Systematic minds, unsystematic models: Learning transfer in humans and networks

Steven Phillips

Information Science Division

Electrotechnical Laboratory

1-1-4 Umezono, Tsukuba, 305, Japan

stevep@etl.go.jp

Running head: Systematic minds, unsystematic models

I thank Graeme Halford for discussion on the psychological experiments, and Yoshiro Miyata for making available his PlaNet simulator. Author's address: Information Science Division, Electrotechnical Laboratory, 1-1-4 Umezono, Tsukuba, Ibaraki 305 Japan. Email: stevep@etl.go.jp. Tel: +81 298 543315. Fax: +81 298 545857.

ABSTRACT

Minds are said to be systematic: the capacity to entertain certain thoughts confers to other related thoughts. Although an important property of human cognition, its implication for cognitive architecture has been less than clear. In part, the uncertainty is due to lack of precise accounts on the degree to which cognition is systematic. However, a recent study on learning transfer provides one clear example. This study is used here to compare transfer in humans and feedforward networks. Simulations and analysis show, that while feedforward networks with shared weights are capable of exhibiting transfer, they cannot support the same degree of transfer as humans. One interpretation of these results is that common connectionist models lack *explicit* internal representations permitting rapid learning.

Key words. Systematicity, learning transfer, connectionism, classicism, weight sharing, normalization, Klein group

INTRODUCTION

Mind is, to *some* extent, systematic: being able to think some thoughts confers to certain other related thoughts. But the degree to which it is systematic, and what that implies for the underlying architecture is to say the least controversial. Since Fodor and Pylyshyn's (1988) seminal paper, numerous subsequent articles purport to address the issue in one way or another. Yet, relatively few have proposed new connectionist models as a direct consequence of the limitations expressed in the original article. Perhaps the so-called *new connectionism* already provides for sufficiently powerful models, or perhaps the implications of systematicity for (connectionist) cognitive architecture are not sufficiently well understood.

Part of the difficulty from a computational perspective is that it is not clear what constitutes a definitive test for any particular model. If one considers systematicity as a type of generalization (Hadley, 1994; Phillips, 1995), then what constitutes a fair test of generalization? While one may accept that knowing "John loves Mary" implies the capacity to know "Mary loves John", it is less clear how logical inference, for example, should extend from thematic to abstract cases (van Gelder & Niklasson, 1994). Furthermore, even though the "John loves Mary"-"Mary loves John" symmetry eliminates an elemental association based architecture[1], such symmetries are enforcible in a network augmented with an appropriate learning function.

Adding a learning function to a network, makes certain network (weight) states inaccessible. When links are updated upon learning to represent the complex object "John loves Mary", for example, they simultaneously represent the related object "Mary loves John". Symmetry is enforced by 'energy' minimization over an appropriate training set. Minimization by gradient descent causes a change in state from not being able to represent either

[1]Briefly, such architectures have states that permit the capacity to represent one instance but not the other, by setting appropriate links from constituents (e.g., "John", "loves", etc) to just one of the two complexes (e.g., "John loves Mary"). Such states have no correspondence in human cognition. And there is nothing within the architecture itself that enforces this symmetry (Fodor & McLaughlin, 1990).

complex object to being able to represent both. But, as a consequence of the shape of the energy surface, it does not permit a change in state whereby only one of the two objects is representable. It would appear, at least in principle, that energy minimization is sufficient to account for certain symmetries in behaviour.

The use of generalization to explain systematicity, however, is crucially dependent on what one regards as an appropriate training set. It is of no use if the proposed architecture must make use of an unrealized number of training examples to attain a particular symmetry. It is also necessary to justify the degree of generalization supported by the model.

One approach is to attempt to characterize the observed grouping of acquired cognitive behaviours. Such characterizations become benchmark tests for generalization in network models. In this way, Hadley (1994) identified problems with connectionist claims of generalization. From language, Hadley proposed, in part, "generalization across syntactic position" as a definition of systematicity. His analysis of generalization data for existing networks showed no evidence of this degree of generalization. Subsequent simulation and analysis showed why (Phillips, 1998). (See also Marcus, 1998, for another characterization and network analysis.)

Although Hadley's definition of systematicity appears reasonable, this approach runs the risk of presupposing a construct to which the behaviour is supposed to constrain. For instance, according to Christiansen and Chater (1994), *syntactic position* is not even a standard term in linguistics. A consequence of this approach is that systematicity becomes a property of a particular model of cognitive behaviour, not of cognitive behaviour *per se*. That is why Matthews (1994) regarded systematicity is an unresolvable issue for the connectionist, since Fodor and Pylyshyn's original definition is specified in terms of a particular architecture (i.e., classical), leaving no room for an alternative theory.

An alternative, more theory-neutral approach is to propose a particular structure from

which instances are generated, and test humans on their generalization capacity over this set. Performance can then be compared to particular models over the same set, with the goal of determining what computational properties are responsible for enforcing (the lack of) systematicity. In the next section, one such case is presented from learning transfer experiments (Halford, Bain, Maybery, & Andrews, 1998). I then examine a feedforward network shown to exhibit transfer using the technique of weight sharing (Hinton, 1990). The simulations and analysis show that, although this network has the capacity to demonstrate transfer, it cannot demonstrate the same degree of transfer as humans. Finally, some extensions to the standard feedforward network are considered, which suggest greater emphasis should be placed on networks that directly access their own internal representations for the purposes of modeling higher cognition.

LEARNING TRANSFER IN NETWORKS AND HUMANS

Learning transfer is the phenomenon whereby performance (as measured by the number of learning trials, for example) on one task is significantly better having learnt a structurally related task. Since Harlow (1949) it is known that the degree of transfer differentiates species (see Kendler, 1995, for a review). Rats, for instance, show almost no transfer even after hundreds of tasks, whereas humans show the greatest degree of transfer (Kendler, 1995). The degree to which humans transfer learning suggests some form of symbolic representation. Thus, it provides an interesting domain for connectionist modeling: A demonstration of transfer by a connectionist model would provide the first step[2] of a counter-example to the claim that cognitive architecture must be *classical*[3]. Hinton (1990) invites this sort of interpretation upon demonstrating generalization across isomorphic data sets using networks with shared weights.

[2]The second step is to show that the connectionist solution is distinctly non-classical.

[3]Having syntactic representations, and processes sensitive to their structure (Fodor & Pylyshyn, 1988).

Neural networks: Family trees task

In Hinton's work, a feedforward network was trained to make inferences about relationships in two isomorphic family trees (see Figure 1). The network is required to learn the mapping: Person 1, Relation $\rightarrow$ Person 2 (e.g., John, Wife $\rightarrow$ Mary) for each family tree. The network consisted of input units for each family member in each family; and each family relation. Input units were connected to a layer of hidden units, which encoded distributed representations of local input vectors. The hidden units were connected to a common second hidden layer, which was connected to a third hidden layer, which encoded distributed representations of Person 2. Finally, the third hidden layer was connected to output units, representing local target output vectors for Person 2 in each family. The network was trained on 100 of the 104 possible person-relation pairs. In one simulation run, the network generalized to all four remaining test cases. In the other run, it generalized to three test cases.

Insert Figure 1 about here

An analysis of input-to-hidden weights showed considerable similarity between the two families (Hinton, 1990, Figure 5). Very similar representations were learned for corresponding family members at the first hidden layer. Apparently, the shared weights between hidden layers encoded common structure. If all the relations are learned for one member of a family, then mapping the corresponding member for the other family to the same location guarantees generalize to all its relations. Importantly, since there was no input similarity between vectors representing the family members, generalization was based on common structure, not similar input patterns. Hence, it apparently answers the criticisms of Fodor and Pylyshyn that connectionist models are not structure-sensitive. But, the degree of generalization was very limited, and at best only four cases in 104. It is questionable whether such demonstrations

are indeed tapping the mechanisms of human inference.

Human subjects: Klein 4-group task

Halford et al. (1998) reported rapid transfer over a series of four tasks derived from the Klein-4 group. Each task conformed to the following structure:

Klein	a	b	c	d
H	b	a	d	c
V	d	c	b	a

where a, b, c, d are states; and H and V are state transitions. When states are depicted as vertices of a square, H and V appear as horizontal and vertical transitions, respectively (Figure 2(a)). A task instance consists of four randomly generated strings and two shapes. Subjects are presented with a string and a shape, and asked to predict the response string. For example, in Figure 2(b), **PEJ** and $\triangle$ predicts **BIP**. In each trial, all eight possible string-shape pairs are presented one at a time in random order. After making a prediction, subjects are informed of the correct response. No reference is made to the structure and underlying meaning of the task. Learning within a task instance continues until all eight pairs are correctly predicted within a single trial, or to a maximum of six trials. After the intra-task learning criterion is reached, or after six trials, the next task instance is presented, until four task instances have been completed. By the fourth task, the first trial error rate was 2.00 for a group of 12 participants.

Insert Figure 2 about here

Halford et al. showed that this degree of transfer can be obtained if one interprets strings and shapes as states and state transitions. For example, given the two stimulus-response pairs (from Figure 2(b)): 1. (**PEJ**, $\triangle$) $\rightarrow$ **BIP**; and 2. (**SIY**, $\bigcirc$) $\rightarrow$ **PEJ**, one can make the

interpretations:[4] (**PEJ**, **a**); (△, **H**); (**BIP**, **b**); (◯, **V**); (**SIY**, **d**); and (**JAS**, **c**). The third stimulus pattern (**BIP**, ◯) is predicted as **JAS** via the interpretation and task structure. The three steps are: 1. **BIP** → **b**; and ◯ → **V**; 2. (**b**, **V**) → **c**; and 3. **c** → **JAS**.

FEEDFORWARD NETWORKS WITH SHARED WEIGHTS

Before detailing and expanding upon an analysis of transfer initiated in Phillips and Halford (1997), some justification is given for the choice of local input/output representations[5] for task elements used here, and in Hinton (1990). Generalization performance is sensitive to the choice of input/output representations. In an extreme case, if all elements are represented by a single real number, then arbitrarily many generalizations are possible after training on only two examples in a linear system with one variable. Although there is some similarity between elements between task instances (e.g., strings may share common characters), this similarity is not the basis for generalization, since the assignment of task elements is arbitrary across task instances. The use of local representations acts as a control measure to eliminate any possibility of generalization on the basis of surface (input pattern) similarity.

Simulation

A common method for demonstrating generalization is to partition data into training and testing sets, where the inputs and outputs range over the same vector space. However, in schema induction typically the stimulus and response materials do not appear in more than one task. Therefore, particular attention must be given to the way input and target vectors are represented in the network. One way is to use a different group of input/output units for each task, with learning transfer on the basis on common connections between

[4](**JAS**, **c**) comes from the knowledge that they are the only remaining uninterpreted elements.
[5]That is, a single unit with activation 1, and the rest 0.

hidden units (Figure 3(a)). This style of network is very similar to that used by Hinton (see Figure 1). However, this approach is cumbersome for a longer series of tasks since it adds many additional weights and units that are only updated during one of the tasks. The approach adopted here was to use the same units and weights for each task, but to reset the input-to-hidden and hidden-to-output connections. This approach simulates the use of novel materials across tasks, while permitting knowledge transfer by not resetting the hidden-to-hidden unit connections (Figure 3(b)). Dashed arrows indicate the actual weights reset during simulations (see Method).

Insert Figure 3 about here

Another consideration is the number of hidden layers (at least two) and the number of units within each layer. Importantly, the number of weights (free parameters) must be small enough to facilitate generalization, but large enough to support a solution. Preliminary simulations suggested a 6-3-2-4 network, where the six input units (four states plus two operators) are connected to the first hidden layer of three units, connected to the second hidden layer of two units, connected to the four output units. For units with linear threshold activation functions (e.g., sigmoid: $f(x) = \frac{1}{1+e^{-x}}$, used here), it can be shown that two is the minimum number of units for the second hidden layer, and that two is a lower bound for the first hidden layer, under the condition of local input/output vectors (Phillips & Halford, 1998, Appendix A). Preliminary simulations with a 6-2-2-4 failed to learn all patterns, so the 6-3-2-4 network was used. Use of more hidden units only decreases the likelihood of generalization as it introduces more free parameters for the same number of examples.

Method

Preliminary simulations showed that the network failed to learn many of the patterns in the second task despite successfully learning all patterns in the first task. This, despite the fact that one of the solutions to the second task is the same set of weights learnt from the first task, some of which were already specified. The inability to learn the second task introduces a methodological problem: how to examine generalization when the network cannot find any of the available solutions to the training set. There are several ways to overcome this problem, for example: use more trainable weights; fix fewer weights common to both tasks; or, use more powerful learning methods. However, a failure to demonstrate generalization under these circumstances is always subject to the "what if you tried ..." response. Alternatively, one can take a bounds approach by identifying limits to the degree of generalization capable by the network. If the limit falls below the generalization criterion then we can say the architecture does not support learning transfer.

Accordingly, the following procedure was adopted: (1) train the network to predict the final state given the initial state and operator for all first task instances; (2) reset only those weights connected from the input unit corresponding to the test pattern for the second task; and (3) retrain the network with all other patterns, and all other weights and biases fixed. In effect, the network is being trained on all patterns for the first task and all patterns except one for the second task. The exceptional pattern is used for testing. If the network cannot demonstrate generalization under this condition, then it cannot support learning transfer since there is no further information (training examples) available from the environment. Therefore, support for learning transfer can only come from additional network specific information biases. In other words, architectural properties that support learning transfer.

Figure 3(b) shows the weights (dashed arrows) reset for learning the second task. The network weights were randomly initialized from a 0 mean 0.5 variance normal distribution,

and updated using the standard backpropagation algorithm (Rumelhart, Hinton, & Williams, 1986) with squared difference between output and target patterns as the error function, and a learning rate of 0.1 (no momentum). Training continued until the average squared error for each output unit and pattern reached 0.01 training[6]. Networks were examined for transfer in the cases where one and three weights from the input unit representing the state in the state-operator pair for the test pattern were reset. Since local input/output vector representations were used, this condition corresponds to retraining on all patterns for the new task except the single test case where the input pattern corresponds to that unit. Results are reported for the second task instance.

Results

When one weight was reset, after retraining the network correctly responded to all seven training patterns in all 10 trials. The network correctly predicted the test pattern in seven of 10 trials. When three weights were reset, after retraining the network correctly responded to all seven training patterns in four of 10 trials. In the other six trials, the network correctly responded six of the seven training patterns. For two of these six trials, the network correctly predicted the test pattern. For the other eight trials, the test pattern was not predicted.

Analysis and discussion

Although the network demonstrated generalization when one weight was reset, there was no evidence of generalization when three weights were reset. When three weights were reset the test pattern was predicted in only two of 10 trials (20%), which is not better than chance (25%). Furthermore, in both of these trials the difference between the two most activated output units was only marginal (< 0.02), and in both cases the network did not respond correctly to all seven training patterns.

[6]Sufficient for correct response to all training patterns.

The most pertinent result is the four trials where all seven training patterns were learnt without generalization to the single test pattern. In these cases, the network has learnt a solution to the training set that was not a solution to the test set. Given the host of parameters accompanying backpropagation, negative learning results may simply be attributed to unsuitable parameter settings: a large step size may result in overstepping the global minimum; an early (or late) stopping criterion may result in over-generalization or over-specification. Rather than attempt some 'reasonable' coverage of the parameter space, a potentially more informative approach is to plot the error surface for a region of weight space to gain an understanding of the difficulty of the learning task.

Although it is not possible to view the entire (32-dimensional) error surface directly, we can obtain an understanding of the difficulty of the task by viewing a portion of the error surface around the global minimum. By plotting the error surface along two dimensions in the vicinity of global minimum for both the seven training patterns and all eight patterns, we can observe how the error surface changes with the additional test pattern (Phillips, 1997).

The network was trained until correct response on all eight patterns. Weights from one input unit to all first layer hidden units were reset (Figure 3(b)). The network was retrained to correctly respond to seven of the eight training patterns (simulating learning on the second task). The error surface along two of the three reseted weights was plotted for the seven training patterns (Figure 4(a)). The vertical z-axis indicates total error as a function of two weights (x,y-axis). The flat region (grey area) is the global minimum for the training set. The error surface along the same two weights, but for all eight patterns (including the single test case) is shown in Figure 4(b). The global minimum (grey area) in this case is much smaller than for the training condition.

Insert Figure 4 about here

Clearly, the training set does not constrain the error surface to be the same as for the test set. In other words, the network cannot guarantee generalization to even a single test case. Similar results where found for the other two weight pairs, and for other trials. The error surface plots make it clear that the two apparently successful trials were the results of fortuitous initial weight settings.

It should be noted that the training method employed here does not guarantee that shared weights will represent common task knowledge. The absence of subsequent task constraints when learning the first task means that both structural and specific knowledge could be distributed across shared weights. Since these weights are frozen after learning the first task, knowledge specific to the first task stored at the shared weights lessens the likelihood of transfer as it is irrelevant to learning the next task. Hinton's method attempts to address this problem by learning tasks in parallel. Task specific information represented on a limited number of shared weights may leave the other task instance unrepresentable. Barring local minima, the resulting error in the unrepresented task should force an alternative solution. However, this approach appears psychologically implausible. Although task relevant information must be used in learning subsequent task instances, it is unlikely to be of the form of specific input/response patterns.[7]

Ideally, one wants the weight space that will facilitate the highest likelihood of transfer. Independent of the serial/parallel training procedure, one can consider the smallest weight space that will support a solution to each task instance. As well as number of weights, the effective weight space of the network can be restricted by enforcing fewer activation states for its units (e.g., binary, rather than real valued). In the extreme case, the smallest number of identifiable states for the second hidden layer is four (i.e., one state for each possible

[7]Which could be checked by testing whether subjects remember responses for previous task instances.

response). Fewer states means that at least one state must be mapped to two different responses. Such reduced representations can be implemented with a single binary valued unit for each internal state and 0/1 weights connected to the output; or, by one real valued unit and weights with non-monotonic activation functions at the output units. In the second case, since it is not possible to partition four points on a line using single threshold units, double threshold units are required. Gaussian functions, for example, have two thresholds permitting each point to be separated from every other point.

In either case, how many patterns are required to learn a new task? The lower bound is four (i.e., one pattern for each possible response). Each output unit must be trained to discriminate between two types of points: the pattern it represents; and all other patterns. With fewer patterns, at least one of the output units will be trained on only one type of point (i.e., the patterns represented by the other output units). In this case, the training set provides no information about discrimination for that output unit. Since the upper bound on generalization is four patterns, this architecture cannot be said to support systematicity, as defined in terms of learning transfer.

Before moving on to extensions to the feedforward network, it is also noted that this analysis has implications for the simple recurrent network, which Elman (1990) and others (Christiansen & Chater, 1994; Niklasson & van Gelder, 1994) have shown exhibits some degree of generalization across common structure. A simple recurrent network applied to the Klein 4-group task includes all the weights and units of the feedforward network, plus additional weights for mapping the context (hidden unit vector from the previous time step) to the hidden units. Since the additional connections in the simple recurrent network make the weight space even less constrained than the feedforward network, it is highly unlikely to exhibit generalization for the Klein 4-group task. Therefore, it also cannot be said to support systematicity.

VARIATION ON THE 'STANDARD' MODEL

Reduced descriptions, or minimum description length codes are a recent popular connectionist technique for improving generalization. Given their close connection to physical concepts such as entropy, one may be tempted to hypothesize their existence in higher cognitive mechanisms. However, the bounds identified in the previous section raise an important limitation: reduced descriptions, and the like cannot be the (sole) basis for learning transfer in humans. Although reduced descriptions improve generalization, they do not make use of the fact that in each task, one and only one task element was generated from one and only one structural element.

One-to-one correspondence is routinely incorporated as a basic property of analogical models (e.g., Hummel & Holyoak, 1997). Hummel and Holyoak's LISA model of analogical reasoning and schema induction makes use of weight normalization to enforce one-to-one correspondence between task and structure elements. While a full analysis of this model is beyond the scope of this paper, it is important to consider the mechanism as it highlights a change of philosophy in connectionist modeling: what I consider to be a move toward *explicit* internal representations. What follows is a sketch of how weight normalization could account for learning transfer as a prelude to a discussion on its characteristic difference from standard (i.e., more common) connectionism.

Enforcing one-to-one correspondence

Realizing that each task element corresponds to one and only one structural element permits greater transfer. For example, if we know from the first two stimulus-response trials that elements A, B and D map to particular structural elements, then we can infer that element C maps to the last remaining unaligned structural element, even though C does not appear in the first two trials. Use of the one-to-one correspondence principle permits correct

prediction after two training examples (for a detailed example see Introduction).

Suppose a group of input/output units (I_a, I_b, I_c, I_d) for representing task stimuli, and a group of hidden units (H_a, H_b, H_c, H_d) representing corresponding structural elements. Following along the lines of Hummel and Holyoak's (1997) model, a procedure for ensuring one-to-one correspondence is outlined in the following example:

- Suppose input pattern A is presented at the input/output units. The unit I_a becomes active and propagates its activation to the hidden units. Mutual inhibition between hidden units ensures only one unit (say, H_a) is most active.

- Weights linking co-active units have their weights increased. The outgoing weights from a unit are normalized. Normalization increases the weight between the two most co-active units (I_a-H_a) at the expense of the other outgoing weights. It ensures that unit I_a only activates the hidden unit H_a (i.e., task element A is aligned to one and only one structural element). However, at this stage unit H_a could be activated by other input/output units.

- To constrain hidden units from being activated by other input/output units, incoming weights to a common unit are also normalized. As with outgoing weights, normalization increases one weight at the expense of others. In this case, it has the effect of ensuring that a structural element corresponds to one and only one task element.

- If weights are bounded above by one, and normalized to sum to zero (Hummel & Holyoak, 1997), then weights between aligned pairs (e.g., I_a-H_a) will approach one and non-aligned pairs (i.e., I_i-H_a and I_a-H_j, where $i, j \neq a$) will be negative.

Suppose the first two trials consisted of stimulus-response patterns: (A,H) $\rightarrow$ B; and (D,V) $\rightarrow$ A. By the end of the second trial, all three elements have been aligned to their corresponding structural elements. For the sake of exposition, suppose elements are aligned

by setting weights I_a-H_a, I_b-H_b and I_d-H_d to one. By virtue of normalization, all other weights into or out of these six units will have small negative values. This leaves the single I_c-H_c connection. Although, this weight was not directly trained by the presence of the C element (since it does not appear in the first two trials), it is indirectly trained by its absence. All other connections from the unit I_c will have small negative weights, since they are connected to non-aligned units (i.e., hidden units already aligned to other input/output units). Similarly, all other connections to the unit H_c will have small negative weights. Assuming a small positive starting weight for connection I_c-H_c, then normalization will force this value to become strongly positive. On the third trial, when element C is presented to the network it will already be aligned to the structural element represented at H_c, resulting in correct prediction of the response element. After training on two patterns, non-local weight modification permits generalization to the other six patterns.

DISCUSSION

This paper was motivated by comments on earlier work from an anonymous reviewer who questioned how it is possible to determine the systematic nature of human cognition, given the difficulty of observing a subject's representational state, and their sensitivity to contextual information. The answer lies in externalizing the underlying task structure by specification, and controlling context through structurally consistent but otherwise meaningless materials. The experimental paradigm devised by Halford et al. has these criteria, and provides the basis for an objective measure of systematicity in both humans and networks. Once a test for systematicity has been specified it then becomes meaningful to analyze network models for the same property. So, it was shown that standard feedforward networks do not support systematicity, defined as a degree of learning transfer. The standard feedforward network does not enforce one-to-one correspondence between input and internal representa-

tional states. Weight normalization has this property and can, in principle, achieve the same degree of transfer.[8]

This difference in generalization property highlights a number of points. Firstly, although its not feasible to evaluate all of the almost infinite variety of network architectures directly, one can rule out a significant subclass and thereby indirectly evaluate its specific members. The limitation of the feedforward network lies in the shape of its error surface, which is derived from the data set, connectivity, activation and error functions. In this regard, one can eliminate the numerous variations based on step size and/or direction (e.g., acceleration terms, conjugate gradient methods, etc). In general, these changes affect the time taken to find the minimum, but not its shape relative to that determined by the test set.

Secondly, subtle differences in generalization levels transcend architectural boundaries. Thus, generalization alone is not sufficient evidence for a particular hypothesis regarding the nature of mental representation. For example, Hinton remarks that his family trees simulations provide support for a *componential* (feature-set) representation of concepts, which had been rejected for its lack of structure-sensitivity (Fodor & Pylyshyn, 1988):

> "The family trees example shows that componential reduced descriptions can be learned from structural information about how concepts go together within propositions. Given a sufficiently powerful learning procedure, the structuralist information can be converted into componential representations that facilitate *rapid* intuitive inference."
>
> (Hinton, 1990: p.59, emphasis here).

But, as is evident from the analysis presented here, any claim about the nature of concepts

[8]However, see Phillips and Halford (1998) where reverse inferences (e.g., what operator results in a given start-end state transformation) were also considered, which has further implications for connectionist architecture.

is contingent upon quantifying the notion of rapid inference. Local feature set based concepts were rejected by Fodor and Pylyshyn because there was no mechanism for enforcing systematic generalizations. They assumed the same also applied to distributed features sets (Fodor & Pylyshyn, 1988, footnote 10). Hinton's demonstration of generalization apparently provides a counter-example. But, the work here shows that this method cannot achieve the same level of generalization as humans. Hence, the distributed version of feature sets is also rejected.

Finally, it is worth generalizing the difference between the feedforward network with and without weight normalization in anticipation of how connectionism should depart from its standard reliance on externally driven learning. This difference is characterized in terms of *implicit* versus *explicit* internal representations. With normalization internal representations are explicit in that learning is a function of weights within the same layer, which represent the alignment of task to structure elements. By contrast, in standard backpropagation, learning is a function of activation vectors, and possibly weights from outer layers. But, weights within the same layer are not part of weight update function. In this sense, the internal representations that these weights maintain are implicit to the learning function, indirectly affecting learning by the output vectors they generate. But, why should this indirect cause and effect matter? Because multiple internal representations can result in the same response, only some of which are ultimately correct, further examples are needed to isolate and correct the error. Conversely, a consequence of explicit internal representation is that learning can proceed in the absence of external input. Thus, it was shown in the previous section, weight normalization permitted correct alignment of the last remaining pattern even though it did not appear anywhere in the training set. The distinction follows from Kirsh's (1990) proposal that information is explicitly represented when it is efficiently accessible: explicit representation affords greater computational efficacy, realized in this

context as greater generalization.

The emphasis on explicit internal knowledge is a recurring theme in discussion on the nature of cognitive architecture. Karmiloff-Smith (1992), for example, proposed that cognitive development includes a process of redescribing one's internal representations, which entails making them explicit: explicit representations are objects of manipulation by mental processes. Clark and Thornton (1997) argued that a significant class of problems are not directly solvable from their input-output specification. Instead, inputs must be recoded by internal mechanisms, which is tantamount to explicitation. Both examples underscore a need to move away from common *first-order connectionism*, where networks make explicit use of external, but not internal states of affairs, to what could be called *second-order connectionism*, where internal states are also explicit, in at least the sense previously described. Of course, this distinction in itself does not solve any specific problem - the critical question is how to make use of those internal representations. But, failure to make the distinction confines connectionism to the role of providing theories for biological transducers.

CONCLUSION

Rather than presuppose the structural basis for a grouping of particular cognitive behaviours, I have started with data generated from a particular structure and examined whether networks exhibited the same grouping as human subjects. Networks that rely on weight sharing alone cannot support the same degree of systematic generalization as humans. Weight sharing may work for large tasks where the number of patterns approaches the number of shared weights, but not for tasks with fewer patterns. There are simply too many possible solutions in weight space for the training set to sufficiently constrain the learning mechanism. Many solutions to the training set lie outside the region that contains solutions to the test set. Hence, generalization is highly unlikely.

Generalization was improved to a level closer to that of humans using a learning mechanism that implemented one-to-one correspondence between task and structure elements. In this case, weight change is a direct function of other weights representing task knowledge. By contrast, standard feedforward and recurrent networks have direct access to inputs and target outputs only. Consequently, common task knowledge embedded in shared weights is implicitly represented, and at best affords limited generalization. The results suggest renewed emphasis on a class of (second-order) connectionist models that have direct access to their own internal representations.

It is fair to say that the systematicity debate has been largely confined to the philosophical wing of cognitive science. Connectionists, for the most part, having sighted(/cited) the furor proceed in the belief that a little tuning and a generous amount of interpretation will resolve the issue. Yet, systematicity made sufficiently rigorous narrows the scope for interpretation leaving, in one case, no room to maneuver. That should be reason enough to reconsider what should be the basic elements of a connectionist cognitive architecture.

REFERENCES

Christiansen, M. H., & Chater, N. (1994). Generalization and connectionist language learning. *Mind and Language*, *9*(3), 273–287.

Clark, A., & Thornton, C. (1997). Trading spaces: Computation, representation and the limits of uniformed learning. *Behavioral and Brain Sciences*, *20*, 57–90.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*, 179–211.

Fodor, J. A., & McLaughlin, B. P. (1990). Connectionism and the problem of systematicity: Why Smolensky's solution doesn't work. *Cognition*, *35*, 183–204.

Fodor, J. A., & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, *28*, 3–71.

Hadley, R. F. (1994). Systematicity in connectionist language learning. *Mind and Language*, *9*(3), 247–272.

Halford, G. S., Bain, J. D., Maybery, M. T., & Andrews, G. (1998). Induction of relational schemas: Common processes in reasoning and complex learning. *Cognitive Psychology*, *35*, 201–245.

Harlow, H. F. (1949). The formation of learning sets. *Psychological Review*, *42*, 51–65.

Hinton, G. E. (1990). Mapping part-whole hierarchies in connectionist networks. *Artificial Intelligence*, *46*(1-2), 47–76.

Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review*, *104*(3), 427–466.

Karmiloff-Smith, A. (1992). *Beyond Modularity: A developmental perspective on cognitive science*. Cambridge, MA: MIT Press/Bradford Books.

Kendler, T. S. (1995). *Levels of cognitive development*. Mahwah, NJ: Lawrence Erlbaum Associates.

Kirsh, D. (1990). When is information explicitly represented?. In P. Hanson (Ed.), *Information, Language and Cognition: Vancouver Studies in Cognitive Science*. Vancouver, BC: UBC Press.

Marcus, G. (1998). Rethinking eliminative connectionism. *Cognitive Psychology*, *37*, 243–282.

Matthews, R. J. (1994). Three-concept monte: Explanation, implementation and systematicity. *Synthese*, *101*, 347–363.

Niklasson, L., & van Gelder, T. (1994). Systematicity and connectionist language learning. *Mind and Language*, *9*(3), 28–302.

Phillips, S. (1995). *Connectionism and the problem of systematicity.* Ph.D. thesis, The University of Queensland, Department of Computer Science, Brisbane, Australia. Available from http://www.etl.go.jp/~stevep.

Phillips, S. (1997). Limits of generalization: An error surface view. In *Proceedings of the 1997 Annual Conference of Japanese Neural Network Society: JNNS'97*, pp. 188–189.

Phillips, S. (1998). Are feedforward and recurrent networks systematic? Analysis and implications for a connectionist cognitive architecture. *Connection Science*, *10*(2), 137–160.

Phillips, S., & Halford, G. S. (1997). Systematicity: Psychological evidence with connectionist implications. In M. G. Shafto & P. Langley (Eds.), *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*, pp. 614–619.

Phillips, S., & Halford, G. S. (1998). An analysis of learning transfer in neural networks with relevance to connectionism.. Unpublished manuscript. Available from http://www.etl.go.jp/~stevep.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning Internal Representations by Error Propagation*, Vol. 1 of *Computational models of cognition and perception*, chap. 8. Cambridge, MA: MIT Press.

van Gelder, T., & Niklasson, L. (1994). Classicism and cognitive architecture. In A. Ram & K. Eiselt (Eds.), *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society*, pp. 905–909. Lawrence Erlbaum.

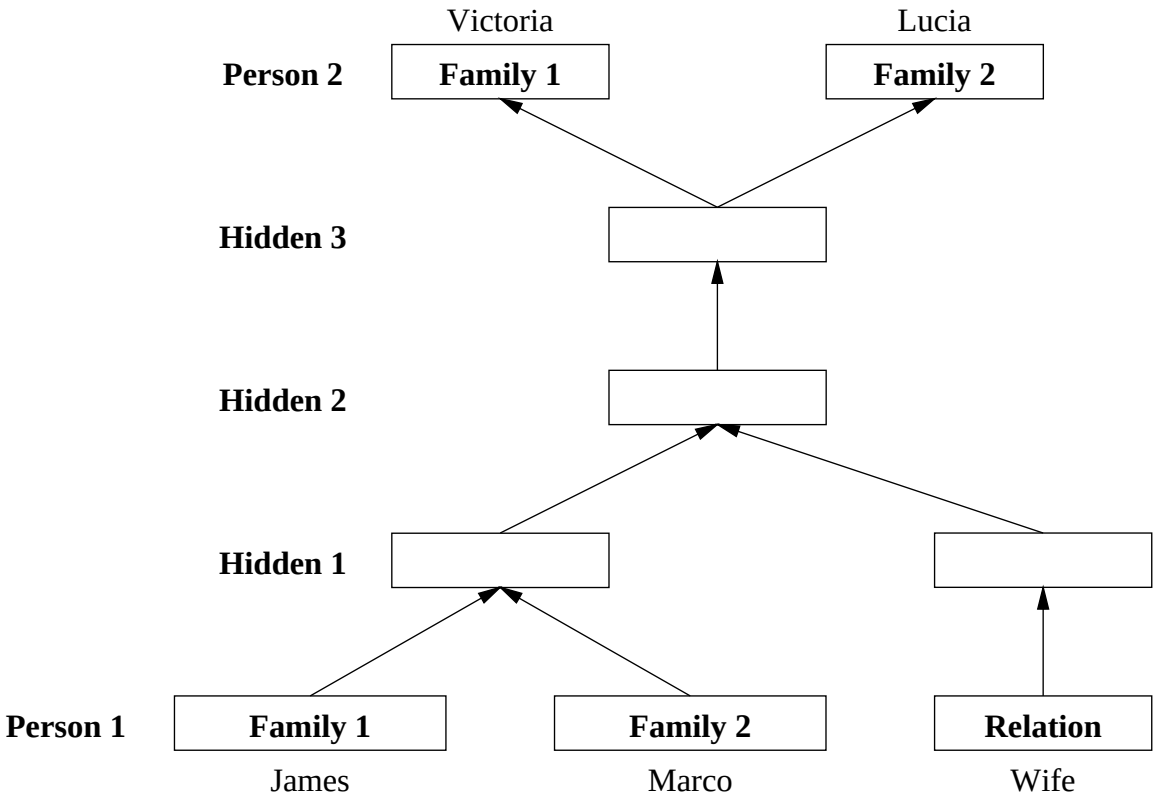
Figure Captions

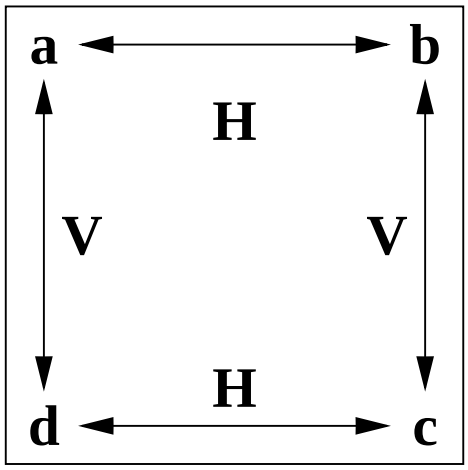
Figure 1. Feedforward network with shared weights used in "family tree" simulations (adapted from Hinton, 1990).

Figure 2. Klein 4-group with two operators (a) and task instance (b).

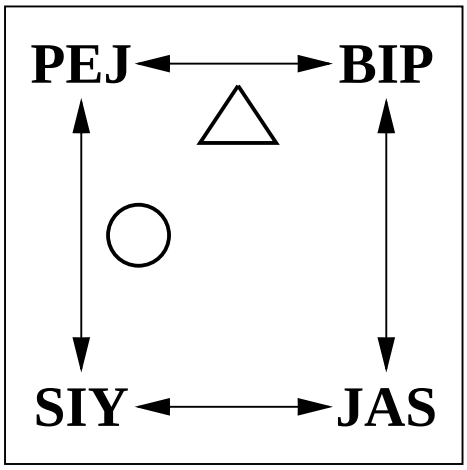
Figure 3. Feedforward network with different (a) and same (b) input/output units for each task.

Figure 4. Error surface for 7 (a) and 8 (b) patterns.

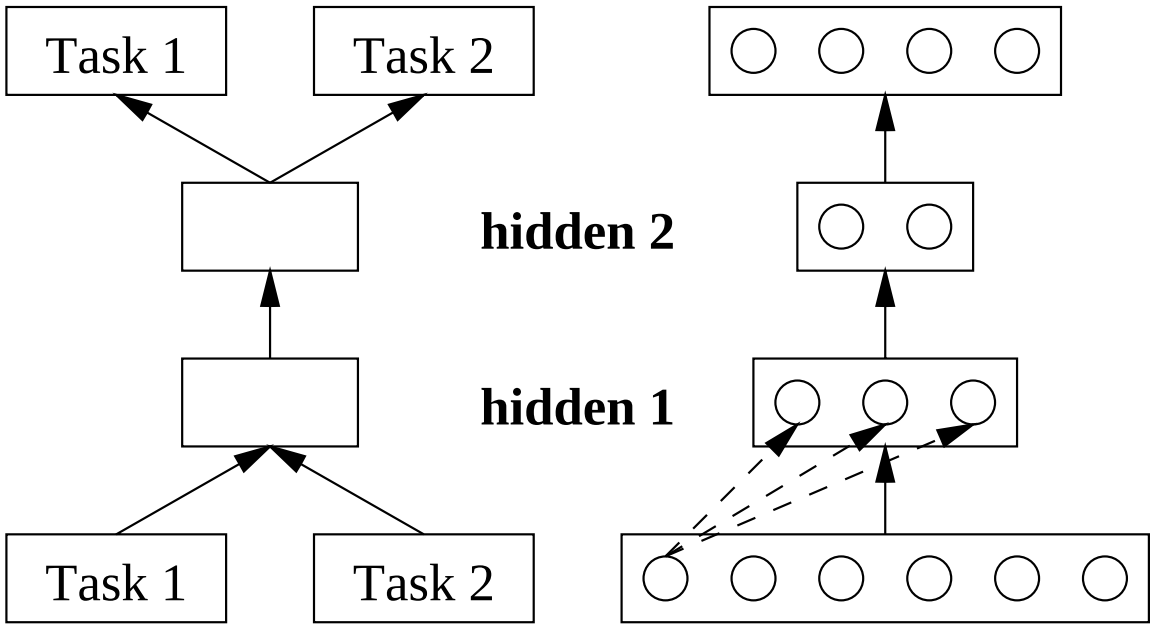

Victoria
Lucia
Person 2
Family 1
Family 2
Hidden 3
Hidden 2
Hidden 1
Person 1
Family 1
Family 2
Relation
James
Marco
Wife



(a)



(b)


Task 1
Task 2
hidden 2
hidden 1
Task 1
Task 2
(a)
(b)