

Compression, Dilation and the Redescriptive Role of Explicitation

Chris Thornton
Cognitive and Computing Sciences
University of Sussex
Brighton
BN1 9QH
UK

Email: Christopher.Thornton@firenet.uk.com
Tel: (44)1273 678856

May 21, 2003

1 Introduction

Traditionally, artificial intelligence (AI) work has stressed *representation* as a key concept for research. However, with some researchers beginning to adopt a more behaviourist outlook on cognition, there is an increasing need to justify this stress. Clark and Toribio (1) put forward a case which is of interest here. Their arguments stress the significance of representation and help to bring out the shortcomings of the anti-representationalist position. Their paper also expresses the beginnings of a *general* explanation for representation's role in cognition. In rough summary this states that representation may be a vehicle for certain types of behaviour-facilitating input-space transformation. As they put it

... behavioural success often depends on the ability to *compress* or *dilate* an input space. That is, the cognizer must be able to treat inputs whose immediate codings (at the sensory peripheries) are quite similar as deeply different (dilation) and conversely, to treat inputs whose immediate codings are quite different as deeply similar (compression). On the modest reading which we recommend, internal states developed to serve this end just *are* internal representations whose contents concern the states of affairs thus isolated. Such internal states function as feature detectors and enable the system to differentially respond to situations for which it has no dedicated transducer-level

detection mechanisms. (1, p. 32).

The provision of a technical interpretation for the terms ‘compress’ and ‘dilate’ is not a goal of Clark and Toribio’s paper. However, the present paper shows that such an interpretation can be mapped out. This casts compression and dilation as basic operations within a redescriptive framework, similar to the one envisaged by Karmiloff-Smith [2]. To develop this interpretation and show its connection to redescription, I will first present a *task analysis* for behaviour acquisition. This will show that the behaviour-acquisition task has two forms, one of which is considerably more challenging than the other. The challenging form will then be shown to be tractable to representation-construction processes which fit the ‘compression’ and ‘dilation’ prototypes.

2 Task analysis of behaviour acquisition

The learner of a behaviour necessarily receives some form of **inducement** regarding the behaviour which is to be learned. This inducement most commonly takes the form of a training set of example input/output pairs (i.e., a set of examples showing what output or response the agent should produce for a given input). However, even where it takes some other form (e.g., in reinforcement learning) we can still regard it as having been *derived* from such a training set. Whatever the inducement provided, it must necessarily differentiate between actions that are appropriate from actions that are inappropriate for the relevant behaviour. Thus we can always imagine that it has been derived from an explicit listing of appropriate actions, i.e., a training set of examples.

The aim in learning from examples is (1) to absorb the examples and (2) to be able to generalize from those examples to unseen cases. The first task is trivially accomplished if the learner has an adequate memory resource. So it is the second task on which attention usually focuses. Generalization involves ‘guessing’ the output for an unseen input. A good guess is a *justified* guess. The only source of justification for guesses is the training set of examples. Thus a good guess is a guess that is justified by the contents of the training set. We can analyze the ways in which generalization can be accomplished, then, by looking at how output guesses can be justified by a training set of examples.

Let us define x as the set of input-variable instantiations making up a particular input. Imagine a learner guesses that y is the correct output for x . Imagine also, for the sake of argument (and without loss of generality), that it makes this guess with absolute certainty, i.e., that it states that the probability of seeing output y given input x is 1. If the guess is justified the contents of the training set must somehow imply that $P(y|x)=1$. If the guess really *is* a guess the training set cannot contain the pair that specifies y as the correct output for x . Thus the probability $P(y|x)=1$ cannot itself be observed in the training set. The only way, then, that $P(y|x)=1$ can be justified is if x has properties upon which y — where it appears in the training set — appears to be conditional.

These properties may be explicit or implicit. The explicit properties of x are obviously the input variable instantiations which it contains. Thus, if x has explicit properties associating it with y , there must be some subset of x upon which y appears to be conditional. If x has implicit properties associating it with y then there must be some *evaluation* of those properties upon which y appears to be conditional. In other words, it must be the case that y appears to be conditional on a particular value of a function which evaluates the properties. We can summarize this formally by saying that the guess $P(y-x)=1$ may have two different forms of justification, namely

explicit justification because $P(y-x')=1$ is observed in the training set, and x' is a subset of x ¹ or

implicit justification because $g(x)=z$, and because $P(y-g(?x)=z)=1$ is observed in the training set. Here, $?x$ is any input from the training set and g is the function which evaluates the property.

This dichotomy between explicit and implicit justification turns out to be fundamental. To learn from a training set of examples, it is obviously necessary to exploit justifications for making particular output guesses. But doing so means ‘finding’ a particular probability value in a given distribution. Thus, the hardness of exploiting a particular justification must be related to the size of the relevant distribution. Since the target mapping is finite, the probability distribution for explicit justifications (i.e., for $P(y-x')$) is also finite: there are a finite number of inputs in the target mapping and therefore a finite number of subsets of those inputs. But the distribution for implicit justifications ($P(y-g(?x)=z)$) is *infinite* due to the infinite range of choices to be made with regard to the function g .²

This leads us straight to an important insight concerning the hardness of learning problems. Problems which involve exploiting explicit justifications involve sampling a finite distribution, while problems which involve exploiting implicit justifications involve sampling an infinite distribution. Other things being equal, then, the task of exploiting explicit justifications has a *lower* theoretical hardness than the task of exploiting implicit justifications.

The g function used above is, of course, what measures the implicit property which underpins the justification. No assumptions are made about the nature of the property; so we assume that g may be *any* computable function. But consider the case where g simply tests for the presence in x of certain absolute values. Then the implicit justification reduces to an explicit justification since we can restate it in terms of an ordinary probability conditional upon the relevant values. (If g tests for the presence of a *set* of states, then the justification can be rewritten in terms of a *set* of explicit justifications.) The value that is computed by the g in an implicit justification, then, cannot depend on absolute values of the input. It must depend on relational values. In other words, g must compute a *relationship*.

¹Note that x' may be empty. The probability is then effectively unconditional.

²It is well established that there are an infinite number of computable functions.

These observations allow us to add useful labels to the two classes of learning problem we have identified. The class of hard learning problems, which involve exploiting implicit justifications, can be called **relational** since solving them involves finding a g , or a **recoding function** as I call it, which computes a relational property of the input-variable instantiations. The class of easy problems, on the other hand, can be called **statistical** since exploiting an explicit justification involves sampling for a statistical effect in the form of an observed probability.

The task analysis, then, offers us a fundamental distinction between two variants of the learning problem: an easy variant which involves the exploitation of **statistical effects** (i.e., explicit justifications) and a hard variant which involves the exploitation of **relational effects** (i.e., implicit justifications). This distinction is, in fact, well known to Machine Learning researchers in the form of the heuristic which states that ‘learning relationships is hard’ [3]. In the terms of computational learning theory [4] the distinction between relational and statistical problems is a representation-independent hardness result.

3 Statistical learning yields knee-jerk behaviours only

The task analysis tells us that statistical acquisition (i.e., acquisition using purely statistical justifications) is easier in general than relational acquisition. We should not be too surprised, then, if we discover that the majority of behaviour acquisition methods are biased towards the easier, statistical form of the task. But relying on statistical acquisition methods for acquisition *in general* is impractical. For a behaviour to be statistically learnable, it must be the case that each behavioural response is firmly grounded in the input statistics. In effect, each response must be associated with *particular* values of *particular* input variables. This means that if a robot is to be constructed which can acquire a particular behaviour (based on inputs from a given sensor array), each response in that behaviour must be associated with particular values of particular sensors. In effect, the robot has to have sensors which test for each behavioural contingency *explicitly*.

It is easy to see just how restrictive this can be. Imagine that we want to build a robot that uses a visual system to perform some sort of assembly task in a dynamic environment. One of the behavioural responses in the task is the flexing of a joint in the robot’s left gripper. For the robot to acquire the relevant behaviour using statistical methods it must be the case that there is a (statistically) clear sensory ‘trigger’ for the flexing action. In effect, the robot must have some close approximation of a ‘time-to-flex-joint-in-left-gripper’ sensor. For complex behaviours this approach leads to an unacceptable proliferation of over-specialized sensors.

4 Compression and Dilation

In the sections below I will deal further with the problem of relational acquisition. However, it is worth pausing at this point to note that, using the task analysis provided, we can already re-interpret Clark and Toribio’s notions of input-space transformation. They note that cognizers must be able to ‘treat inputs whose immediate codings are quite different as deeply similar’ and we can now see that this requirement is guaranteed to arise in *any* relational acquisition task. In a relational task, statistical (i.e., similarity) effects are channeled via the unknown function g , which measures a *relational* property of its inputs. If the solution to the acquisition problems involves exploiting a relational effect then there cannot be any statistical solution which implies that there cannot be any isomorphism between input-level similarities and output-level (i.e., behaviour level) similarities. Thus, input-space transformation (compression/dilation) is necessitated just in case relational acquisition is necessitated and vice versa.

Compression and dilation, then, can be seen as variants of relational acquisition. Compression is required in the case where behaviour-level similarities have no isomorphism with input-level similarities. Dilation is required if input-level similarities have no isomorphism with behaviour-level similarities. Intuition would lead us to expect that compression is the more common variant and dilation — which presupposes the existence of spurious, misleading input-level similarities — is the special case. However, it is not hard to imagine realistic scenarios in which dilation would be necessitated.

Imagine a cognizer using a visual system to track a disk-shaped object flying through the air. The visual system is low-resolution and provides the cognizer with what are, in effect, proximity measures for the left and right edges of the disk. When these measures have a certain relationship the disk is spinning directly towards the cognizer necessitating rapid evasive action. However, almost identical sensory situations are produced for the opposite behavioural situation, i.e., ‘no action required’. Acquiring the disk-avoidance behaviour, then, involves capturing a relational effect. This, in turn, involves a ‘dilation’ of the input space which effectively filters out the misleading sensory-level similarities.

5 Exploitation of step effects

Equating compression/dilation with relational acquisition gives us a more detailed understanding of the problem which these processes aim to address. But the question of how input-space transformations might be implemented remains unanswered. As noted above, relational acquisition involves the identification of an unknown function and thus has — in the worst case — infinite complexity. However, it turns out that there is a way of approximating the task which effectively gets around the apparent necessity of finding a single computational ‘needle’ in the haystack of all computational functions.

The method is based on the observation that, in any realistic behavioural scenario, relational and statistical effects are unlikely to be *disjoint*. For exam-

ple, consider a simple object recognition task which involves detecting four red dots arranged in a rectangular configuration. The underlying effect here has both a statistical aspect (the objects of interest are always *red*) and a relational aspect (the dots are arranged in a rectangular configuration). A statistical acquisition process might easily capture the statistical aspect of the task (i.e., the significance of redness). But in doing so it will automatically acquire a sample of rectangles.

The key idea here is that these examples can serve, in combination, as a prototypical definition of rectangularity. Operationalizing the definition simply involves introducing machinery able to examine an arbitrary input for similarity to one of the examples. By capturing a statistical effect, then, an acquisition mechanism can obtain access to an accompanying relational effect without resorting to any search whatsoever.

The method can deal with relational *measures* as well as relational tests. For example, imagine that the acquisition task involves estimating the *size* of the rectangular configuration of dots. The examples netted via our statistical acquisition method will now vary in size. If we arrange them into a linear ordering according to their internal similarity, we obtain a set of control points for a virtual size measure. When we compare an arbitrary input against these control points, we effectively discover how far ‘along’ the size dimension the input falls. Thus, simply by returning the linear position of the best-matching control-point we obtain (an approximation of) the size of the original rectangle.

The advantages of accessing a relational affect via a statistical effect are best realised by allowing each relational test or measure to instantiate an invented variable. The values of these invented variables correspond to relational states of affairs among the original input variables. From the perspective of the acquisition mechanism, they are literally symbols representing those states of affairs but, as symbols, they stand for objects at a new, higher level of description.

Once we have a set of such invented variables we can recode our original inputs in terms of the invented variables. Continuing with the rectangles example we might imagine a further acquisition phase producing a higher-level, invented variable measuring size variation. Certain values of this variable would stand for *increases* in apparent size and these would serve as symbols representing ‘looming rectangular objects’. Such objects are at a higher level of description than the ‘dot’ level used in the original inputs. So the variable-invention process, which relies on the accessing of a relational affect via a statistical effect, effectively provides a step-up to a new level of description. For this reason, I call statistical effects which have a relational accompaniment ‘step effects’.

6 Explicitation

With the brakes fully off, the process described above has the ability to cycle repeatedly, identifying step-effects, producing associated invented variables, recoding inputs, sampling those inputs for step effects and so on. (The cyclical nature of the process is illustrated in Figure 1.) Unless external factors intervene

we would expect the looping to continue for as long as the original input stream, or any derived recoding of that input stream, provides any evidence of step effects. By the time it terminates, it can be expected to have reified objects and relationships throughout a multi-layered structure of levels of description. The net effect is that many implicit properties of the original input (i.e., sensory) environment have been turned into explicitly represented properties.

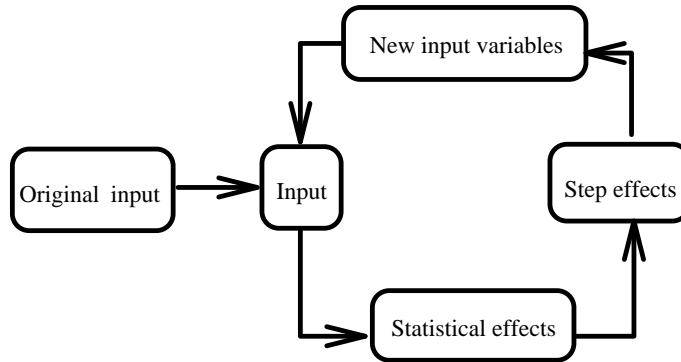


Figure 1:

7 Computational implementation

A working computational model of the process described — called ‘explicitation’ — has been implemented. Full details are provided elsewhere [5]. However, the general flavour of the method is that of a constructive, connectionist regime, such as cascade-correlation [6] or Upstart [7]. The process starts by initializing a network containing an appropriate number of input nodes. It then incrementally adds internal nodes to this net and, using a regime similar to competitive learning, tunes their weights so that each one comes to operate as a sensor for a particular statistical effect. This tuning process associates a significance value with each weight and modifies it according to how well the weight value predicts its input when the weight’s host node has the strongest response. This ensures that nodes are able to tune-in to statistical effects of any order exhibited by the current training inputs.³

The node construction process continues until each input produces a strong response in *one* internal node, i.e., until the network has a node for each statistical effect exhibited by the inputs. The input weights of the internal units are then checked for commonalities (i.e., nodes which respond to patterns over the *same* set of variables) since it is these situations which are indicative of the

³The process is, in fact, biased towards the identification of higher-order statistical effects in order to prevent nodes being created for effects which are actually components of effects *already* accounted for.

presence of a step effect.⁴ New variables are created as appropriate and the process is repeated with the next set of training inputs. Following each new presentation of inputs, the statistical learning process is applied at every level of the network, i.e., to the primitive inputs and to all derived input sets (all sets produced via instantiations of internal variables). Thus the network is able to extend vertically adding layer upon layer of internal variables and nodes.

8 Learning to discriminate ‘V’ shapes

The experimental implementation has been used to acquire the behaviour of discriminating ‘V’ shapes, i.e., configurations of four dots which subdivide into *two* oblique lines which join at the bottom. The general procedure was as follows. An initial input environment was presented which, via one iteration of step-effect identification, produced invented variables representing respectively position-of-left-sloping-oblique-line and position-of-right-sloping-oblique-line. A new input environment was then presented to the acquisition mechanism which effectively presented positive and negative examples of ‘V’ shapes.

Positive examples produced recordings at the first level of invented variables of similar ‘position’ values while negative examples did not. A further round of step-effect identification produced a variable representing ‘same-position-of...’ and this effectively served as a fully operational, discriminative representation of ‘V’ shapes. Note that by varying the input environment in this experiment, the acquisition mechanism was driven along a kind of cognitive trajectory, i.e., it was caused to produce explicit representations for particular implicit properties of the environment.

A graphical representation of a network that was generated in the experiment described is shown in Figure 2. At the bottom of the figure we have a row of five boxes. These represent the primitive variables of the net — two for each pair of end-points and one for the target output. The size of the shaded portion in each box represents the absolute value of the variable at the point when the figure was constructed. Two numeric variables were used to represent the heights of the two end points of the left oblique and two to represent the end-points of the right oblique. The final input variable was used to provide the training stimulus (i.e., target output) in the latter part of the training sequence.

Above the input variables we have a row of circles and these are connected by lines to the primitive input variables (lower boxes). The circles represent the first level of nodes in the network. The lines represent their input connections from the input nodes (variables). Only those connections with a high significance value are shown. Thanks to the acquisition regime, the first few nodes have all been tuned to statistical effects related to either the left oblique (first two input variables) or the right oblique (second two input variables). The shading of the circles simply represents the activation levels of the relevant nodes at the point when the figure was generated.

⁴This process is actually effected via a re-application of the basic statistical learning process.

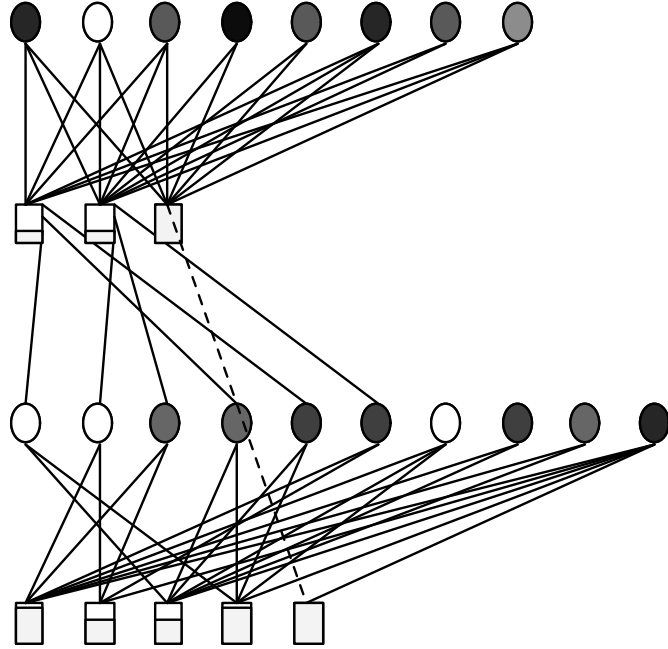


Figure 2:

In the first level of nodes, the first, fourth and fifth nodes were discovered to have captured patterns over the same set of input variables and were thus assumed to have identified a step effect. They were then used to create the first internal variable (first box in the third row up). Note how the intersection of each connection line from node to derived variable effectively shows which value of the variable is instantiated when the relevant node is most strongly active. At the top of the figure we see the nodes that have been created as a result of statistical operations applied to the derived inputs.

The network creation sequence is shown in Figure 3. Networks 1, 2, 3 and 4 are the networks created as a result of the initial phase of node creation. Networks 5 and 6 result from the exploitation of step effects, the creation of derived variables and the representation of the current training inputs. Networks 7 and 8 result from statistical operations in the lowermost network applied to the next phase of primitive inputs (the ones which exemplify the distinction between Vs and non-Vs). Network 9 results from statistical operations in the derived space applied to the relevant set of derived inputs.

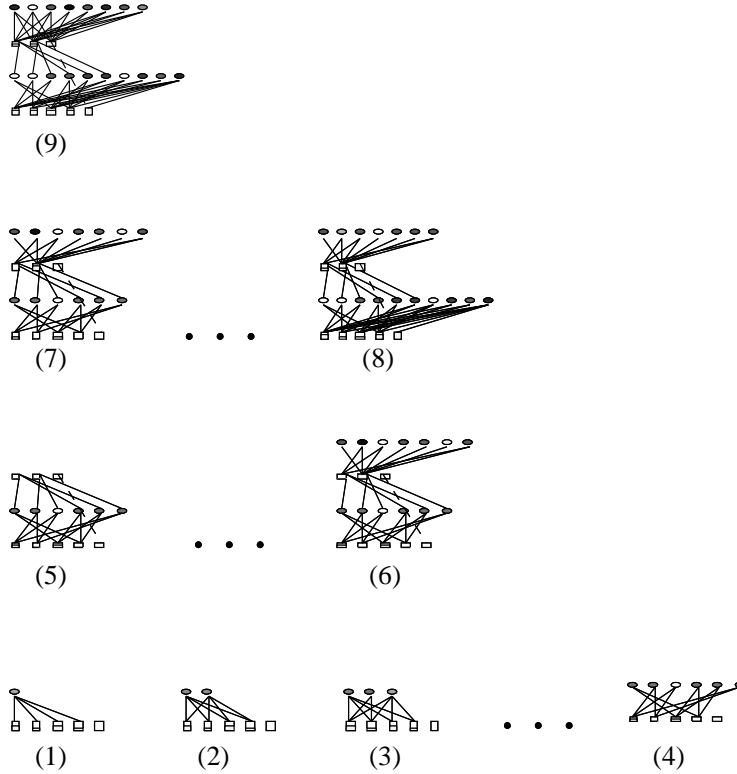


Figure 3:

9 Concluding comments

The explicitation model is interesting for a number of reasons. First, it provides a principled, experimentally viable approach to the notorious problem of relational behaviour acquisition (relational learning). Second, and more importantly for present purposes, it provides a computational model for the input-space transformation processes envisaged by Clark and Toribio. An added attraction of the model is that it engages in a fairly obvious way with Karmiloff-Smith's notion of *representational redescription* [8] which played an important role in some of Clark's earlier work on representation and which clearly underpins, to some degree, the concepts of compression and dilation.

In the explicitation model, the original input environment is iteratively re-coded ('redescribed') in terms of objects and processes at increasing levels of abstraction ('description'). The variables produced within the process literally stand for ('represent') these objects and processes within the dynamics of the explicitation procedure. Thus, on purely terminological grounds, explicitation would appear to provide an adequate model for both input-space transformation

and representational redescription.

However, there are some worries to be confronted. In particular, Karmiloff-Smith's notion of representational redescription places considerable emphasis on the notion that redescription involves a recoding from procedural to declarative representations. This emphasis is not in any sense a feature of the explicitation model. In fact there does not appear to be any obvious way to interpret the concepts of procedural/declarative representation within the the explicitation model.

Another worry involves the 'approximating' character of the relational acquisition process. In explicitation, relational effects are captured by what are, in effect, sparse lookup tables. In some cases these tables can be given a linear ordering based on internal similarity but this does not change their fundamental character. Clearly, such tables can only be regarded as providing a very imperfect definition of a relation. The imprecision of the definition can be expected to increase with the sparseness of the relevant table. But finessing this problem by making the tables very *large* is rarely an option due to resource constraints applying to the acquisition mechanism.

A further worry concerns the limited experimental testing that has been carried out to date. One positive result does not in any sense prove that the model is a robust, general method for relational acquisition let alone representational redescription. However, given the general lack of computation-level interpretation for the representation-level theories of Clark, Toribio and Karmiloff-Smith, the development of even a special-case model is certainly a step-up in the right direction.

References

- [1] Clark, A. and Toribio, J. (1994). Doing without representing?. CSRP 310, Cognitive and Computing Sciences, University of Sussex, UK.
- [2] Karmiloff-Smith, A. (1992). *Beyond modularity: a developmental perspective on Cognitive Science*. Cambridge, Ma.: MIT Press/Bradford books.
- [3] Dietterich, T., London, B., Clarkson, K. and Dromey, G. (1982). Learning and inductive inference. In P. Cohen and E. Feigenbaum (Eds.), *The Handbook of Artificial Intelligence: Vol III*. Los Altos: Kaufmann.
- [4] Kearns, M. (1990). *The Computational Complexity of Machine Learning*. The MIT Press.
- [5] Thornton, C. (1994). Knowing where to go without knowing where that is. CSRP 361, Cognitive and Computing Sciences, University of Sussex, UK.
- [6] Fahlman, S. and Lebiere, C. (1990). The cascade-correlation learning architecture. In D.S. Touretzky (Ed.), *Advances in Neural Information Processing Systems 2* (pp. 524-532.). Morgan Kaufmann Publishers, Los Altos CA.

- [7] Frean, M. (1989). *The Upstart Algorithm: A Method for Constructing and Training Feed-Forward Neural Networks*. Edinburgh Physics Preprint 89/479, Dept. of Physics, University of Edinburgh.
- [8] Clark, A. and Karmiloff-Smith, A. (1991). *The Cognizer's Innards: a psychological and philosophical perspective on the development of thought*. Brighton: University of Sussex (Price: 2.00).