Trading Spaces: Computation, Representation and the Limits of Learning*

Andy Clark and Chris Thornton†

Cognitive and Computing Sciences

University of Sussex

Brighton

BN1 9QH

U.K.

E-mail: Andy.Clark@cogs.sussex.ac.uk

Chris.Thornton@cogs.sussex.ac.uk

June 15, 199

*Research on this paper was partly supported by a Senior Research Leave fellowship granted by the Joint Council (SERC/MRC/ESRC) Cognitive Science Human Computer Interaction Initiative to one of the authors (Clark). Thanks to the Initiative for that support.

Abstract

The increasing level of attention being given to incremental learning is, we argue, fully justified. Although some view the process as something akin to an 'efficiency hack', we argue that it is, in fact, a key cognitive process. The argument is based on a statistical observation. Learning, for most purposes, is all about acquiring the ability to generate appropriate outputs from given inputs, i.e., it is all about acquiring the ability to predict — or, in general, give a probability to — specific bindings of output variables. Such predictions can be justified in two quite different ways by available training data (i.e., input/output examples). They can be justified *directly*, in virtue of being in 1-to-1 correspondence with probabilities (i.e., frequencies) directly observed in the training data. Or they can be justified *indirectly*, in virtue of being in 1-to-1 correspondence with probabilities observed in some re-coding of the data. Thus, where learning is driven by supplied training data, it must exploit some combination of these two types of justification.

Since the space of *indirectly* observed probabilities is grounded in the space of possible re-codings (i.e., applicable Turing machines), searching through it is intractable. But we should not infer from this that learning does not (and cannot) make use of indirectly observed probabilities. As we show, learning problems whose solution necessarily entails exploiting such probabilities (or *type-2* problems, as we call them) seem to be the norm in many realistic learning scenarios. We are thus left in need of an

[†]The order of names is arbitrary.

as adaptations

the data. Thus any process for regularity-extraction must attend to two different sources. The search space for regularities of the directly-observed variety is large but not, in general, intractably large. The search space for regularities of the indirectly-observed variety, on the other hand, is *infinitely* large, since it is grounded in the space of possible re-codings of the input data, i.e. the space of applicable Turing machines. Thus, on complexity grounds, we classify one form of regularity as being relatively 'transparent' and the other as 'opaque'.

When confronted with problems that involve extracting regularities of the opaque variety, connectionist learning algorithms such as standard backpropagation are unreliable. Parity generalization problems provide the obvious example. If one presents standard backpropagation (Rumelhart, Hinton and Williams, 1986b) with a complete parity mapping — allowing the algorithm to make use of a sufficiently rich internal architecture — then the learning will almost certainly succeed in achieving perfect performance. However, if one presents only a plausible training set (i.e., some large proportion of the complete mapping) then success is not assured. For example, if one presents 12 of the possible 16 cases from the 3-bit parity mapping, backpropagation typically learns those 12 cases rather rapidly and rather well (see Section 1.4 below). But it usually learns them in such a way that the four unseen cases are not handled correctly; i.e. it fails to generalize to the full mapping. The reason for this is fairly straightforward. With parity mappings, the significant probability effects are tied not

0.6398hizeit—fairly-1500bt

for all combinations of explicit values are necessarily at chance levels.) This means (for reasons that are explained in detail below) that solving the learning problem involves deriving an appropriate re-coding; i.e., it means extracting regularity of the opaque variety. The problem cannot be solved by extracting the transparent form of regularity — for the simple reason that it is not present.

The fact that connectionist learning algorithms (such as backpropagation) may fail on parity generalization problems shows that their capacities for extracting regularities of the less accessible form are limited to some degree. (Note that the problem here is *all* to do with the ability of such systems to learn *generalizable* solutions: the fact that a non-generalizable solution can be *learned*, or that a generalizable solution can be *represented* is not what's at issue.) Possible responses to the problem are (1) to blame connectionist learning algorithms and hope to find something better or (2) to deny that most of the problems solved by the human brain require the sort of re-coding that opaque regularity-extraction may require (i.e., to claim that parity-generalization is a pathological case). We reject (1) on the grounds that the problem is not, as far as we can tell, a mere artifact of connectionist learning; instead it is a problem which will arise for any learning algorithm which operates without specific inbuilt knowledge about a target domain. And we reject (2) because the problem of re-coding seems to arise even in relation to apparently simple 'robotics-style' tasks (i.e. in manifestly non-pathological cases — see Section 1.6 below).

The *correct* response to the problem, we argue, is to reconsider the difficulties in the light of an accumulating body of work, e.g., (Elman, 1991) which highlights the role of representational trajectories in connectionist learning. A representational trajectory is just a connected path through a temporally or spatially extended sequence of different learning tasks. By following such a trajectory a single intractable problem can be reduced to an incremental sequence of different and individually tractable problems. The solution to each sub-problem yields, in effect, a re-coded representational base. Such re-codings progressively reduce the complexity of the task of learning the target mapping until it becomes tractable. Otherwise put, each re-coding allows us to avoid a quantity of computational search by providing a more suggestive set of 'virtual inputs'. We thus trade computation off against representation.

While this strategy is not unfamiliar (it is just a kind of 'temporal homuncularism' and has been the subject of a well-known investigation by Elman, 1991), es1k,,bu

derstood in these terms. Such ploys and mechanisms range from simple evolved filters and feature-detectors all the way to complex cases of analogical reasoning. The goal, in every case, is to systematically re-configure a body of input data so that computationally primitive learning routines can find some target mapping, i.e. to trade representation against computation. A topical moral is that it is virtually impossible, in such cases, to avoid talk of internal representations as the products of each stage of re-configuring. Pace the more radical apologists of Artificial Life, even

regularities, consist in frequency effects found via some re-coding of the original data.

1 Statistical properties of training sets

Let us begin by making some general observations about the statistical properties of training sets (i.e. sets of training examples). Consider the training set shown below. This is based on two input variables (x1 and x2) and one output variable (x3). There are six training pairs in all. The pairs are laid out with one pair per line. An arrow separates the 'input vector' of the pair from the 'output vector'. The values of the two input variables appear on the left of the arrow. The value of the output variable appears on the right.

```
x1  x2       x3


1   2  -->   1

2   2  -->   0

3   2  -->   1

3   1  -->   0

2   1  -->   1

1   1  -->   0
```

In this training set we can observe a number of instantiation n-tuples, henceforth

instantiation 2-tuples. An example is <x1=3, x2=1>. This case is observed in the fourth line of the training set. A second-order case from the second line of the training set is <x3=0, x1=2>. Since there are only three variables in all there is exactly one third-order case for each member of the training set.

Given a particular case, we can compute the frequency (i.e., probability) with which it appears in the training set. The frequencies for all first and second-order cases in the training data above are shown in Table 1. Note that the frequencies for the third-order cases (i.e., the cases that specify values for all three variables) are degenerate. Assuming there is no duplication in the training data, each third-order case occurs exactly once. Thus its frequency is necessarily 1/n where n is the size of the training set.

1.1 Conditional frequencies

The frequencies shown in Table 1 are *unconditional* frequencies. We can also derive *conditional frequencies*.[2] These are frequencies that exist with respect to a particular constraint over variable instantiations. In Table 2 we see the frequencies for particular instantiations of the output variable (x3) given possible constraints on other variables. The column headed 'Fr' shows the absolute frequencies for the constraints. The column headed 'Fr: x3=0' shows the conditional frequency with which x3=0 when the relevant constraint applies. The

[2] These can be construed as Bayesian probabilities (Duda and Hart, 1973).

Case	Fr
	1
x2=2	0.5
x2=1	0.5
x3=1	0.5
x3=0	0.5
x1=3	0.33
x1=2	0.33
x1=1	0.33
x2=2, x3=1	0.33
x2=1, x3=0	0.33
x1=3, x2=2	0.17
x1=2, x2=2	0.17
x1=1, x2=2	0.17
x1=3, x2=1	0.17
x1=3, x3=1	0.17
x1=2, x2=1	0.17
x1=2, x3=1	0.17
x2=1, x3=1	0.17
x1=3, x3=0	0.17
x1=1, x3=1	0.17

Table 1:

column headed 'Fr: x3=1' does the same thing for the case x3=1.

By the argument used previously, the 2nd-order conditional frequencies here are of no interest since there is necessarily exactly one occurrence of each 2nd-order case of the constrained variables.

1.2 Type-1 versus type-2 frequencies

A clear distinction must be made between cases (such as those considered above) that can be observed *directly* in the training data, and cases that can only be observed *indirectly*. For our purposes, a case can be observed indirectly if it can be observed directly in some systematic *re-coding* of the original data. What this means is that an instantiation n-tuple that occurs in some re-coding of the original data, is considered to be a case that is 'observed indirectly' in the *original* data. We will call frequencies for directly observed cases type-1 frequencies. We will call frequencies for indirectly observed cases type-2 or *derived* frequencies.

The difference between the two types of frequency can be illustrated by re-coding our original training set. Imagine that we re-code the inputs (from above) by substituting — in each training pair — the original input variables with a single variable whose value is just the difference between the original variables. This gives us a set of derived pairs as shown in Figure 1 (the value of x4 here is the difference between the values of x1 and x2). The frequencies we *directly* observe

in this derived training set are type-2 (derived) frequencies with respect to the original training data. However, they are still frequencies.

Constraint	Fr
	1
x3=0	0.5
x3=1	0.5
x4=1	0.5
x4=0	0.33
x4=2	0.17
x4=1 + x3=1	0.5
x4=0 + x3=0	0.33
x4=2 + x3=0	0.17

Table 3:

Constraint	Fr	Fr: x3=0	Fr: x3=1
	1	0.5	0.5
x4=0	0.33	1.0	0.0
x4=2	0.17	1.0	0.0
x4=1	0.5	0.0	1.0

Table 4:

surface. Once this has happened it is a straightforward matter for a learning algorithm to exploit it. Recognizing the strong, mutual interdependence between learning and regularity leads us to distinguish three classes of learning problem.

- **Pure type-1 learning problems**: problems that involve exploiting type-1 regularities only,
- **Pure type-2 learning problems**: problems that involve exploiting type-2 regularities only,[4] and
- **Hybrid problems**: problems that involve exploiting some mixture of both types.

1.4 Parity problems are pure type-2

The distinction between type-1 and type-2 problems is nicely illustrated by the parity problems (cf. Rumelhart, Hinton and Williams, 1986a). Complete parity mappings show no type-1 regularity at all. Their observed frequencies are always *exactly* at their chance levels. (Hinton and Sejnowski, 1986) The input/output rule for a parity mapping is simply that the output should be 1 (or true) just

[4] A special case of the type-2 problem occurs when the relevant re-coding can be performed by deriving simple probability effects from *subsets* of the training data. The complexity of this variant exceeds that of the type-1 problem due to the additional cost of exploring possible partitions of the training data. However, it is not as great as that for the unrestricted type-2 case since there is no necessity to explore any part of Turing-machine space.

in case the input vector contains an odd number of 1s (or, in general, an odd number of odd values). The complete mapping for the third-order, binary-valued parity problem (i.e., 3-bit parity) is as follows.

x1	x2	x3		x4
1	1	1	-->	1
1	1	0	-->	0
1	0	1	-->	0
1	0	0	-->	1
0	1	1	-->	0
0	1	0	-->	1
0	0	1	-->	1
0	0	0	-->	0

Every single first and second-order conditional frequency for this mapping (for values of the output variable x4) is at its chance level of 0.5. And, in fact, the frequency statistics for parity mappings are *always* like this. If we are dealing with n-bit parity then the highest order, non-degenerate frequencies are the (n-1)th-order frequencies. Given binary variables we will necessarily find exactly two occurrences of each (n-1)th-order case in the training set, and these two cases will necessarily show a different value for the variable excluded from the case. Thus the conditional frequencies for the case in question will be

evenly distributed between the two output cases and the conditional frequencies for instantiations of the output variable will always be identical. If they are identical, they must be precisely at their chance level. Thus, parity problems are always pure type-2.[5]

1.5 Complexity implications

Distinguishing between type-1 and type-2 problems helps to shed light on the complexity implications of different learning scenarios. Type-2 regularities are non-chance frequencies for cases observed in some *re-coding* of the original data. Thus, points in the space of type-2 regularities correspond to possible data re-codings, i.e., possible computational devices capable of processing those original data. The space of possible type-1 regularities, on the other hand, is made up of the set of all frequencies (conditional and unconditional) for the problem. Suffice it to say that the former space is, in general,

(i.e., which are constructed on the basis of an input/output rule that implicitly invokes a re-coding step) may well exhibit 'spurious' type-1 regularity.

The example training set used above illustrates this. The problem is 'intrinsically type-2' since the input/output rule used to construct the pairs assumes the re-coding step of converting the original input variables to their difference. And yet the type-1 frequencies show some marked, non-chance values (see the frequencies for the cases <x2=1> and <x2=2>). These would be straightforwardly exploited by processes that perform no re-coding whatsoever, e.g., learning algorithms such as the perceptron learning algorithm (Minsky and Papert, 1988) or Quinlan's ID3 (1986).

Even where intrinsically type-2 problems show very little spurious type-1 regularity they may still be solved by sophisticated learning algorithms such as backpropagation (Rumelhart, Hinton and Williams, 1986b), cascade-correlation (Fahlman and Lebiere, 1990) or copycat (Hofstadter, 1984).[6] It is, after all, well known that backpropagation can solve problems based on parity, symmetry or 'shift' relationships and that all these typically involve the algorithm deriving what can be thought of as an internal re-coding scheme.

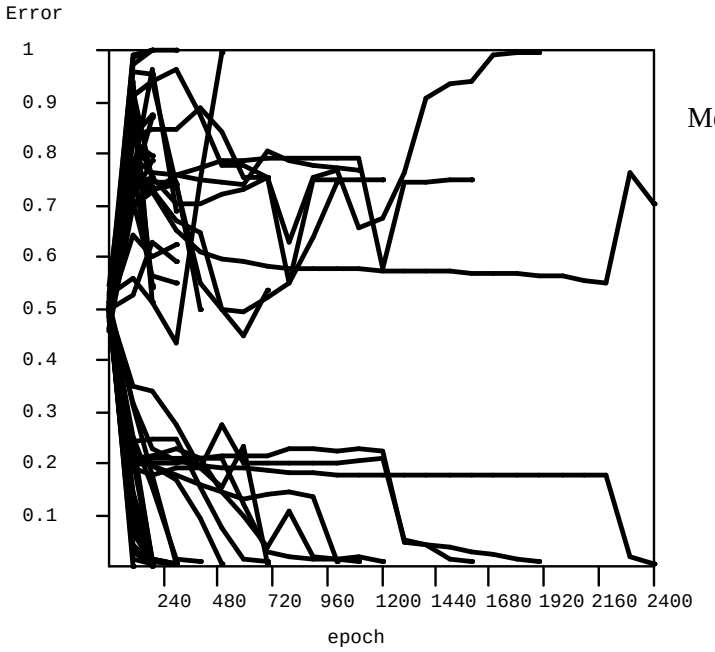
However, we should not overestimate the generality of such methods. All of them introduce restrictive assumptions about the nature of the type-2 regularity to be

[6] This latter is not usually presented as a learning algorithm. However it can certainly be construed as such.

discovered. Backpropagation for example effectively assumes that the required re-coding can be expressed in terms of the user-fixed architecture of semi-linear transfer functions, and that it can be discovered by the gradient descent method embodied in the learning algorithm. If the assumption is invalid, the learning necessarily fails.

This may help to explain why backpropagation often fails to solve low-order parity problems when they are presented as generalization problems (i.e., when some cases are held back for testing purposes). The graph shown in Figure 2 was produced from an empirical survey that involved running standard backpropagation (Rumelhart, Hinton and Williams, 1986b) on 4-bit parity generalization problems (with four, randomly selected cases used as unseens) using a wide range of internal architectures. All the curves in the upper half of the graph are error profiles[7] for the testing set of four cases. All the curves in the lower half of the graph are error profiles for the training set. There are 32 pairs of curves in all although many of them are bunched together in two clumps at the far left of the graph. Rather obviously, generalization over the testing cases was never observed to improve much beyond the chance level in any of the runs recorded. But the point to note is that the training-set error profiles typically go to zero rather rapidly (usually within 1000 epochs). This tells us that the generalization

[7] The error measure is the average difference between actual and target activations. For these experiments we used standard learning parameters; i.e., a learning rate of 0.5 and a momentum of 0.9.


Error
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
240
480
720
960
1200
1440
1680
1920
2160
2400
epoch

Mean error

failure occurs in the context of perfectly 'successful' learning, i.e., perfect acquisition of the training cases. This is a particularly concrete sort of generalization failure since it cannot be overcome by increasing the amount of training or by changing parameters. Once a supervised algorithm has learned the training cases perfectly, generalization grinds to a halt. As far as the algorithm 'knows', it is

in a visual field are *necessarily*

2 Incremental solutions

Given the likely ubiquity of learning problems involving relational (type-2) regularities how should we react to the ease with which natural systems appear to deal with them? One possibility is that there exists a more powerful, as-yet-undiscovered class of learning algorithms capable of performing type-2 search in an individual lifetime. Alternatively, we might conclude that nature's achievement is somehow to exploit forms of learning which involve only type-1 search in ways which somehow cumulatively lead to the solution of type-2 problems. Given our discussion of the statistical roots of the difficulty of type-2 search, we suspect that no conceivable individual learning algorithm will be able reliably to negotiate such spaces in biologically realistic time-spans. We will therefore investigate a version of the second response in which type-2 problems are reduced to incremental complexes of type-1 problems.

We can illustrate the basic idea using an example in which we imagine a type-2 learning problem being solved by a learner with rather limited computational abilities. The example is based on the training set shown below. This uses 21 variables and these are shown in the usual fashion starting with the first variable (called x1) on the far left and finishing with last (called x21) on the far right. (Only the integer parts of names for variables between x2 and x21 are shown). At the very end of each line there is a comment in square brackets.

```
x1 2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18 19 20 21
```

1 2 1 1 1 4 10

first card in hand A. Variables x3 and x4 represent the number and suit values of the second card in hand A, and so on.

vectors. In the simplest case, this involves applying 'hand-evaluation' functions to the relevant sub-ranges of input values and 'evaluation-comparison' functions to the values thus produced. But what happens if the re-coding is to be carried out by an agent (e.g. a learner) that is limited in its computational properties? What if we assume that it is *unable* to carry out such sophisticated operations as 'hand-comparison' and 'evaluation-comparison'?

It is tempting to say that the regularity in question then becomes *inaccessible* to the learner. But this is certainly an exaggeration. The learner might have access to functions that could be combined together somehow so as to bring the regularity in question to light. In this case, the regularity would be accessible but discoverable via a different route (i.e. trajectory).

Imagine, for the sake of argument, that the learner has access to a function that computes the equality of an arbitrary number of inputs and a function that computes the difference of two values. Let us also imagine that the learner is able to feed arbitrary constant values into these functions as they are applied. Given these computational properties, is there a re-coding of the Poker training set that brings out the underlying regularity?

Figure 3 illustrates one possibility. It shows a re-coding that involves several applications of both the accessible functions. Applications of the 'equals' function correspond to nodes marked with an 'eq'. Applications of the 'difference' function correspond to nodes marked with 'diff'. Inputs for these applications arrive

via the relevant branches. They may be the outputs of function applications or, at the

straight and hand B is two-pairs. According to the rules of Poker, a straight beats two-pairs. So we would expect values of this final output to be correlated strongly with values of the output variable x21. Thus we find that the re-coding

the limited function base, some of the computation that underpins the relevant regularity is 'picked off'. Eventually, a level of representation is reached at which the computation can be performed using one of the basic functions.

It is important that the sense of 'representation' in play here is not as rich

3 From Feature Detection to Analogical

> is required is some way to artificially constrain the solution space to just that region which contains the true solution. (Elman, 1991, p.8)

By 'false solutions' Elman means the extraction of the wrong regularities, i.e. finding spurious type-1 regularities which will fail to determine successful performance on unseen cases.

suing the type 1 statistics of a fragment or fragments of the overall training data and then using the feature detection skills thus acquired to reduce subsequent search; i.e., the overall problem is solved by the devious (temporally structured) exploitation of type-1 learning focussed initially on subsets of the overall training data.

This assumes, however, that the specific feature detectors needed to reduce subsequent search are discoverable just by looking at fragments of the training data appropriate to the final task. But not all learning problems (not even all those prone to a *kind* of incremental solution) will fall into this class. Instead, some target mappings may be learnable only if the search space is controlled by the exploitation of feature detectors which could *not* be acquired by focussing on subsets of the training data specifying the target mapping.

As an example, consider once again the conditional approach problem briefly described at the end of section 1. Successful learning here depends on the presence of two feature detectors: one for closeness and one for apparent width. The crucial feature which the net must later identify to learn the target mapping is actual width — a ratio between these two lower level features. But no amount of exposure to the data which embodies the conditional approach mapping will prompt the net to develop these two lower level feature detectors. Instead, they must be developed as a result of the early attempts of the system to learn to perform other kinds of task. In such cases the target problem has only what

we can call an *extended* incremental solution. The type 2 task (conditional approach) *can* be learned from the given data as long as the two prior feature detectors are present. But these feature detectors will not themselves arise merely as a result of exposure to that data, nor as a result of exposure to any subset of it. In these extended cases the right learning trajectory over time will indeed ensure the acquisition of the target skill. But the trajectory involves the importation of other feature detection skills acquired as a result of attempts to solve different problems.

Conservative incremental learning is thus self-sufficient in a way which extended incremental learning is not. Elman's grammar acquisition case is conservatively incremental and hence is well suited to a treatment involving a single network and a single (but usefully fragmented) body of training data.

Such modular

feature-detecting modules in wholly new overall problem solving contexts.)

Other ways of exploiting spatially extended cascades of processing devices

even if the statistical presence of human faces in the gross visual inputs to the system is low (relative to e.g. walls and buildings) the effective statistics (further

limited short term memory and techniques of modular decomposition all play

Such categories, he goes on to argue, must, however, be in some sense given (either explicitly or implicitly) in the body of input data. A category counts as explicitly given if the input coding isolates those very items (e.g. the category 'verb' is explicitly given if the input involves a variable which stands for verbs). And a category is implicitly given if the training data allows the net to induce such a category, although no individual variable coded for it in the input (e.g. NETtalk's induction of a partition corresponding to the noun/verb distinction on the basis of a corpus of unmarked sentences, see (Sejnowski and Rosenberg, 1987)).

Churchland's worry is that in some cases the category needed to make systematic sense of a body of data may not be present in that body of data — not even implicitly. Suppose that what is present (implicitly or explicitly), is conceived of as, in a broad sense, being observationally available. The question then becomes how a net can learn a category which would (a) make sense of a body of data but (b) is not observationally available in the data. The problematic case is thus described as one in which:

> ... a functionally essential category is simply not present in the input vectors at all, not even implicitly. Here the network must fail to learn if there is no functional relation that binds the observationally available categories in the absence of the hidden category. A net cannot learn a function that does not exist. (Churchland, forthcoming,

p.19)

The problem is pressing insofar as the course of human knowledge has involved the repeated positing of categories which (according to Churchland) are not given (implicitly or explicitly) in the data. Instead, we 'reach behind the appearances' to posit atoms, electrons, electromagnetic waves etc. etc.. How is this possible if learning consists (as the connectionist suggests) in a curve-fitting procedure defined over observables?

Churchland's answer is fascinating and suggestive. But before we consider it, it is worth pausing to question the way the problem has just been set up.

On the one hand, Churchland wants to recognize the ability of connectionist learning to induce new variables and representations. This is what happens when a net fixes on a regularity which was not explicitly marked in the input data. On the other hand, he sees that very often connectionist learning will be unable to discover a given regularity precisely because a specific variable is missing. To balance these two observations, he appeals to the notion of what information is actually present (implicitly or explicitly) in the input data. But on what grounds do we decide what information is or is not thus present? Churchland's underlying criteria looks to be this: if a curve-fitting procedure applied to that set of data points can unearth the regularity, it is counted as present (observable) in the data, and otherwise not. But this is hardly explanatory. What we want to know is why, given that the mapping is learnable if the data

is viewed through the lens of a specific category, is a network sometimes able to induce knowledge of the essential category and sometimes not? The distinction between type-1 and type-2 regularities provides the answer.

When the statistical regularity captured by a given category is type-1 relative to a training set, the kind of search undertaken by standard learning algorithms can be relied on to discover it. When the regularity is type-2, that kind of search will fail. What is needed in the latter case is some means of reducing the problem presented by the input data to that of extracting a type-1 regularity. The problem is not well understood by asking what is in some elusive sense in the training data. Rather, the issue concerns what can plausibly be got out of the training data given a certain manner of searching the space of potential relations between input items.

Thus (re)construed, Churchland's general problem concerns not just those rare cases where essential data is genuinely absent but also (and more commonly)

what he terms conceptual redeployment. Conceptual redeployment is a process in which a set of categories successfully induced to facilitate success in one domain is imported to another. Such importation (promoted by e.g. the chance juxtaposition of the intractable problem with a reference to the other, successfully-theorized, domain) allows the data from the problematic domain to be systematically reconceptualized via the lens of the imported categories. Thus, to follow Churchland's example, we may induce the categories of wave phenomena to make sense of observable input data concerning liquid behaviour. But once these categories (wavelength, velocity, frequency, etc.) are available, they may be redeployed to make sense of bodies of data from other domains (sound, light, etc.). Churchland comments that:

> While the conceptual prototypes of wave phenomena could not have been learned in either of these comparatively opaque domains, they were certainly capable of being redeployed there... (Churchland, forthcoming, p.25)

Without the transforming lens of the feature detectors for wave phenomena, the bodies of data concerning sound and light presented intractable problems of search. What Churchland has described, we believe, is a representational trajectory in which more tractable bodies of data (the water data) yield knowledge of variables (or even just feature detectors) whose use in pre-processing transforms the sound and light data into a form in which the target regularities are closer

to the surface and hence prone to succumb to the kind of search methods we actually have available. Churchland tends to depict conceptual redeployment (aka analogical reasoning) as a strategy in which no process of 'curve-fitting learning' occurs. But it is not clear that this is so. We claim instead that the imported categories/features reconfigure the data so that the curve-fitting is relatively trivial. Achieved representation is thus traded against intractable computation. But computation is not altogether abandoned.

4 Evolution, Creativity and Cognitive Closure

The single most important key to cognitive success, we have argued, lies in the use of a variety of diverse methods by which to transform intractable type-2 learning problems into temporally or spatially extended sequences of problems of an easier statistical stripe. Such methods may range from the genetically determined provision of simple feature detectors to the analogical redeployment of achieved knowledge. There is, however, a snag.

The snag is that all the methods available depict the solution of type-2 learning problems as in a certain sense fortuitous. By this we mean that it looks to be impossible, if our analysis is correct, to actively take a type-2 problem and work backwards to unveil a successful problem-solving trajectory. Type-2 problems get solved, it seems, only when we find ourselves in a position to solve them. We cannot put ourselves in a position to solve them.

learning. Or we may be lucky enough to be genetically provided with filters — pre-processors which transform the gross input data into a manageable form. What is lacking in all cases, however, is any general purpose type-2

whose effect is to promote the preservation of the structure of the successful sub-nets of previous generations. The complex problems which will succumb most gracefully to such a system are, of course, those problems which decompose into a set of tractable sub-problems.

Second observation: evolution flexibly co-constructs problems and solutions. The point here is that it is easy to overestimate the statistical difficulty of an evolutionary search which has terminated in a solution to a type-2 problem (such as conditional-approach). One source of such overestimates is a mistaken tendency to depict evolution as facing a specific problem and needing to search for a solution. Thus in the case of e.g. supervised connectionist learning, the system is given the task of learning a specific mapping M and must search the space defined by its architecture and inputs for a solution. But the evolutionary case is rather different. Any 'solution' which yields a surviving and reproducing being will do. If evolution then hits on the solution to a type-2 problem, we should not imagine that it had to perform a search whose unique terminus nothisevat14000(termin)999.95

which criss-cross individuals and outstrip human lifetimes. In addition, we can (by grace of such cultural institutions as schooling) easily re-create, time and again, the kind of learning trajectory which leads to the solution of key complex problems. In these ways, the occasional fruits of good fortune (the discovery of a powerful input re-coding (a concept) or a potent sequence of training items) can be preserved and used as the representational base-line of the next generation's mature explorations. Language and culture thus enable us to trade achieved representation in any member of the species, past or present, against computation for all posterity. Given the necessarily fortuitous nature of the search for new representations, this is an advantage whose significance cannot be exaggerated.

It is interesting to compare this vision (of language and culture as a means of

We think Dennett is almost right. He is right to depict language as a key factor in our abilities to frequently and repeatedly appear to exceed the bounds of ABC (or, as we would have it, type-1) learning. Yet in a very real sense there is, we believe, no other type of learning to be had. What looks like type-2 learning is in fact the occasional re-formulation of a type-2 problem in terms which reduce it to type-1. Language, we suggest, simply enables us to preserve and build on such reductions, insofar as the key to each reduction is an achieved re-representation of a body of data. But language is not, we think, the root of such re-representations. Instead, such re-representations must be discovered essentially by chance (perhaps aided by an endogenous, though undirected, drive to continuously seek re-coding of existing knowledge) in either individual or species learning. Language is a preserver of chance representational discoveries and of useful learning trajectories. It also doubtless feeds and augments the trick of analogical reasoning which we discussed earlier. Language-users will thus indeed steer a profoundly deeper course into the type-2 problem space than anyone else, but for reasons which are, we suspect, a little more pedestrian than Dennett imagines.

The three observations just rehearsed combine, we hope, to make our cognitive situation seem a little less bleak. True, we are type-1 learning devices inhabiting a world populated by much more complex regularities. But by trading representation against computation, in both individual and species time, we nonetheless make significant inroads into the type-2 space. That said, we must still live with

two depressing prospects. The first is that our image of true creativity threatens to become somewhat emaciated. For the important conceptual innovations are just those which bring what was previously an intractable type-2 learning problem down into the space of problems solvable by type-1 methods. These innovations turn problems which previously could not have been solved (in the very real sense of being intractable to type-1 learning) into ones which can be solved. We thus give a precise meaning to the distinction which Boden, in her (1990) investigation of creativity, makes between ideas which as it happens did not occur before and ideas which, in some real but elusive sense could not have occurred before (see Boden, 1990; p.31-41.) But alas, on our account, there is no intelligent means of searching for the re-codings which turn a previously type-2 problem into a type-1 format. Hard work, the constant juxtaposition of ideas in the hope of analogical 'trajectory hopping', the blind endogenous exploration of re-codings and sheer good fortune (genetic or otherwise) just about exhaust the possibilities. Intelligent reduction of the new categories needed to suck hitherto intractable problems into our cognitive ambit is, regretfully, just not to be had.

The second depressing prospect is that large tracts of genuine regularity in the universe threaten to be forever unknowable to us, and probably to any learning creature. For we can discover deeper (type-2) regularities only when such regularities lie on a convenient trajectory of achieved representations. But there is, we suppose, no reason to believe that all the interesting truths about the universe will lie on such trajectories. If a regularity is type-2 and there

is no natural sequence of problem solutions relative to whose representational products it reduces to type-1, then it is effectively unlearnable. We are thus quite dramatically cognitively closed (to use the terminology of McGinn (1989)) to phenomena of a certain well-defined type. Of course, we do not actively perceive this closure, as our universe is conceptualized by a body of achieved representations, and we continue to solve the problems defined in those terms. If nature itself then looks systematic and incremental to us, it may be because we are systematic and incremental learning devices who are cognitively closed to truths which lie outwith cumulative problem-solving trajectories.

To sum up, we have scouted both good news and bad. The good news is that despite the unexpected weakness of currently conceivable (type-1) learning methods, there exist a variety of ploys which enable us to solve type-2 problems. What all these ploys have in common is that they trade achieved representation against prohibitive computational search. The bad news is that there is no general, intelligent way of seeking just those re-codings which would transform a specific type-2 learning task so as to present a tractable type-1 regularity. Instead, the re-coding manoevre will save us only in those cases in which the required redescriptions are to be encountered along some natural problem solving trajectory. The boundaries of the genuinely *learnable* thus probably fall well short of the boundaries of what is in principle *representable* by a computing device such as the brain.

Conclusions: Probing the Unobservable

It is widely understood that the 'difficulty' of a particular computation varies according to how the input data are presented. With the data presented one way, achieving particular computational effects may require elaborate and expensive processing. With the data presented differently, it may be possible to achieve the same effects much more straightforwardly. (For a classic discussion, see Marr, 1982, p.21.) What is less well understood is the effect of this computation/representation trade-off within the learning paradigms. We have suggested that existing learning algorithms tend to rely predominantly (and in some cases exclusively) on the extraction of a specific type of regularity from a body of input data. This type of regularity lies close to the surface of the training data, in the form of pronounced frequency effects and is thus fairly straightforwardly extracted by a variety of strictly empirical methods. Some appearances to the contrary, the extraction of type-1 regularities is really all we currently know how to achieve — axclusivtra54(y)-y-dominanw

trick by which inherently weak learning devices regularly solve apparently tough type-2 problems?

The trick, we suggested, is to trade representation against computation; to use luck, genetic evolution and individual type-1 learning to yield representations which re-code the input sample and hence gradually transform the nature of the learning task. Being computationally weak, we compensate by valuing representational richness and constructing long problem solving trajectories in which a cascade of coding and re-coding is the essential prerequisite to effective learning.

Since no intelligent method of searching for the specific representations needed to render a target mapping learnable exists, good fortune must play a major role in our successful forays. But this reliance on good fortune begins to look less counter-intuitive once we see (a) that evolution is itself a naturally incremental species-level learning device, (b) that problems and solutions co-evolve (so we never really seek a solution to a specific problem) and (c) that language and culture provide us (the most successful explorers of type-2 problem space) with an invaluable means both of preserving and building on each and every fortuitous representational movement, and of recapitulating successful learning trajectories once they are achieved.

It may be useful, finally, to en000(ou2)-15999.s[(orers)-1399iutionr3(jor)-935Td[ut67(e)-1299noainstrvituir

data, and type-2 regularity which takes the form of non-chance frequency effects in some re-coding of the original data. Type-2 regularities are tractably discoverable only if the input data is systematically re-coded so as to highlight certain properties. Each such re-coding can be seen as effectively altering what is observable to the system in question. The basic task of higher cognition is thus to progressively expand an organism's 'observable' universe so as to suck in as many useful, previously type-2, regularities as possible. Such expansion always involves seeking and exploiting re-codings of the input data (representations) which re-shape the search space for other interesting regularities. This process is both effectively blind (unintelligent) and highly incremental. It results in a cascade of re-codings most reminiscent of Karmiloff-Smith's hypothesis of repeated 'representational redescriptions'. What we now see is why such a process is rapidly forced on us, courtesy of the surprisingly weak nature of achievable (type-1) learning.

It is no surprise, then, that incremental learning, in a variety of forms, has loomed so large in recent published attempts to expand the horizons of connectionist knowledge acquisition. Despite important individual differences, a common thread links all such treatments. What they all add to standard connectionist learning, is an increased opportunity to discover transformation factors: processing episodes which re-configure the statistical task involved in learning a given mapping. Modularization, incremental memory expansion, batched

means to the achievement of this common, statistically intelligible end. And the underlying trick is always the same: to maximise the role of achieved representation, and thus minimize the space of subsequent search. This now familiar routine is, as far as can we tell, obligatory. The computationally weak will inherit the earth, just as long as they are representationally rich enough to afford it.

Acknowledgements

Thanks to Bruce Katz, Noel Sharkey and Inman Harvey for useful comments on the text.

References

[1] Boden, M. (1990). *The Creative Mind: Myths and Mechanisms.* London: Weidenfeld and Niicolson.

[2] Churchland, P. (forthcoming). Learning and conceptual change: the view from the neurons. In A. Clark and P. Millican (Eds.), *Essays in honour of Alan Turing Vol.II: Concepts, Connectionism and Folk Psychology.* Oxford: Oxford University Press.

[3] Churchland, P. and Sejnowski, T. (1992). *The Computational Brain.* Cambridge, Ma.: M.I.T./Bradford Books.

[4] Clark, A. (forthcoming). *Associative Engines: Connectionism, Concepts and Representational Change.* MIT/Bradford.

[5] Clark, A. and Karmiloff-Smith, A. (forthcoming). The cognizer's innards: a

[12] Fodor, J. and Pylyshyn, Z. (1988). Connectionism and cognitive architecture: a critical analysis. *Cognition, 28* (pp. 3-71).

[13] French, R. (1991). Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. CRCC Technical Report 51, University of Indiana, Bloomington, Indiana., 47408.

[14] Hinton, G. and Sejnowski, T. (1986). Learning and relearning in boltzmann machines. In D. Rumelhart, J. McClelland and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Vols I and II* (pp. 282-317). Cambridge, Mass.: MIT Press.

[15] Hofstadter, D. (1984). The copycat project. A.I. Memo 755, Masachusetts Institute of Technology.

[16] Jacob, F. (1977). Evolution and tinkering. *Science, 196*, No. 4295 (pp. 1161-1166).

[17] Jacobs, R., Jordan, M. and Barto, A. (1991). Task decomposition through competition in a modular connectionist architecture: the what and where visual tasks. *Cognitive Science, 15* (pp. 219-250).

[18] Jacobs, R., Jordan, M., Nowlan, S. and Hinton, G. (1991). Adaptive mixtures of local experts. *Neural Computation, 3* (pp. 79-87).

[19] Johnson, M. and Morton, J. (1991). *Biology and Cognitive Development: The Case of Face Recognition.* Oxford: Blackwell.

[20] Karmiloff-Smith, A. (1979). *A Functional Approach to Child Language.* London: Cambridge University Press.

[21] Karmiloff-Smith, A. (1992). Nature, nurture and PDP: preposterous development postulates?. In A. Clark (Ed.), *Connection Science, special issue on Philosophical Issues in Connectionist Modelling, 4*, No. 3 and 4 (pp. 253-270).

[22] Marr, D. (1982). *Vision.* New York: W.H. Freeman.

[23] McGinn, C. (1989). Can we solve the mind-body problem?. *Mind, 98* (pp. 349-366).

[24] Minsky, M. and Papert, S. (1988). *Perceptrons: An Introduction to Computational Geometry* (expanded edn). Cambridge, Mass.: MIT Press.

[25] Nolfi, S. and Parisi, D. (1991). Auto-teaching: networks that develop their own teaching input. Institute of Psychology. C.N.R. Rome. Technical Report PCIA91-03.

[26] Plunkett, K. and Marchman, V. (1991). U-shaped learning and frequency effects in a multi-layered perceptron: implications for child language acquisition. *Cognition, 38* (pp. 1-60).

[27] Quinlan, J. (1986). Induction of decision trees. *Machine Learning, 1* (pp. 81-106).

[28] Ridley, M. (1985). *The problems of evolution.* Oxford: Oxford University Press.

[29] Rumelhart, D., Hinton, G. and Williams, R. (1986a). Learning internal representations by error propagation. In D. Rumelhart, J. McClelland and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Vols I and II* (pp. 318-362). Cambridge, Mass.: MIT Press.

[30] Rumelhart, D., Hinton, G. and Williams, R. (1986b). Learning representations by back-propagating errors. *Nature, 323* (pp. 533-6).

[31] Sejnowski, T. and Rosenberg, C. (1987). Parallel networks that learn to pronounce english text. *Complex Systems, 1* (pp. 145-68).

[32] Thornton, C. (forthcoming). *The Statistics of Conditional Approach.*

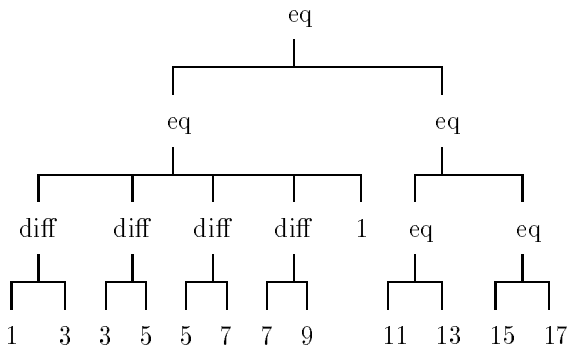


Figure 3: