Benjamin Newman
Computational Cognitive Science
Swarthmore College 2001

In Search of the Elusive "Emergent Concept"

In this paper, I explore one of the central problems facing artificial intelligence and cognitive science today — the search for models of emergent concepts. There are many models, of various kinds (and degrees of merit) of "conceptual" processes, and many models in which initially underspecified behaviors emerge from the interaction of subsystems, but there is still a gap — related, but not identical, to the gap between the symbolic and connectionist visions of AI. In this paper I will survey some of the philosophical and methodological issues surrounding the search for emergent concepts as well as exploring the growing body of relevant empirical research. I will conclude by highlighting several promising-looking avenues of further investigation.

1) What are we looking for?

Since this paper is an account of a search, it seems natural to begin with an account of the object of that search — the "emergent concept." What is an emergent concept? Simply stated, an emergent concept is a *concept* that arises as the result of a process of *emergence.* Well, then...

1.1) What is a concept?

Opinions within the cognitive science community differ as to how to define the word "concept," but, if I may begin with a tentative dictionary definition, let us define a concept as *the representation of a category.* I use the word "representation" advisedly — some connectionists claim there is no such thing, and at any rate it is just as hard a word to define as "concept." In this case, I mean that a concept is whatever stands for a category in the internal information economy of an agent. In other words, when an agent talks[1] to itself about a category, it employs a concept.

[1] In some cases this linguistic metaphor may be quite apt, but here it is intended merely to be suggestive. Most of the cognitive models discussed in this paper do not "talk" *per se*, to themselves or to anyone else.

Although this is a very preliminary definition, elaborating on it tells us a great deal about the object of our search. As part of an "information economy," a concept is an item of communication. This does not necessarily mean that *copies* of concepts are passed back and forth among an agent's subsystems, for communication can take many forms, but it does mean that there are certain roles we can expect concepts to play. Since this economy is "internal" to an agent, a concept must be used by the agent, not merely output for the appreciation of outside observers — when we discuss models that purport to demonstrate emergent concepts, this criterion will weigh heavily in our analysis.

As for "standing for a category," this raises the thorny question of *reference*. There is not room in this paper for a full discussion of this intensely relevant issue, so once again a perfunctory definition will have to suffice for now. A component of a cognitive system "stands for" a category if the activity of that component facilitates generally appropriate actions on the basis of generally correct judgements of percepts' membership in that category. These two criteria are not separate — if correct judgements of category membership do not lead to appropriate actions, whatever that may mean in the case of a given agent, it is questionable whether or not the agent is in fact making use of these judgements. If appropriate actions are based on "incorrect" judgements of membership in a given category, it is more likely that the component in question stands for some *other* category.

A concept, so described, is a tool that answers two questions for the agent using it: "Is this percept in this category," and "Given that it is, what do I do about it?" To answer the first question, a concept must incorporate a feature detector of some sort. To answer the second, it must be connected or related to other components of the agent's cognitive system, including other concepts, in such a way that the use (i.e. activation of, transmission of, output from, *etc.*) of the concept tends to lead to actions that are appropriate in the circumstances to which the concept applies. It is important to note that there is nothing wrong or even unusual about a feature detector that is the only good definition of the feature it detects — many of the sensory features we call *qualia* have this property, which accounts for the difficulty of describing them in terms of other features. Sometimes all we can say about a category *M* is that it is what an *M-detector* detects.[2] Thus, it should not be criterial of concepts that we be able to state concisely which categories they stand for — otherwise, we would never be able to recognize any agent as having a concept that we ourselves lack.

---

[2] Dennett 1991, pp. 374-383.

There is still much to be said about "concepts," and our definition as it stands will thus admit of a wide variety of possible referents. It is possible that our best use of the word "concept" would be to apply it wholeheartedly to everything that falls under the broad definition outlined above, but it is more likely that some candidates for concepthood will be *better* than others, on some more-or-less continuous scale, in ways that are not immediately clear. What we need is more data — data we will obtain in our analysis of selected recent research. Hopefully, if we investigate enough models, a *concept* of "concepts" will begin to *emerge* — which brings us back to the question...

1.2) What is emergence?

"Global effects arise from local interactions"[3] neatly sums up the idea of emergence — we call properties of a system "emergent" when they are the result of interactions among parts of the system which do not individually display those properties. There are several other relevant senses in which the word "emergent" is used, of course (we shall deal with them shortly), but for now this definition will give us enough trouble. The question of what it means for the smallest constituents of a larger system to display the same properties as the whole — thus making those properties *not* emergent in this sense — is not at all trivial.

As a case in point, consider water.[4] Macroscopic quantities of water have certain properties which are due, in various ways, to the properties of individual water molecules. Among these properties are mass, surface tension, and cohesion. Mass we can straightforwardly dismiss as a non-emergent property, for the individual molecules do each have mass, and the mass of the whole quantity of water is just the sum of these. Surface tension, on the other hand, is certainly emergent, for an individual water molecule does not meaningfully possess a "surface" — only in the aggregate does this property have any meaning whatsoever. But what of cohesion? Water *sticks to itself.* Do water molecules? What if we say, rather, that water *attracts other water* over small distances? This is certainly true of individual water molecules, and is in fact the direct cause of cohesion (and surface tension). But here the derivation of the larger-scale property from the smaller is much more complicated than in the case of mass, and our

[3] Elman *et al.* 1996, pp. 85-86.

[4] Inspired by Hofstadter and Dennett 1981, p. 144, among other sources.

intuitive familiarity with cohesion (and unfamiliarity, at least in any everyday sense, with intermolecular forces) strains the claim that these are the *same* property. So, is cohesion an emergent property of water, or is it built into the molecules?

The sense of emergence discussed above deals with properties that emerge as we pass from one *level* of description to another. Properties can also emerge over *time* — in the case of cognitive systems, we would probably call this "learning,"[5] in other contexts "development" or "evolution" might be more apt. The two senses are quite well correlated, for often properties that emerge over time do so by virtue of changes in the relationships between the components of a system, rather than in the components themselves, making them candidates for level-emergence. They are also very similar in essence, applying generally to properties that are present under one description of a system but absent from another (logically or temporally) prior description. The primary dimension of disagreement, hinted at by our uncertainties regarding the cohesiveness of water, concerns the degree of absence required. At one extreme, we can demand that emergent properties not even be *caused* by anything present in the prior description — Hofstadter[6] warns us sensibly that this way lies dualism. Somewhat less extreme is the requirement that emergent properties be absent to the extent that they are "difficult (if not impossible) to predict" from the prior description.[7] Alternatively, we could loosen the definition of emergence to cover any property whose presence in a later or higher-level description could not be inferred *straightforwardly or trivially* from a prior one. As in the case of "concepts," our ultimate understanding here must be tempered by data.

For all that "emergent behavior" is a catchphrase often wielded by connectionists against proponents (and perpetrators) of symbolic AI, emergence is not limited to continuously variable systems — discrete systems also display such properties. Consider a network of "nodes" of the following sort: Each node can be in any of several discrete states, and can send any of several discrete signals to any of its several neighboring nodes. Whenever a node receives a signal, it responds in the following discrete time step by changing state and sending a signal to a neighbor — which new state,

---

[5] See, *e.g.*, Blank *et al.* 1991, Elman *et al.* 1996.

[6] Hofstadter and Dennett 1981, p. 144.

[7] Elman *et al.* 1996, p. 44.

which signal, and which particular neighbor being determined by the node's former state and the signal it received. The large-scale and long-term behavior of this system cannot in general be predicted from the description of its numerous identical components, because the system is a Turing machine (with *active tape squares* instead of a moving head). Thus, many questions about its behavior are formally undecidable in the general case. Of course, when attempting to infer the behavior of a system from that of its components, we are not limited to explicitly formal methods, but it is important to realize that discreteness is no defense against complexity in the aggregate behavior of simple components, whether we are seeking such complexity or trying to get rid of it.

Discrete or indiscrete, the systems we are particularly concerned with are mostly *models*, which allows us to say a bit more about what it means for them to have emergent properties. When looking for a prior description at which to check that a purportedly emergent property is indeed absent, we are aided by the fact that a model necessarily has a *most prior* description, both logically and temporally. This is the model's specification, the description according to which it is constructed. It is thus the lowest-level description of the model we can need, and it describes the model in its initial configuration. Even better, since models are constructed by people, it is generally the case that this description is fairly explicit and at least somewhat comprehensible. This can greatly simplify the evaluation of the claim that a given property demonstrated by a model is emergent. However, it is important to note that, for a property to be *absent from the specification*, it is not sufficient that it be *not designed into the model*. Many characteristics of a model that were not planned out in advance can turn out in retrospect to follow quite straightforwardly (but inadvertently) from the model's lowest-level features.

1.3) So What?

Now we know approximately what we are looking for — we are looking for components or features of cognitive systems that *represent categories*, but *are not present* (or *do not represent*) under the system's *most prior relevant level* of description. Alright — why? What exactly will we do with these emergent concepts when we find them? As AI researchers, our answer to this question depends on which of our fellow subfields within cognitive science is doing the asking. To psychology, we can answer that human cognition appears to involve emergent concepts, so accurate models of human cognition must incorporate them in some form. To engineering, we can point out that systems which make use of emergent concepts may outperform other systems in real-world applications, especially

those involving adaptive optimization or mapping. To philosophy, we can suggest that emergent concepts are a necessary condition of "genuine mentality" — whatever that is (see my *other* honors paper) — and thus a necessary step on the long road to strong AI, the search for computational systems that *have* genuine mentality. Even setting aside the question of strong AI, though, these are bold claims for us to be making on behalf of these "emergent concepts" that we are, after all, still searching for. Once we have had a look at a few candidate models, we may be able to substantiate some of them.

2) Some Philosophical and Methodological Concerns:

First, though, it may help to take a step back from our definitions for a wider perspective, before diving into the intricate details of recent empirical work. In this patchwork section I will explore a few philosophical and methodological questions that, although they will be touched upon briefly in our investigation of particular models, deserve to be given a separate discussion. Philosophically, are the emergent concepts displayed by our models *real* concepts, or merely *simulated* ones? Are concepts the sort of things that could be *accurately* simulated without becoming real? (The realness of *poorly* simulated concepts is of somewhat less concern to us.) Also, how does the search for emergent concepts relate to the celebrated problem of *symbol grounding*? Are concepts and symbols the same things? How about grounding and emergence? Methodologically, how are we to interpret empirical results in this area of investigation? What does the success of a model at learning or performing a task, or at mimicking the performance of humans, tell us about its aptness as a model *of emergent concepts*? How important is it that our models be psychologically or neurologically plausible? Are emergent concepts a human phenomenon, or are we looking for something too general to be limited by faithfulness to what we already know about human cognition?

2.1) Are Simulated Concepts Really Concepts?

Colloquially, saying that you have "acquired a concept" of something is the same as saying that you have understood it. Do our models *understand* the categories they work with? Can a computational system understand *anything*? This last question should be familiar as the one famously raised by John R. Searle in "Minds, Brains, and Programs."[8] Searle emerges from his Chinese Room with a resounding

[8] Searle, 1980.

"No," but his view, which amounts to a claim that simulated concepts cannot be real because they are not grey and mushy, is not favored among AI researchers. Nevertheless, the philosophical community is still out regarding necessary and sufficient conditions for *real* concepts, understanding, mentality, *et cetera.* If there is a new consensus, it is that facts about intentionality — which are of concern to us because the relationships between concepts and the categories which they represent are intentional relationships — are facts about patterns, not essences. This is a mixed blessing for strong AI, since the absence of any *essence of intentionality* means we will never have any to sprinkle on our programs to bring them to life! In that case, we had better get on looking for models which demonstrate the kinds of sophisticated organizational principles — such as Dennett's "multiple drafts" and "Joycean machines,"[9] and Hofstadter's "fluid concepts"[10] — that must underlie human thought.

2.2) Are Concepts the Same Things as Symbols?

Symbols are "physical states which can be both nonsemantically individuated ... and reliably interpreted."[11] In other words, it is possible to tell whether or not two symbols are the same without any reference to their meanings, and yet it is also possible to associate symbols with their meanings. Can symbols be concepts? I am usually inclined to be charitable to symbolic AI, but I am afraid my answer is "No" — symbols cannot *be* concepts. However, I would caution connectionists that vectors also cannot *be* concepts. The reason for this is that concepts are fundamentally *processes*. It is a process that transforms an input into a judgement that the input is in a particular category, and it is another process, or complex of processes, that receives this judgement (in the form of a data structure) and *acts* on it. We can call these processes the *input* and *output* channels of a concept, respectively. To identify the concept solely with the place where they meet — with the data structure — is to miss most of what is going on. In the case of a classically symbolic system, of course, it is to miss *all* of what is going on, since a classical symbol all on its lonesome contains no information whatsoever, but the vector representations generated by connectionist networks are just as meaningless outside of the context of the

[9] Dennett 1991.

[10] Hofstadter and FARG 1995.

[11] Clark 1993, p. 5.

system to which they belong. In this sense, vectors *are* symbols.

This realization allows us to shed some light on the relationship between concepts and the problem of *symbol grounding.* If a symbol is the bridge between the input and output channels of a concept, what connects the symbol to other symbols and to the world, *via* perception and action? Quite simply, *concepts ground symbols*. This does not necessarily answer the question of how to ground existing (ungrounded) symbols, because where concepts are emergent, it is likely that the symbols they ground did not exist in the system before the concepts did — in other words, that the symbols grounded by emergent concepts are themselves emergent. This is of course what is intended by Plunkett *et al.*'s title, "Symbol Grounding or the Emergence of Symbols?"[12] I share with Hofstadter[13] a disdain for the notion that "symbol-groundedness" is another magical essence, beyond mere functionality, to which our models must aspire in order to become real. I am not worried about being fooled by a system that isn't *really* mental because its symbols are ungrounded, because, for exactly the reasons discussed above, ungrounded symbols can't *do* the same things as grounded ones — without a concept's input and output channels, it's unclear how a genuinely ungrounded symbol could do *anything.*

2.3) What Does a Successful Model Tell Us?

Just because a model succeeds at performing, or learning to perform, a given task, or in mimicking the performance profile or learning curve of the natural system being modeled (e.g. a 3-year-old human), we cannot necessarily infer that the model has done so using emergent concepts. In the general case, of course, this should be obvious — *most* computational models are of systems like the weather, which, although they may demonstrate a great deal of emergent behavior, are not often accused of having concepts. But, when it comes to modeling phenomena which do seem to involve emergent concepts, it's easy to leap to the conclusion that a successful model must capture this aspect of the phenomenon. At one level, this is probably true — if a system that was initially unable to perform some task in which it must discriminate between categories later learns to perform that task, it must have acquired *category discriminators that weren't there before*. However, it is possible that the representations used by the

[12] Plunkett *et al.* 1992.

[13] Hofstadter and FARG 1995, p. 290.

model *were* initially present, and are thus not emergent, but were for some reason not used. Also, if category determinations are the system's sole output, we may legitimately question whether or not the feature-detectors involved are sufficiently interrelated to count as concepts. This is particularly important where the concepts implicated in the original phenomenon are known to be strongly related to other concepts which are clearly not part of the model — for example, in models of particular linguistic phenomena,[14] where the concepts formed by the model are not comparable to their real counterparts because they are not properly connected to the (unmodeled) remainder of the language faculty.

How, then, are we to evaluate claims that a model displays emergent concepts? Since we want to know more than we can infer from overt behavioral similarities between the model and the system being modeled, we must take our models apart and see how they work. For symbolic models, this can be fairly straightforward — after all, shouldn't the researchers who constructed the model in the first place know how it works? For connectionist models (or strongly evolutionary symbolic ones), however, the model as initially specified doesn't contain any of the components in question — we are, after all, looking for *emergence.* In this case, a variety of techniques can be deployed to infer the presence of would-be concepts in the fully developed model. Data from neural networks, which can include patterns of activation across intermediate layers or patterns of input favored by particular nodes, take the form of collections of vectors, whose structure can be clarified by cluster analysis or principal components analysis. However, it is important to realize that the analysis is not part of the model *per se*, and thus that the features of the analysis (e.g. internal nodes in a cluster tree) are not themselves the concepts that we are looking for — merely guides to the ways in which those concepts may be interrelated.

2.4) Should Our Models Always Model Us?

In a previous section, I mentioned three other fields with close ties to computational cognitive science — engineering, psychology, and philosophy — and how our relationship to them colors our goals as AI researchers. Based on this, we can discern three overlapping movements within the AI community — respectively, *intelligence engineering*, which is concerned with the creation of expert systems which excel at tasks, regardless of the human-likeness or mentality of the mechanisms involved, *cognitive modeling*, which is concerned with faithfully modeling the processes behind human cognition

[14] See Elman *et al.* 1996, pp.130-147

so as to better understand our own minds, and *pure strong AI*, which is concerned with the creation of systems for which we can claim genuine mentality, whether or not they are good at anything in particular *or* at all like us (very few projects fall solely under this last heading, but strong AI does remain a distinct goal for some portion of the research community). In light of these distinctions, the above question can be rephrased in the following form: "Is cognitive modeling *more important* than intelligence engineering or pure strong AI?"

Personally, I am ambivalent regarding the answer to this question. At one level, the answer is clearly "Yes" — cognitive modeling represents our contribution to a well-defined and respected scientific inquiry, the study of the human mind. It has the potential to answer specific questions about something (the human mind) that already exists, and in which we are all immensely interested, in contrast to the often narrow and esoteric scopes of individual projects in intelligence engineering or the abstruse game played by proponents of pure strong AI with their philosophical critics. Thus, I would grant that cognitive modeling is *more important* as a goal for AI research. Nevertheless, the demand that models be faithful to what is known about the human mind and brain can be taken too far — in particular, I think that to hold any particular degree of human-likeness as strictly criterial for concepts can lead us to underestimate the significance of many models without effectively focusing our search for emergent concepts onto models in which they are more likely to be found. Also, without significant further advances in connectionist techniques, it may be difficult to design models that are *both* psychologically and neurologically plausible.

Looking at the question another way, the first modern thinkers to seriously consider the nature of cognition saw the human mind as very generic — so generic that a thorough analysis of it would reveal the abstract idea of *thought* in its full generality, without spurious human-specific details. In this context, the expectation that models (at that point, philosophical rather than computational) should be faithful to what was known about cognition *in humans* would not have seemed very restrictive. Now, we understand that human nature is quite specific — so specific that a model scrupulously tailored to remain faithful to it may be of such narrow relevance as to shed no light *whatsoever* on the abstract notion of "thought." Both of these views remain current in the philosophical community, and, to varying degrees, in other academic communities as well. Thus, our task is to keep each model faithful to human thought at a level of specificity which is appropriate to the goals of the particular project. The task of determining this appropriate level will be one of our concerns as we move on to our...

3) Survey of Empirical Work on Emergent Concepts:

There are a wide variety of projects which might be considered relevant to the search for emergent concepts, ranging from self-organizing adaptive neural maps to quasi-parallel symbolic models of creative analogy-formation. The survey that follows is in no way exhaustive of current work, merely somewhat representative. I have divided the models to be considered into three groups. The first group consists of self-organizing neural networks, of several sorts. The second group features models created by the Fluid Analogies Research Group (at Indiana University under Douglas Hofstadter). The program architecture pioneered by FARG is sufficiently different from anything else to merit its own section in this paper. The third group of models consists of control systems for robots (real or simulated). This work is the most recent.

Within each group, I will describe the models discussed, largely summarizing from the models' creators' own articles. I will then apply the criteria established in the preceding sections, asking whether or not the models in that group display concepts, and whether or not these concepts are emergent, while highlighting the advantages and disadvantages of each approach.

3.1) Self-Organizing Neural Networks:

The term "self-organizing" has as many meanings as "emergent." Here, it means two things. Any neural network will discover, in the course of training, ways of mapping its input from one vector space to another — of *organizing* it — that are efficient and meaningful from the point of view of the problem the network learns to solve. In this sense, all neural networks are self organizing. Here, though, the "self" in "self-organizing" is taken to imply an unsupervised mode of learning. None of the networks considered in this section required any information during training that was not given as input.

3.1.1) Auto-Associative Networks:

When working with feed-forward backpropagation networks, one way to eliminate the teaching signal is to make it the same as the input. Viewed like any other task for a feed-forward network, as an input-output mapping to be learned, this does not seem to be a very interesting problem. However, if the network to be trained has a hidden layer that is substantially smaller than the input to be reproduced, a new and interesting problem is revealed — how to compress the input into a smaller representation without losing any of it.

The simplest form of auto-associative neural network model does nothing but compress and decompress its input. The success of the network in reproducing its input shows how compressible the input data are, and an examination of the network's weights and hidden-layer activations can reveal just which regularities in the input data were exploited. However, when this is all that a network is asked to do, it is hard to argue that it displays emergent concepts — that a collection of *n*-vectors all lie in a (possibly not quite flat) subspace of dimension *(n - k)* is not what most of us have in mind when we think of "conceptual knowledge." Luckily, the idea of auto-associative networks has been extended in a number of ways which are more promising in terms of the search for emergent concepts.

The model of Plunkett *et al.*[15] used a four-layer auto-associative network to associate images with labels. It had separate banks of input nodes for images and for labels, leading to separate second layers. These two channels combined in the third layer, which was then connected to two banks of output nodes. Each *image* was a sparse arrangement of points generated from one of several prototype images by displacing some of its points. Each prototype image was associated with an arbitrary localist *label*. The network's ultimate task was to learn the mapping between labels and prototypes through repeated exposure to the labels in conjunction with the distorted images — the network was never exposed to the prototypes themselves during training. However, the network was never asked to produce a label given only an image, or *vice versa*, during training — instead, for each image-label pair in the training set, the network was asked to associate the label with itself, the image with itself, and the complete pair with itself, in rapid succession, before moving on to the next pair.

The network was tested in both *production* — producing the correct label given an image (including prototype images which had not been presented to the network during training) and *comprehension* — producing an appropriate image given a label. The network's production performance was higher on the prototype images than on the distorted images, even though the prototypes were not included in the training set, and it learned comprehension more quickly and smoothly than production. Also, the learning curves for both tasks show a pronounced growth spurt. Plunkett *et al*. compare these results with similar findings regarding category judgements of random dot patterns in humans (essentially the same task), and to patterns of vocabulary growth in children — in both cases the similarities are quite

[15]Plunkett *et al.* 1992.

striking. Could this simple neural network have captured the essential features of the mechanisms by which we learn to name things? More to the point, does it demonstrate emergent concepts?

That the network learned to recognize the prototype images doesn't necessarily show that the category-representations it formed were prototype-based, but it does show that the network was able to represent each category in a unified way, and to develop a mechanism for deciding which category a given image belonged to — a key part of our definition of "concept." Despite the fact that the localist labels were given in advance, the concepts which allowed the network to form a unified representation of an image and a label together clearly emerged during training. The similarity of the network's learning profile to human data is an added bonus from a cognitive modeling point of view, but any claim that the concepts themselves were similar in structure to human concepts would be premature, especially when considering the linguistic character of the human tasks to which the network's task was compared.

RAAM (Recursive Auto-Associative Memory)[16] is another innovative extension of the idea of auto-associative neural networks, developed as an answer to the criticism that connectionist systems were not equipped to deal with recursive data structures. A RAAM architecture is defined by a fixed representation size n and branching factor k. The network that is trained has n nodes in its hidden layer and kn nodes in its input and output layers. Thus, it learns to compress k-tuples of n-vectors into a single n-vector (at the hidden layer), which can then be re-input to the network as part of another group of k vectors. If the network is able to faithfully decompress the representations formed on the hidden layer — a somewhat chancy proposition for high values of k — it has thereby developed a fixed-size representation for a class of recursive structures, which can then be used by other neural networks.

Initially, the training data will only include k-tuples of the primitives of the class of recursive structures being learned. As the network learns to auto-associate these k-tuples, it will produce compressed representations of them which must then be included in the training data. Thus, the training set grows as the network learns, and drifts as the network's weights change — this necessitates careful management of the training set and tuning of the parameters used in backpropagation. When training is complete, the network is then metaphorically "sliced in half." The bottom half of the network, from the input layer to the middle layer, can now be used recursively to encode tree-like data structures as vector

[16] Pollock 1990.

representations, while the top half can be used to decode them — it is important to be able to distinguish primitives from compressed *k*-tuples so that the decoding process can stop, of course. An alternative version, sequential RAAM, represents sequences by composing primitives onto an initially "empty" compressed representation, analogously to LISP lists. Here, the compressed sequences do not have to be the same size as the primitive data.

Given the high degree of compression involved, a RAAM should only be able to reliably encode and decode a small fraction of possible input patterns. Pollock found that the most common error was the decoding of a novel structure as a structure that had been presented during training. However, when trained to represent a non-arbitrary subset of possible combinations of primitives (for example, words as sequences of letters, or grammatical parse trees), Pollock's RAAMs learned to represent valid novel structures better than invalid ones, showing that neural networks are capable of learning grammar-like classifications of recursive data structures — a task that was once considered the exclusive domain of symbolic systems. Even more intriguingly, Blank *et al.*[17] showed that other neural networks can be trained to perform syntax-related tasks on a RAAM's compressed representations *holistically*, without decoding them (the networks used for these tasks did use supervised learning), suggesting a variety of roles for RAAMs as components in future models.

Does a RAAM develop emergent concepts? Certainly, it develops emergent representations, and it implicitly represents a category of valid structures — those it will encode and decode faithfully. Yet, the best case for the concepthood of a RAAM's representational machinery is that, because of their recursive character, the structures it represents are related to one another in a rich variety of ways, beyond simple similarity. Richness of interrelatedness is an important feature of concepts as used by humans, and it is likely to be an essential feature of concepts wherever they appear.

3.1.2) Kohonen's Self-Organizing Feature Maps:

The self-organizing feature maps described by Kohonen[18] are a completely different kind of neural network architecture. A feature map consists of an input layer of nodes connected to a special

---

[17] Blank *et al.* 1991.

[18] Kohonen 1989, chapter 5.

map layer. The nodes in the map layer are associated with a lattice of points in a metric space, such that there is a well-defined distance between any two of them. The map nodes are also connected to one another — these connections are excitatory for short distances, inhibitory for middle distances, fading to zero at longer distances. When a pattern is input to the network, each node in the map layer is activated to a greater or lesser extent, depending on the weights of the connections between it and the input layer. These activations are then redistributed by the connections within the map layer, ultimately converging to a small cluster of activated nodes. These nodes are then trained to prefer the given input pattern, while nodes within a certain distance are similarly trained, but less strongly. The process is repeated for the next input pattern.

The fact that nodes which are nearby in the map are trained together means that the function from the space of possible input patterns to the space in which the map nodes are located tends towards continuity. Even better, the map tends to retain those dimensions of the input space which contain the most variation, but on a *local* basis, and it tends to conform to the input space such that the density of map nodes covering a given region of the input space is proportional to the probability of a random input falling in that region. Kohonen's analysis suggests that maps of this type may be good models of some kinds of lower-level perceptual capacities in the brain, both functionally and structurally.

Do Kohonen maps exhibit emergent concepts? The function of a Kohonen map is to transform a distributed representation into a localist representation — a single activated map node. These localist representations, however, are related by their association with points in the map *space*, which plays a vital role in the training of the network. Connected regions of the map space could be thought of as representing the categories of inputs that mapped into them — these categories would be related to each other by the topology of the map space. But, then what?

3.1.3) Pros and Cons: Then What?

A common drawback of models whose primary task is to *create* representations is that, by themselves, they seldom make *use* of them. Self-organizing neural networks are especially prone to this shortcoming. After a RAAM has compressed a recursive structure into a vector, or a Kohonen network has located an input pattern on its map, then what? How are these representations to be used? In the language of our process description of concepts, we would say that the concepts which emerge from these models are a bit weak on the output end. In order to qualify as holders of fully-fledged concepts,

these models must be integrated with a system that uses their concepts to facilitate some kind of action. The image-label association performed by Plunkett *et al.*'s network could qualify, but here we have concepts taken one-at-a-time, with no clear interrelationships. What we need is a model that makes sophisticated and flexible *use* of its concepts, deploying them several-at-a-time to describe complex situations, and responding to those situations on the basis of their descriptions.

3.2) The FARG Projects:

This last is a very tall order, but a number of projects from Hofstadter's Fluid Analogies Research Group[19] go a long way towards filling it. These projects are symbolic, not connectionist, in terms of their representational strategies, but they are also parallel and nondeterministic, and their implementation of concepts involves activation spreading through a network. Out of the variety of FARG projects, I will focus on *Copycat,* both because I am most familiar with it and because it best encapsulates the notion of *fluid* concepts which is the core of FARG's philosophy.

3.2.1) The Architecture of Copycat:

Copycat[20] is a model of analogy-making (which FARG views, not as a peripheral subfield of problem-solving, but as a fundamental part of conceptual and creative thought). Copycat in particular deals with analogies in a restricted domain of letter-strings, solving analogy problems of the form "If **abc** becomes **abd**, what does **ijk** become?" Although Copycat's knowledge is limited — it can recognize groups of letters that are identical or in alphabetical order, and can count to 5 — some of the problems that it can solve are quite subtle, and admit of answers which are clever and not at all obvious.

Copycat represents a problem symbolically, as a collection of letter *tokens* organized into strings in its *Workspace*. Each is identified as a particular letter by a link to that letter's node in the *Slipnet*, which is where the model's conceptual knowledge resides. Over the course of a run, Copycat will build a structure on each string by seeing letters as organized into groups (which may be nested), and then will construct a mapping between the first and third strings by identifying components of one with

---

[19]Hofstadter and FARG 1995.

[20]Mitchell 1990.

components of the other. Every structure in the workspace is labeled by links to one or more nodes of the Slipnet. All processing is performed by small procedural objects called *codelets*, several of which must run in succession to perform even the simplest action in the Workspace. Codelets are chosen at random from a pool called the *Coderack*, so that many processes can be in progress at a given time, in effect taking turns as codelets contributing to each process are chosen and carried out. By adjusting the probabilities that particular codelets will be chosen — their *urgencies* — the architecture allows those processes which seem more promising to proceed more quickly, resulting in a *parallel, terraced scan*. The degree of organization in the Workspace is tracked by the *temperature*, which in turn regulates the parallel, terraced scan by controlling the degree of randomness in the Coderack as well as the model's willingness to let the structures in the Workspace be destroyed.

When it is ready, the system will use the mapping between the first and third strings to translate an explicit representation of the difference between the first and second strings. This translated rule is then applied to the third string to yield a fourth string, which is returned as the program's answer. If the rule cannot be translated, or the translated rule cannot be applied, extra attention is focused on the source of the problem in an attempt to build a new representation which will lead to a solution. However, Copycat is not very self-aware, and can run up against the same impasse many times in a row, a problem addressed by the sequel program *Metacat*,[21] which adds to Copycat a greater degree of self-monitoring while remaining quite similar to its predecessor in terms of basic architectural features. Although no thorough study of human performance on analogy problems of this type has been carried out, Copycat's performance has been compared with humans' informally, with encouraging results. Solutions which humans consider obvious, Copycat finds with high frequencies, while obscure solutions are found rarely. On the other hand, the temperature at the end of a run, indicating the degree of structure in the Workspace, corresponds well with human judgements of how elegant a particular solution is.

3.2.2) Looking for Concepts in the Slipnet

The Slipnet is the locus of Copycat's conceptual knowledge, and, in the developers' terminology, each node of it is identified with a "concept." These include the letter-types of which the individual letters in a problem are tokens, the successorship relationship which defines the topology of the

[21] Marshall 1999.

alphabet, the two kinds of groups (successorship and sameness), ways of identifying the position of a component within a string, and abstract relationships between other concepts, such as oppositeness. Concepts are linked to each other, and activation, which represents how relevant a particular concept seems to the situation at hand, can flow along these links. Additionally, when a mapping is made, a concept does not always have to be mapped to itself — it can be mapped to another concept that is near it in the Slipnet. This is known as a *slippage.* Because of slippage, the creators of Copycat urge us to think of concepts, not as identified with single nodes, but as spread over a cluster of connected nodes. Is this the best way to think of the relationship between concepts and Slipnet nodes?

In light of our process view of concepts, it may be more useful to us to think of Copycat's concepts as more than just neighborhoods in the Slipnet. Rather, each concept also directly involves the codelets that make judgements of category membership by creating structures in the Workspace and linking them to nodes in the Slipnet, as well as those which take further actions based on these judgements of category membership. In this view, nearly every part of the program is constitutive of one or more concepts, not just the Slipnet, which defines the relationships between them. Copycat's concepts are thus clearly emergent in the structural sense of arising from the interaction of smaller components. However, since the structure of the Slipnet and the kinds of possible codelets are all set in advance, Copycat's concepts aren't emergent at all in the temporal sense of being learned — a common shortcoming of symbolic systems.

Despite the absence of learning, Copycat is a *tour de force* when it comes to concepts, because it demonstrates so many of the features that make concepts interesting in humans. Copycat's concepts are richly connected to one another in many different ways, and include supercategories and subcategories as well as metaconcepts and relations — concepts that apply to the connections between other concepts. These higher-level concepts are particularly hard to learn using current connectionist techniques, and thus good targets for a predesigned symbolic model. In addition, although Copycat's concepts are judges of category membership as per our definition, they also satisfy the other portions of that definition, which require that concepts play a role in further action *beyond* mere categorization judgements. When an object in Copycat's Workspace is labeled with a link to a particular concept, this profoundly affects how it will be treated in subsequent processing — and because activity in the Workspace is significantly nondeterministic, this sensitive dependence on its *ad hoc* representations allows Copycat to return a variety of responses to a given problem.

3.2.3) Can Learning be Added to the FARG Framework?

It is interesting to consider whether or not a model like Copycat could be adapted into a more developmental model. Clark claims that learning in symbolic systems is necessarily weaker than in connectionist systems because a symbolic system, with its fixed repertoire of primitive symbols, cannot expand its representational capacities in any fundamental way.[22] However, this is true of any system! A connectionist network cannot transform its input into just any representation — it is limited to vectors, and in most networks these must be of a certain fixed length. If a symbolic system were provided (innately) with connectives that gave it a sufficiently generative conceptual grammar, it might be able to build up something like Copycat's Slipnet from scratch. A more significant obstacle than the inability of symbolic systems to outgrow their representational schemas is the fact that we have no idea how highly abstract metaconcepts like "oppositeness" are learned by humans. Thus, any attempt to adapt FARG's architecture into a model of learning concepts from scratch must aim much lower than Copycat's rich domain.

My own work, which unfortunately did not pan out, was on a program based on the FARG architecture called Kitten (as an infant relative of Copycat) for categorizing strings over a very small alphabet (just **x** and **o** in its last incarnation before it was abandoned). However, I spent a great deal of time brainstorming possible representational systems for Kitten, and thus developing a sense of what must go into a symbolic model of concept learning (and why we don't see more of them).

Kitten's represents a linear string of symbols as a binary tree, just as Copycat represents a string as being made up of groups of various types. More than one tree is be built up and analysed at once, nondeterministically, so a tree may have many trees above it that claim it as a subtree — conflicts to be resolved at a later stage of processing. To each subtree is attached a *literal description*, which is just a copy of the tree, built up out of primitive symbol descriptions (**x** and **o**) combined using the LISP **cons** operator, which creates ordered pairs. However, as the trees are being built up by one family of codelets, another type of codelets is comparing Workspace structures with one another. When two literal descriptions are found to be indentical, they are combined into one description. This is the first level of abstraction — recognizing that different instances of the same pattern are the same. At this point Kitten's representational language contains only **x**, **o**, and **cons**.

---

[22] Clark 1993, p. 14.

The next level of abstraction introduces the possibility of mappings that slip, and with them several new terms for Kitten's language. Now, instead of mapping only whole tree to whole tree, a number of other, *slipped* mappings will also be attempted, in simulated parallel. The whole of one tree can map to just one branch of another — its **car** or **cdr**. Or, tree can be mapped to tree *backwards*, **car** to **cdr** and **cdr** to **car**, a change summed up neatly by the new unary operator **flip.** (This operator is in fact recursive, reversing every subtree all the way down. Otherwise, it's a **flop**, not a flip. The reason the flip operator is preferred is because a flop can be expressed by three flips, while expressing a flip in terms of flops requires one at every internal node.) Finally, the mapping between two trees can be reversed alphabetically, exchanging **x**'s for **o**'s with the recursive **not** operator (this is the reason the alphabet was ultimately reduced to two symbols). Whenever two descriptions are mapped together with some slippage, a link is built between them, labeled according to the type of slippage involved. These links are the first threads in Kitten's makeshift Slipnet.

The third and final level of abstraction introduces templates with free variables and the possibility of recursive structures. I'm still not sure of the exact circumstances under which these would be created in the workspace — in fact, this was the final impasse that lead to my abandoning the project. However, the representational scheme itself is not that complicated. The operator **lambda** identifies a description as a *template*, which must contain one or more variables. When a template is mapped to another structure by itself, variables act as wild cards — they match anything. However, when a template is *applied* to one or more other descriptions using the **alpha** operator, the resulting description requires that subtrees which map to the variables in the template match the corresponding one of the descriptions to which the template was applied. The **beta** operator can be used to define recursive classes of structures. If **T** is a template and **U** is a description, **(beta T U)** matches **U** *or* **(T (beta T U))**. Thus, **U** is the *base case* of the class of structures so defined ("beta" stands for "base").

The representational schema defined above is quite versatile — the lack of anything resembling a conditional operator means it is not universal. However, a symbolic model needs more than a language for its representations. It needs procedures. The primary difficulties with Kitten's design were not in the design of the language in the abstract. Rather, they were in determining how and in which contexts the various terms of the language were to be introduced into the Workspace (e.g. what kinds of attempts at mapping together unlike descriptions might lead to the creation of a **lambda**-template), and what links would be maintained among the items in the Workspace. In other words, it was relatively easy to design

the front ends of the concepts that would ground the symbols of Kitten's representational language, to specify which simple trees of **x**'s and **o**'s would be matched by a given description. The difficulty was in setting out what would be done subsequently, based on such a match, which is to say in designing the output ends of Kitten's concepts.

3.3) Robotic Control Architectures:

A number of recent projects involve the bottom-up development of control architectures for (real or simulated) robots. A model in which a robot builds structured representations of its environment or its actions based on sensory data would be a very good example of emergent concepts. This section describes two very different models which meet, or will soon be extended to meet, this criterion.

3.3.1) Clustering Experiences:

Oates and Cohen[23] describe a number of experiments in which time series of data recorded from a robot's sensors are analyzed and placed into categories which correlate well with qualitative descriptions of the events which they represent. A robot (guided by an unspecified control architecture) wanders around a room recording the data from its several sensors, including sonar proximity detectors. The sensor data was then segmented (by hand) into short time series corresponding to describable events, such as "passing an object on the left." These segments were then organized into categories by cluster analysis — the primary innovation being the use of a time-stretching metric in clustering, so that time series depicting the same events taking place at different speeds could be marked as similar to each other. The results of the cluster analysis were largely coextensive with the authors' by-hand organization of the segments into categories.

Oates and Cohen recognize that, no matter how interesting a structure is found on the segmented sensor data, it isn't doing the robot any good yet, because it isn't yet used in controlling the robot! They are concerned that, however useful they may be, the prototypes arrived at by averaging over the segments in each cluster are not compositional — in other words, there is a prototype that can be used to identify the event of passing an object on the left, but there is no way to modify it so that it can indicate *which* object is being passed on the left. For this, the authors are also testing a completely

[23] Oates *et al.* 2000, Cohen 2000.

different system, which uses the data from a camera to generate a running commentary in a symbolic format similar to the predicate calculus. It is unclear whether or not they plan to integrate these two systems, or how they could be integrated. For either of these two systems, however, the next step is to incorporate the representations created into the robot's control system, perhaps as part of a planning facility geared towards performing some specific task. Until then, it is unclear whether the "concepts" produced can be said to belong to the robot at all.

3.3.2) The CLARION Model:

Although never used to control an actual robot, CLARION[24] models the learning of strategies for action in a real-time domain with noisy and limited available sensory data — constraints that make it applicable for use in robots in the future. CLARION learns both low-level procedural knowledge and high-level declarative knowledge, and is relatively unique among such models in that it learns its procedural knowledge first, then abstracts it to declarative knowledge. It uses a form of reinforcement learning, in which adaptation in a neural network is based on a scalar teaching signal which can be interpreted as a continuum of reward and punishment (or pleasure and pain). The model was trained to navigate through a minefield to a target.

The CLARION system consists of two neural networks. One is a standard feedforward network that is trained to predict, given the system's current state and a proposed action, the Q-value of that course of action (Q-value is a time-discounted aggregate of expected future reward[25]). The network learned the Q-values of particular state-action pairs inductively — the values returned by the network got closer and closer to the actual expected values over the course of learning. The action returned by the network was chosen randomly, with probability proportional to Q-value. By contrast, the second network used a localist representation of states and actions and a small number of high-valued links to encode discrete rules prescribing particular actions in particular states. A rule leading to a bad outcome was specified to exclude the conditions under which it did so, while a rule leading to a good outcome was generalized to apply under more circumstances. The action chosen by this network was the one

---

[24]Sun and Peterson 1998, Sun and Merril 2001.

[25]Watkins, C., *Learning with Delayed Rewards*, Cambridge University, UK, 1989. Cited in Sun and Merril 2001.

recommended by the most specific rule that applied in a given state. The action actually performed, however, was chosen based on a weighted sum of the scores given to each action by the two networks.

Although the models' creators do not specifically talk about "concepts" in their analysis, there are clearly concepts involved — the model learns categories of sensor states and uses this categorization to choose an action to perform in real time. The structure of the model's representational space is not discussed specifically — it would be interesting to find out if the two levels of the model developed distinct but closely coupled conceptual schemes. What we would like to see is a model that incorporates aspects of CLARION and of Oates and Cohen's work, generating categories of percepts and using them to reason about possible courses of action in pursuit of a goal. There is room for a wide variety of models of this kind, and — given how recent the above cited work is — we can expect to see more of them in the near future.

4) Conclusions: Where Do We Go from Here?

We began by attempting to define "emergent concepts," deciding that concepts are components of a cognitive system that represent categories by making category membership judgements and communicating those judgements to other components where they could help to direct the system's subsequent activities, and that properties are emergent when they are present in a system in its final state or under its highest level of description but absent from the system's initial state or most basic level of description. We considered a number of philosophical and methodological issues that affect our search. Finally, we looked at some actual models of emergent concepts.

Here we found that self-organizing neural networks are good models of emergent representations, but poor models of concepts, because the representations they learn are not used in sophisticated ways by any subsequent processing. We saw that artfully crafted symbolic systems could capture aspects of conceptual thought that are beyond the reach of connectionist networks — for now — but that these systems required hand-crafted concepts to an extent that makes training up such a system from scratch a distant prospect indeed. And we looked at a variety of ways in which concepts can be discovered in and incorporated into the control systems of robots. What have we learned?

One criterion that was mentioned in our definition, but found new emphasis in our survey of actual models, was the fact that concepts, to be concepts, must *do* something — mere feature-detection is not enough. Thus, we cannot expect just any cognitive model to display concepts. Models, to have

concepts, must be *agents* — systems that act — so that they can use their concepts to guide their actions. This is particularly urgent in the case of our self-organizing neural networks, which show few aspects of agenthood. What remains to be done?

We have yet to see a symbolic system that learns a sophisticated range of concepts from scratch and applies them flexibly — given Clark's warnings about the rigidity of symbolic representational schemas, it is unclear exactly what would have to be built into such a system. We have yet to see a model that uses a self-organizing neural network as one component of a combined architecture, putting its emergent representations into action by giving them something to *do*. Recent developments in robot control are encouraging, but some of the most interesting conceptual schemes are not actually used by the robots that generate them, and one of the most promising control architectures has never been tested in a real robot. Clearly, there is a lot more to do.

Many paths lie open, and the choices are ours — which architectures to pursue, and which to ignore, how faithfully to aim our models at specifically human cognitive phenomena rather than the more abstract goals of pure AI, which tasks to place before our agents, and so on. The idea of "emergent concepts" is broad, and a fertile ground for new projects, many of which we can only begin to imagine now. One thing, however, is certain — the search for the elusive "emergent concept" is far from over.

Bibliography

Blank, D. S., *et al*., "Exploring the Symbolic/Subsymbolic Continuum: A Case Study of RAAM." Indiana University Computer Science Department, 1991.

Clark, A., *Associative Engines: Connectionism, Concepts, and Representational Change*. MIT Press, 1993.

Cohen, P., "Learning Concepts by Interaction." University of Massachusetts Computer Science Department technical report, 2000.

Denett, D. C., *Consciousness Explained*. Back Bay Books, Boston, 1991.

Elman, J. L., *et al*., *Rethinking Innateness*. MIT Press, 1996.

Hofstadter, D. R., and FARG, *Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought*. Basic Books, NY, 1995.

Hofstadter, D. R., and Dennett, D. C., *The Mind's I: Fantasies and Reflections on Self and Soul*. Basic Books, NY, 1981.

Kohonen, T., *Self-Organization and Associative Memory*, Third Edition. Springer-Verlag, 1989.

Marshall, J. B., *Metacat: A Self-Watching Cognitive Architecture for Analogy-Making and High-Level Perception*. Indiana University, 1999.

Mitchell, M., *Copycat: A Computer Model of High-Level Perception and Conceptual Slippage in Analogy-Making.* University of Michigan, 1990.

Oates, T., *et al*., "A Method for Clustering the Experiences of a Mobile Robot that Accords with Human Judgements." *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, 2000.

Plunkett, K., *et al.*, "Symbol Grounding or the Emergence of Symbols? Vocabulary Growth in Children and a Connectionist Net." *Connection Science* 4:3-4, 1992.

Pollock, J. B., "Recursive Distributed Representations." *Artificial Intelligence*, 46:1, 1990.

Searle, J. R., "Minds, Brains, and Programs." *Behavioral and Brain Sciences*, vol. 3, Cambridge University Press, 1980.

Sun, R., and Peterson, T., "A Subsymbolic & Symbolic Model for Learning Sequential Navigation." *Proceedings of the Fifth International Conference of the Society for Adaptive Behavior*, 1998.

Sun, R., and Merril, E., "From Implicit Skills to Explicit Knowledge: A Bottom-Up Model of Skill Learning." *Cognitive Science*, 25:2, 2001 (in press).