligence*, vol 1, 527-532. MIT Press, 1993.]
- Taatgen, N. A. & Lee, F. J. (2003) Production compilation: A simple mechanism to model complex skill acquisition. *Human Factors*, 45 (1), 61-76.
- Vera, A. H., Lewis, R. L., Lerch, F. J. (1993) Situated decision-making and recognition-based learning: Applying symbolic theories to interactive tasks. *Proceedings of the 15th Annual Conference of the Cognitive Science Society*, 84-95. Boulder, Colorado.
- Young, R. M. & Lewis, R. L. (1999) The Soar cognitive architecture and human working memory. In A. Miyake & P. Shah (Eds), *Models of Working Memory: Mechanisms of Active Maintenance and Executive Control*, 224-256. Cambridge University Press.