A Modular Neural Network Model of Concept Acquisition

PHILIPPE G. SCHYNS
Brown University

Previous neural network models of concept learning were mainly implemented with supervised learning schemes. However, studies of human conceptual memory have shown that concepts may be learned without a teacher who provides the category name to associate with exemplars. A modular neural network architecture that realizes concept acquisition through two functionally distinct operations, categorizing and naming, is proposed as an alternative. An unsupervised algorithm realizes the categorizing module by constructing representations of categories compatible with prototype theory. The naming module associates category names to the output of the categorizing module in a supervised mode. In such a modular architecture, the interface between the modules can be conceived of as an "information relay" that encodes, constrains, and propagates important information. Five experiments were conducted to analyze the relationships among internal conceptual codes and simple conceptual and lexical development. The first two experiments show a prototype effect and illustrate some basic characteristics of the system. The third experiment presents a bottom-up model of the narrowing down of children's early lexical categories that honors mutual exclusivity. The fourth experiment introduces top-down constraints on conceptual coding. The fifth experiment exhibits how hierarchical relationships between concepts are learned by the architecture, and also demonstrates how a spectrum of conceptual expertise may gradually emerge as a consequence of experiencing more with certain categories than with others.

Funes was, let us not forget, almost incapable of general, platonic ideas. It was not only difficult for him that the general term dog embraced so many unlike specimen of different size and different forms... His own face in the mirror, his own hands surprised him on every occasion... I think that he was not very capable of thoughts. To think is to forget a difference, to generalize, to abstract. In the overly replete world of Funes, there were nothing but details, almost contiguous details. (Borges, 1962, p. 114)

This work was supported in part by Grant NH 41704 from the National Institute of Mental Health. I would like to thank James Anderson, Terry Au, James McClelland, Gregory Murphy, and reviewers for helpful comments on the manuscript. The computer simulations were run on the Stellar GS1000 mini-supercomputer of the Department of Cognitive and Linguistic Science, Brown University.

Correspondence and requests for reprints should be sent to Philippe G. Schyns, Department of Cognitive and Linguistic Science, Box 1978, Brown University, Providence, RI 02912.
"Without concepts, mental life would be chaotic" (Smith & Medin, 1981, p. 1). This often-cited quotation reflects what most researchers in cognitive science think of the essential role played by concepts in cognition. Concepts are so fundamental to our own functioning that the notion of a thought without concepts is hardly conceivable. Basically, concepts enable an organism to produce similar responses to classes of similar objects, events, or situations. Put differently, concepts reduce the enormous amount of variation in the physical world to a small number of categories. Needless to say, these abstractions represent a considerable adaptive advantage, for the interaction of an organism with its environment will succeed at a low cost for its cognitive resources. However, because the world we perceive is filtered by our concepts, concepts also constrain the content of our thoughts. By analogy, concepts are the chisels of the mind, shaping the mind's representation of the world as chisels would carve a rock. The finer the chisels, the more the sculpture can reflect the complexity of reality.

This article describes a neural network model that develops a conceptual representation, and the subsequent constraints the representation imposes on simple lexical acquisition. Almost all existing neural network models of concept learning assume that a concept is learned through the repetitive association of a category name with instances of the category (see McClelland & Rumelhart, 1985, for a counterexample). Each instance is characterized by a set of properties, and this representation is associated with the correct category term. Considered as a general framework of conceptual development, there are three problems with this sort of approach. First, it assumes that concept acquisition relies heavily on a teacher who provides feedback as to the exemplar's category. As will be shown later, this assumption is false as a general principle of conceptual knowledge acquisition. Second, this view assumes that language is a necessary condition to learn concepts. This assumption also appears to be contradicted by psychological data. Third, with this approach, the instances the network has to categorize are represented with the set of properties prescribed by the modeler. Thereby, one of the most fundamental problems of concept acquisition is sidestepped: How do we determine which properties, features, or attributes are relevant to each concept (Murphy & Medin, 1985)?

This article is divided into three parts. Starting with virtually no initial knowledge and a self-organizing learning mechanism, I will explore how a system can construct a dynamic representation of simple categories. By assuming so little initial knowledge, insights may be gained into some of the basic prelinguistic and fundamental principles of conceptual organization as observed in young infants. Next, a naming operation will be added as a separate module to associate representations of categories with arbitrary category names. We will see that, somewhat like children who learn their very first words, the system will first overextend its lexical categories and
CONCEPT ACQUISITION

...narrow them down. Last, I will show how, from this basic form of knowledge, the representations of concepts can self-organize into more complex structure to reflect hierarchical relationships and the continuum of expertise in different domains as observed in conceptual development. As will be observed, structured knowledge acquisition occurs as a side effect resulting from greater experience; the same gradual mechanism will incrementally update categorical representations to capture more and more of the complexity of the external world.

Children's Representation and Naming of Categories

The lexical categories of young children are frequently broader than the lexical categories of adults. As a result, young children often apply a category term to inappropriate referents (Clark, 1973; Gruendel, 1977; Rescorla, 1980). As pointed out by Mervis and Canada (1983), the child’s inadequate denotations in the usage of words are often accompanied by inadequate comprehension of their extensions. For example, in their discourse, children might use the word ball to denote any referents that have a round shape such as a round candle, a yo-yo, and a Christmas tree ornament. During a comprehension test, when asked to point at balls, children may group together all the round objects (Chapman, Leonard, & Mervis, 1986).

Eventually, after progressively restructuring and reorganizing their early conceptual knowledge, the children’s production and comprehension of category terms will narrow down and match adults' concepts (Blewitt, 1983; Clark, 1973; Mervis, 1984; Mervis & Canada, 1983). As summarized by Au (1990, p. 000), “children often revise their representations of the senses and denotations of words in their vocabularies, and this is a fundamental aspect of meaning acquisition.”

These data raise some questions about the representational structure and the mechanisms that underlie the pairing down of a child’s early lexical categories. The classical representation of concepts, the componential theory of meaning (Katz & Fodor, 1963), states that the meaning of a word can be decomposed into smaller units of meaning, called attributes. The attributes, which comprise a conceptual representation, arise from properties that are singly necessary and jointly sufficient to circumscribe the extension of the concept. For example, a componential representation of the meaning of dog would include, at least, the components four legs, body, neck, head, and tail.

The set of attributes defining a category is connected with some sort of bidirectional link to the arbitrary name of the category. The bidirectionality of the link allows the access to a componential meaning from its label in a lexicon, and vice versa. The components of meaning segregate the world of referents into those that satisfy the conceptual representation from those that do not. Using only the attributes from the previous example, the lexical
category *dog* would be overextended to *cat*, *horse*, *sheep*, *zebra*, and so on. In order for the category term to map adequately the set of referents assigned by adults to *dog*, the traditional view holds that a child will have to refine its representation for *dog* by including other relevant attributes, either perceptual or functional, such as *bark*, *bite*, and so forth.

The classical explanation of the narrowing down of infant's early lexical categories assumes that category terms play an essential role: Hearing the same category term applied to a particular set of referents should force the child to seek out attributes that differentiate this category from the initial overextended one (Adams & Bullock, 1986; Clark, 1973; Gruendel, 1977; Hull, 1920; Thomson & Chapman, 1977). In our example, hearing *sheep* applied to the set of referents that look like *dog* but have curly *hair* should facilitate the extraction of *hair* as a distinctive feature.

However, according to Chapman et al. (1986), parental use of the correct label may follow, rather than precede, the child's autonomous discovery of the distinguishing attribute. Put differently, the adequate learning and application of a category label may be contingent upon the construction of a distinct conceptual representation of the category to which the new label refers. In this latter view, the new lexical item would be an *indicator* of knowledge rather than a *means* to acquire knowledge. This view stresses the importance of the early, fundamental mechanisms of concept learning that are present in prelinguistic children (for a review, see Quinn & Eimas, 1986). These mechanisms, as Quinn and Eimas pointed out, "...are involved in recognizing that dogs are alike in ways that cats are not...before hearing the category names applied to these animals" (p. 332). An argument could be made that adults don't have a one-to-one mapping between their conceptual knowledge and their lexicon either. Take, for example, the fine discriminations most of us are able to make between different styles of architecture without being able to name these categories properly, because we just don't possess the appropriate word. If there had to be a one-to-one relationship between our words and our concepts, our knowledge would be very limited!

The independence between words and representations points out an important constraint that should be reflected in a model of concept learning: The development of conceptual representations can be somewhat independent of their lexical tagging. I will call this the *label-concept independence* constraint. This constraint reflects the idea that labels are not required for concept formation. However, we will see later that when labels are provided, they can be exploited to facilitate category learning; this observation tempers a notion of strict independence. In a computational model, label-concept independence prescribes two functionally distinct operations. The first one takes as argument an instance and gives as output its conceptual interpretation. The second operation takes as argument the conceptual interpretation and associates a name to it. In what follows, I will call the first operation *categorizing*, and the second one *naming*. 
Because the conceptual interpretation of an exemplar is the output of categorization, the representation of a conceptual interpretation is largely determined by the theory of conceptual organization considered. The componential theory of meaning presented earlier assumes that the representation of a concept is a set of singly necessary and jointly sufficient features. In this classical account of conceptual organization, categorization becomes an all-or-none process: Either an instance is characterized by the attributes that define a category or it isn’t (Smith & Medin, 1981). However, it is well known that conceptual interpretation isn’t such a simple process: Some exemplars appear to be more typical members of a category than others (Rosch & Mervis, 1975). Compare, for example, apple and tomato as exemplars of fruit, or sparrow and penguin as instances of bird. Category membership seems to be a continuous, rather than an all-or-none function.

Prototype theory is a model of conceptual organization that accounts for the continuity of category membership. This theory states that our concepts are represented as prototypes, abstractions of the category that evolve gradually as one gains experience with instances of the category and contrast categories (Posner & Keele, 1968, 1970). To illustrate, consider children exposed to the category dog. As they see more and more dogs, children eventually pick out a set of statistically relevant features that represents the central tendency of the category. At the same time, they will see cats and birds, and this will allow them to extract the relevant features, that is, the features that categorize cat, dog, and bird. Categorization in prototype theory is straightforward: An instance of a category is compared to each known concept, that is, each prototype. The closer the instance is to a given prototype, the more typical the exemplar is rated. The prototype is usually rated as the most typical exemplar, or it is classified at least as fast as the experienced exemplars, even if it has never been encountered during training (Homa & Chambliss, 1975; Posner & Keele, 1968, 1970; Rosch & Mervis, 1975). Note that categorization in prototype theory gives as output a complete conceptual interpretation of the exemplar. For example, if a hypothetical conceptual system knows just five categories, dog, wolf, cat, bird, and snake, then a dog would be rated as typical of dog, but less typical of wolf and cat because although they share some characteristics, the categories are different. However, a dog wouldn’t be rated as typical at all of bird and snake. Before categorization, there is no way to know to what stored prototype the exemplar has to be compared. In order to have a complete conceptual interpretation, it is necessary that the output of categorization indicate how typical the exemplar is with respect to each known category.

Bomba and Siqueland (1983) and Quinn (1985) observed that, like adults, infants (3 and 4 months old) also seem to have a prototype organization of their categories. This organization seems to underlie a fundamental organizing principle of conceptual memory: It is present very early in prelinguistic children and still in use in adults. This format of conceptual representation
could be imposed as a constraint to be satisfied by a model of conceptual development. However, it should be noted that other models can also account for some of the observed effects. For example, the exemplar-based model of Medin and Schaeffer (1978) accommodates the typicality effect by saying that typical items are more similar to stored exemplars than atypical items. So, a typical exemplar of a category, that is, a robin as an exemplar of bird, evokes more stored exemplars than an atypical exemplar, that is, an eagle, leading to a better typicality response in the first case.

Exemplar-based models do not deal very well with abstraction (Smith & Medin, 1981). For this reason, the kind of conceptual organization they propose will not constitute the framework of reference in the study here.

Three major constraints can be extracted from this brief review of the literature on some aspects of conceptual organization. These constraints are embedded in the model presented in the next section. The label-concept independence constraint requires the implementation of two functionally distinct operations, a conceptual interpretation of the exemplar, and the naming of this conceptual interpretation. The representational constraint states that concepts should be represented as prototypes. The conceptual interpretation given by the categorization operation needs to be complete: Each concept has to give its typicality value for each exemplar.

Architectural Issues and a Modelling Strategy
The first of the three constraints, the label-concept independence constraint, stresses the idea that learning to apply a category term to a set of referents can have an unsupervised and a supervised learning component. Research on concept learning has shown that to abstract information into concepts, a teacher is not a necessary condition. Therefore, learning to represent categories could be, to a certain extent, described as an unsupervised learning task. In neural networks, the class of self-organizing architectures (Grossberg, 1976, 1980, 1987; Rumelhart & Zipser, 1985) is particularly well suited to handle this sort of task. These networks are characterized by their ability to learn without requiring an output signal to associate with the input, as is the case in the pattern-associator. Ritter and Kohonen (1989) and Schyns (1989, 1990, in press) argued that a subclass of these unsupervised Networks, the self-organizing maps (Kohonen, 1982; Von der Marlsburg, 1973), could categorize exemplars through differentiated patterns of activation on what is referred to as a conceptual map. The study here gives a detailed account of the nature and determinants of the conceptual map's categorical judgments, and it shows how the constraint on the representation of concepts, and the constraint of a complete conceptual interpretation are implemented. The unsupervised learning component of the system—the categorizing module—is illustrated in Figure 1a.
b. Figure 1. Abstract representation of the implementation of categorizing and naming. The unsupervised component of the architecture implements categorizing (a) and the supervised learning component implements naming (b). The interface between the two abstract operations is the complete conceptual interpretation of exemplars.

Associating a category name to an exemplar is definitely an instance of supervised learning, because no system can guess the correct name without any feedback (see Hinton, 1987, for a review of supervised learning). The conceptual interpretations given by the conceptual map will be associated with the category term using an auto-associator architecture and an error-correction learning rule (Widrow & Hoff, 1960). Once learning has taken place, in order to name an instance of a particular category, a “brain-state-in-a-box” (BSB) retrieval scheme will try to disambiguate a conceptual interpretation and attach a name to it (Anderson, Silverstein, Ritz, & Jones, 1977; Kawamoto, 1988; Kawamoto & Anderson, 1985). As these architectures have been explained in detail elsewhere, here I will only motivate my selection and describe important properties. Figure 1b illustrates the supervised learning component of the overall architecture, also called the naming module. Note the conceptual output map which constitutes the interface between the two modules.

This approach differs from Ritter and Kohonen (1989) on one crucial point. Although they used a self-organizing architecture, Ritter and Kohonen’s concept learning is supervised. Their system doesn’t satisfy the label-concept independence constraint, because each time their network was presented with an exemplar, it was also exposed to the category name. Therefore, the output of their map has a totally different status. Although Ritter and Kohonen’s model exhibited important aspects of concept learning, the present model, by separating naming from concept formation, should be more powerful and psychologically realistic.

The system presented in this article is characterized on the functional level by categorizing and naming. However, in the model, modularity arises
not only at the functional level. Although it need not be, the structural level is also modular. I am taking a modelling strategy that could be called *mapped functional modularity*. The overall idea is to assign primitives of a functional architecture—here, categorizing and naming—onto different structural modules. A modular system is composed of autonomous, informationally encapsulated subsystems that communicate only at their input and output stages. "Informationally encapsulated" means, roughly, that each subsystem is blind to the other subsystems' processes and representations (Fodor, 1983). Although modularity is still in its infancy in neural networks, it nevertheless represents a promising approach to modelling. This kind of "divide-and-conquer" strategy has recently been proposed by Hinton and Becker (1990) as an attractive alternative to "one-shot," nonmodular networks. Among the advantages of modularity would be the ability to solve the scaling problem that currently plagues powerful learning neural networks based on backpropagation (Rumelhart, 1990; Rumelhart, Hinton, & Williams, 1986). If the modules are organized hierarchically so that unsupervised learning achieves most of the preprocessing of the input patterns, the burden of a supervised learning module that stands on top of the hierarchy is greatly reduced. Another advantage is the potential interest of building up *task-independent representations* that could be shared by many different modules for many different purposes, allowing many problems to be dealt with by a single system. The work presented in what follows is a modest step in this direction. More specifically, one of the goals of this article is to present how, in this simple modular architecture, a rich internal code can be proposed as a medium to propagate the informational flow from the categorizing module to the naming one.

*Mapped* functional modularity isn’t without its problems. In general, connectionism is a claim about the microstructure of cognition (Hofstadter, 1979; Rumelhart et al., 1986; Smolenski, 1988), whereas modularity is a claim about macrostructures. A potentially insightful property of connectionist models is their ability to represent two functionally distinct macroscopic characteristics of a functional architecture as two different states of a single microstructure: Think, for example, of short-term and long-term memory interpreted as two different modes of processing of a unique piece of hardware. In mapped modularity, macroscopic properties determine the microstructure, that is, the hardware configuration. Because a functional decomposition of the task is imposed directly on the hardware, an incorrect identification of functional primitives wouldn't suffer pressures from a hardware that doesn't fit with the suggested decomposition. Therefore, the selection of the primitives that should be implemented by different structures from those that should be realized by the same structure needs clear motivation on empirical as well as on theoretical grounds.
DESCRIPTION OF THE CATEGORIZING MODULE

The first experiment explores the categorizing module (see Figure 1a). I will present how it recodes a set of data by constructing interesting internal representations of the input categories without a teacher. The architecture that implements the categorizing module is based on the work of Kohonen (1982, 1984). Structurally, this architecture is characterized by a n-dimensional input vector i fully connected to a two-dimensional map of output units. Therefore, each output unit o; is connected with a n-dimensional weight vector wO to the input vector i (see Figure 2). This architecture is linked to a discriminant function and a learning rule. Formally, it achieves a quantization of the input space according to a Voronoi tessellation (Kohonen, 1988). In other words, it segregates the input space into distinct regions and encodes them on the two-dimensional map with zones of correlated levels of activation.

The first adjunct to Kohonen’s architecture is a discriminant function. It is characterized as follows. At each iteration of learning, the total activation of the output map is computed by taking the inner product of the sample vector i and the weight vector w afferent to each output unit. A winner-take-all scheme (WTA) selects the output unit o_w with the highest level of activation. Formally, this is summarized in (1).

$$o_w = \max_i (w_i^T i)$$  (1)

The second adjunct to Kohonen’s architecture is a learning rule. In order to enhance its biological plausibility, I have modified the standard architecture (Kohonen, 1982) by adding to each output unit dense local excitatory connections with its neighbors, up to a given Euclidian distance. The fixed strength of each connection is a Gaussian function of the distance between a
particular output unit and its neighbors. The learning rule is shown in (2) and (3).

\[ \Delta w_i = x \left[ \left( 1 - |o_{W}| \right) \cdot \text{lc}(o_i, o_{W}) \right] \text{ for } i \in N_w \]

(2)

\[ \Delta w_i = 0 \text{ for } i \text{ not in } N_w \]

(3)

The neighborhood structure \( N_w \) refers to the lattice of locally connected neighbors surrounding the winning output unit \( w \). The increment of the weight vectors \( w_i \) afferent to the output units \( o_i \) in \( N_w \) is a proportion of the input vector \( x \). This proportion is an inverse function of the activation of the winning unit \( o_{W} \) times the value of the local connection between \( o_w \) and the considered output unit \( o_i \), \( \text{lc}(o_i, o_{W}) \). If there is no local connection between the winning unit and the considered output unit, the increment of the weight vector is null.

The learning rule adds a fraction of the input sample to the weights of the output units in \( N_w \). Therefore, with learning, the weights of the output units in \( N_w \) become more and more similar to the input sample—formally, the weights in weight space rotate in the direction of the input sample in input space and the region defined by \( N_w \) has its output units correlated in their activation values. If we generalize this scheme to input samples of different sorts, we get different regions on the map that respond selectively to each kind of sample.

This learning algorithm orders a high dimensional input space on a two-dimensional output map, and the ordering of the output map proceeds from global to local. Two conditions are required for the ordering to converge: The neighborhood structure \( N \) and the increment of the weights \( \Delta w \) have to decrease with time (Kohonen, 1984). If \( N \) is too wide, say half the width of the map, a global order is obtained, but it doesn't converge. On the other hand, if \( N \) is too narrow, the necessary coupling among the elements to obtain a global order is lost. Because the learning rule described in (2) is a function inversely proportional to the activation of the winning output unit, the two conditions follow naturally from its characteristics. In the long run, when the activation of an output unit tends to its maximum, \( \Delta w_i \) tends to zero, and this property implements the decreasing of a gain parameter. As learning goes on, the connections linking the output units far from the winning unit become less and less efficient. As they already had a low value due to the Gaussian distribution, when the activation of the output units saturates, the neighborhood size shrinks implicitly. In Kohonen (1982, 1984), exterior mechanisms were shrinking the neighborhood size and the learning parameter. Here, both effects occur as a by-product of the learning rule and the local connections. Moreover, the standard Kohonen architecture is frozen once the learning parameter and the neighborhood size reach their asymptotic values. Here, due to (2), the system has an automatic learn-
ing mechanism whose onset is triggered by novel categories of objects, not by an external agent. This essential adaptive property will be developed in more depth later. For now, it suffices to observe that if an exemplar is dissimilar to known concepts, the activation of the winner $o_w$ is quite low, and by (2), learning reactivates itself automatically. Note, also, that in this self-organized learning, an external agent—a teacher—never specifies to the network what region of the map the exemplar has to activate, or has to be associated with: The system determines that by itself.

The following section presents a more technical discussion of the motivations for choosing Kohonen maps rather than another network. Readers uninterested in such technical material should skip to the first experiment.

In the modular approach of concept learning outlined here, the categorizing module has to develop conceptual codes gradually, without a teacher. For the naming module to work properly, that is, to learn category names and display the relevant psychological phenomena, the coding scheme of the categorizing module must reflect two essential properties: The encoding of category membership (the correlational structure of the data set) and the probability of each category and subcategories. The construction of an internal code is a speciality of backpropagation-based architectures (Cottrell, Munro, & Zipser, 1988; Hinton, 1986; Rumelhart et al., 1986). However, the Kohonen maps scheme could be better suited to model conceptual development for the reasons outlined in the following.

The task achieved by the categorizing module is known in statistical pattern analysis as dimensionality reduction or clustering (Duda & Hart, 1973). Techniques to achieve these tasks are numerous, and neural networks often implement them (Barron & Barron, 1988; Huang & Lippmann, 1987). An appropriate candidate for the categorizing module's realization could certainly be a variation of the standard backpropagation architecture, known as the encoder network, or self-supervised backpropagation (SSB; Rumelhart et al., 1986). The most sophisticated mathematical analysis of SSB has shown that for most not-too-nonlinear problems, SSB implements something like a principal component analysis (PCA) of the data set (Baldi & Homik, 1989; Cottrell et al., 1988). A brief description of PCA follows; the interested reader may consult Jollife (1986) for a more thorough presentation.

From a statistical point of view, data sets can always be described as clouds of points in high-dimensional spaces. PCA finds a small set of vectors—called principal components or eigenvectors—whose linear combinations best describe a set of data. Technically, the principal components form the set of descriptors that minimize the mean square error between the actual data points and the points described with any small set of descriptors. Each principal component extracted has a weight, or a strength, called its eigenvalue. Technically, an eigenvalue is a measure of the variance accounted for by an eigenvector. The coding scheme constructed on the hidden units of
SSB lies in the subspace defined by the eigenvectors of the system. As different linear combinations of the principal components give rise to the codes of different categories, category membership is represented. It should be noted here that an interpretation of eigenvectors or backpropagation-based representations in terms of the problem at hand is not, in general, an easy task (see Anderson & Mozer, 1981/1989, for an example of the former; and, Gorman & Sejnowski, 1988; Hinton, 1986, for examples of the latter).

If SSB satisfies the first requirement on the coding scheme, the second one, the representation of the categories’ probability structure is more problematic. Due to the error-correction nature of backpropagation, probability densities are not explicitly represented on the hidden units. Instead, they are indirectly reflected by the eigenvalues of the eigenvectors. Crudely put, the more frequent a category, the larger the eigenvalues of the eigenvectors that describe it. Larger eigenvalues give rise to stronger activation values over the hidden units, create codes more resistant to noise, and so forth. This is not to say that frequency-based effects cannot be modelled with these architectures (see Seidenberg & McClelland, 1989), but that a complete understanding of the relationships between probability densities and the gradual development of internal representations may be difficult to achieve.

Kohonen map architecture is a generalized version of a vector quantization method called k-means clustering, also known as isodata (Duda & Hart, 1973). K-means is an iterative optimization procedure to estimate a probability density function. Concretely, if a data set consists of the input vectors \( \{i_1, i_2, \ldots, i_n\} \), distributed with a probability density \( p \), the procedure looks for a reduced set of \( k \)-descriptor vectors \( \{w_1, w_2, \ldots, w_k\} \) that minimizes the sum of squares of a distance measure between each input vector, and its closest descriptor vector. As in PCA, the data set is described by a much smaller representative set of descriptors. But this is where the comparison ends. K-means is particularly useful for mapping a particular input distribution (Hecht-Nielsen, 1990; Kohonen, 1988). The \( k \) descriptors are allocated with the following rule: areas of the input space with high density receive many mean vectors, and are thereby finely represented. On the other hand, regions of low density receive few vectors and are barely represented if they are represented at all. It should be noted that this allocation of mean vectors to represent a probability density is gradual, and this property is the crux of the developmental models presented in this work.

As alluded to earlier, the Kohonen maps scheme is more complex than a simple k-means clustering procedure. The neighborhood structure that decreases with time guarantees that a metric ordering of the input space is reflected in the organization of the set of descriptor vectors. Technically, the network preserves the topology of the input space (Kohonen, 1982, 1988). The mathematical proof of this ordering has been carried in the one-dimensional case only (with a string of output units; Kohonen, 1984). Extensi-
sive testing of the two-dimensional case has shown that Euclidian distance between units on the map reflects a particular measure of distance in the input space.

To conclude, Kohonen maps satisfy the first representational requirement—the uncoding of category membership—through a parallel comparison between the input exemplar and the k mean vectors that represent the data set probability density. We saw earlier that backpropagation-based architecture could also represent category membership with linear combinations of principal components. However, due to their error-correction-based learning scheme, backpropagation architectures are not so well suited to represent probability densities, whereas Kohonen maps excel at this task. We will see that the relationship between the gradual development of conceptual codes and the incremental representation of probability densities—the second representational constraint—through k-means analysis provides a straightforward account of developmental phenomena.

Experiment 1
This experiment illustrates how weight vectors unfold in weight space to map the distribution of the input and to determine the preference of different regions of the map to different categories (dog, cat, and bird). The self-organizing architecture presented earlier was characterized by the following parameters: The input vector of dimensionality 100 was fully connected to a two-dimensional map of 10 x 10 output units. Each output unit had local connections with each of its neighbors up to Euclidian distance of 2.83. The value assigned to each local connection strength was a Gaussian function (sigma = 1.5) of the distance between a particular output and each of its locally connected neighbors.

Stimuli. The categories were composed of distortions around prototypes (Homa & Chambliss, 1975; Knapp & Anderson, 1984; McClelland & Rumelhart, 1985; Posner & Keele, 1968). Each prototype was a vector of dimensionality 100 composed of values equal to 1 (white) or −1 (black). For a methodological reason that will become apparent later, the prototypes were represented as simple drawings on a 10 x 10 array. These drawings are arbitrary of course: It is the similarity relations that matter to the algorithm, not the actual appearance of the patterns.

They were called dog, cat, and bird to represent the fact that the first two were fairly similar, and both were somewhat different from the last category (see Figure 3). The similarities among the prototypes of the different categories can be measured by the interprototype vector cosines. A vector cosine value close to 1 means that the arguments are very similar, whereas a vector cosine value of 0 means that they are orthogonal. The interprototype vector
cosines were 0.52 for dog and cat, 0.16 for dog and bird, and 0.12 for cat and bird.

To create an exemplar, a prototype was chosen and a noise vector was added to it. The features composing the noise vector were constrained to lie around the contour of the prototype drawings of Figure 3. The total amount of noise added to the prototype can be represented as a hat with a random number of pixels—ranging from 1 to 10—in it. Each feature was composed by selecting, from this hat, a random number of pixels, and this scheme was repeated until the hat was empty. Therefore, if the hat had 7 pixels, the combination of features added to the contour of the prototype could range from 7 different features of 1 pixel each to one large feature of 7 pixels. In addition, each unit with a black, “on,” value was turned on with a probability of 0.75. This is an important point because it prevents the category to be defined by singly necessary and jointly sufficient features. Given these characteristics, the vector cosine between an exemplar and its prototype varied between 0.72 and 0.92.

Learning Phase. For each of the 1,000 iterations of learning, a prototype vector was chosen according to a uniform random distribution of the categories, and an exemplar was computed as previously described. The exemplar was fed to the network, and the activation of the output map was computed as described in equation (1). The unit with the highest activation was then selected and its afferent weight vector, along with the weight vectors of its locally connected neighbors, were updated with the learning rule described in (2). It should be noted that the prototype was never presented to the network during the learning phase, but only the noisy exemplars.

Results and Discussion
In order to see the development of categorical judgements, snapshots of the activations on the map were taken after 10, 40, and 1,000 iterations of learning. To record these categorical judgments in the most homogeneous conditions, the network was tested with the prototypes of two categories, dog and cat. These results are presented on Figure 4. The activation of the output units is represented with densities of points where black means no or little
Figure 4. Activation of the output map after 10 (a), 40 (b) and 1,000 (c) iterations of learning. The stimuli are the prototype of dog, left column, and the prototype of cat, right column. The coordinates on the figure reflect the coordinates of each output unit on the map.

activation (0.0–0.2) and white means high activation (0.9–1.0). The coordinate system represents the effective location of each output unit on the two-dimensional map.

Figure 4, reveals that, with time, specific regions categorize exemplars from the different categories by specific patterns of activation. The lower right region responds mostly to instances of dog, the upper right to instances of cat, and although it is not shown in the figure, the middle left region responds preferentially to instances of bird. Before iteration 40, the con-
ceptual map gives an unspecific categorical judgment for each prototype. At that stage, the different categories cannot be distinguished. At iteration 40, although the pattern of activation for bird is already well differentiated (not shown on Figure 4) the patterns for dog and cat are still overlapping. This effect can be interpreted as a difficulty in discriminating between the similar categories dog and cat at this stage of learning. After more presentations of exemplars from each category, the categorical judgments become more and more distinct (see Figure 4, bottom). I indicated earlier that dog and cat are the most similar of the three prototypes: Their vector cosine is 0.52. Therefore, the map shows an interesting psychological property that will be used later: More similar objects are closer in the conceptual representation than are less similar objects.

The weights of the system play a crucial role in storing knowledge. To understand how these weights evolve and how they subsequently determine the activation of the output map, snapshots of the weight vector afferent to each output unit were made after 10, 40, and 1,000 iterations of learning. On Figure 5, each weight vector is represented as a 10 × 10 square of pixel values. The coordinates of the pictorial representation of each vector are exactly its output unit coordinates. Each of the 10 × 10 connection strengths is coded by a level of grey where black and white mean, respectively, maximally inhibitory and maximally excitatory connection.

The categorical judgments observed on the map—the internal representations built by the unsupervised module—now get a clearer account. With learning, two significant changes occur in the pictorial representation of the weight vectors. First, the vectors become locally organized, or ordered into regions. This local organization determines the conceptual regions on the output map. As a matter of fact, we can notice that in the lower right corner of each weight portrait, all weight vectors become more and more similar to patterns of dog, whereas the weights of the upper right corner are similar to cat. The left region of weight vectors is similar to patterns of bird. Second, once the vectors are locally organized, the connection strength becomes more sharply contrasted: whiter and darker. The vividness of the pattern of activation denoting category membership is a direct function of the connection-strength contrast. The parallel between the development of the weights and the development of the patterns of activation can be observed on Figure 5. In particular, we can notice that after 40 iterations, a distinct region of weight vectors is already dedicated to bird, whereas cat and dog are not yet clearly separated, as they will later be. We can also see that as the weight portraits get sharper—as concepts are learned distinctively—the conceptual interpretations of prototypes become more distinct too.

After 1,000 iterations of learning, each weight vector maps nearly exactly the prototype of one particular category. This effect is quite interesting because the prototypes were never presented during the training phase.
With the learning rule presented earlier, the weights of different regions *pick out the relevant features* that characterize the meaningful areas of the input space: the categories. Because the categories are composed of variations around a prototype, in the long term, the variations cancel out, and the weight vectors pick out the center of the input subspaces, the prototypes. Therefore, the system represents the relevant features that characterize the prototypes of the categories, leaving irrelevant features out of the representation, much as human subjects seem to do (Bomba & Siqueland, 1983; Homa & Chambliss, 1975; Knapp & Anderson, 1984; Posner & Keele, 1968). With more noisy categories, Schyns (1989, 1990) showed that the
organization of the weights in a specific conceptual region was as follows: The centroid of the region encodes the prototype of the category, and the surrounding weight vectors encode it with less and less accuracy with distance.

The gradedness and completeness properties of categorical judgments are illustrated in Figure 6, where the activation of the map is computed in response to one exemplar from dog, cat, and bird. The similarity of each exemplar to the prototype of the categories is measured by the vector cosines shown on the figure. In Figure 6b, the exemplar of cat is most similar to the prototype of cat but is also similar to the prototype of dog. So, the region that responds preferentially to cat is most highly activated, but the area for dog, the lower right, is also highly activated. In contrast, if the vector cosine between the exemplar and the prototype of other categories is low, the activation of these particular regions will be low too. In Figure 6c, the exemplar of bird is orthogonal to the other prototypes, resulting in minimal activation values in their regions.

From these examples, we can observe that the map doesn't reflect an all-or-none conceptual interpretation of the exemplars. Instead, the interpretation is graded according to the similarity between the exemplar and each prototype known by the categorizing module: If the exemplar is ambiguous, say a cat that looks like a dog, its conceptual interpretation will reflect this ambiguity with a state of activation that overlaps two regions. It should be noted that the prototypes are the only input samples that maximize the excitation of a particular region and minimize the ambiguity. A comparison of Figures 4 and 6 illustrates this effect. For each conceptual region, the highest excitation is obtained when the prototype is presented, and the minimum amount of overlapping between conceptual regions is also obtained in the presence of the prototype. Therefore, the conceptual interpretations given by the map agree with the conclusion of Rosch and Mervis (1975, pp. 598-599) that "prototypical members of... categories are those which bear the greatest family resemblance to other members of their own category and the least overlap with other categories."

The results of this first simulation agree with prototype theory on most of its major points, and they suggest a particular implementation of its important features. Prototype theory proposes that concepts are stored as prototypes; this architecture encodes prototypes in its mean weight vectors. In prototype theory, the exemplar is compared to each known prototype: here, a conceptual map represents a complete conceptual interpretation of the input sample, where each zone of activation reflects the typicality between the exemplar and a particular prototype. The conceptual interpretation is more vivid and less ambiguous for the prototype of a category than for its exemplars, a result that can be interpreted as a form of prototype effect. Therefore, the categorizing module accounts for the sort of concept learning observed both in linguistic and prelinguistic children. This sort of
Concept learning is characterized by the fact that the representation of a category precedes, rather than follows, the association of a category name.

**DESCRIPTION OF THE NAMING MODULE**

The second experiment explores the supervised learning component of the system; more specifically, how category terms can be retrieved from conceptual interpretation of exemplars. In order to build a bidirectional link between concept and their lexical entries, the concept-name associator (the
SCHYNS

naming module) was implemented with a pattern associator and an error-correcting procedure (Widrow & Hoff, 1960). The idea behind a pattern associator is, as its name indicates, to associate an input pattern to an output pattern. Variants on this architectural theme, differing from one another in associative power, have been widely employed by modelers of concept learning (Amari, 1977; Anderson & Murphy, 1986; Corter, Gluck, & Bower, 1988; Gluck & Bower, 1988; Hinton, 1981, 1986; Knapp & Anderson, 1984; McClelland & Rumelhart, 1985; Rumelhart et al., 1986). A simple variant of the linear pattern associator, the autoassociator, occurs when input and output patterns are the same. This sort of associator will be used here.

The idea behind error correction is to minimize, that is, to correct, the error between a teacher-output vector and the actual output vector by modifying a connection matrix. In this model, the teacher vector is composed of two parts: the conceptual interpretation of an exemplar, that is, the output of the map, and the correct category name. As the pattern associator is autoassociative, the actual output and the input are exactly the same.

To explain the error correction scheme more formally, consider a standard pattern associator that has to learn the association between the patterns represented by the vectors f and t. Formally, the error-correction learning algorithm can be written as follows,

\[ g = A f \]  
\[ e = t - g \]  
\[ \Delta A = \alpha e f^T \]

where g is the output vector computed by multiplying the input vector f with the weight matrix A. In this simulation, f is composed of two parts: the conceptual interpretation f' and the category name f", that is, f = f' | f" (where | stands for the operation of concatenation); e is the error vector between the teacher t, and the output vector g. The modification of the weight matrix A is a proportion of the error e times the transpose of the input vector f. When the output vector g matches the teacher t, that is, when f and t are correctly associated, e is equal to the zero vector 0, and the difference in weights \( \Delta A \) is the zero matrix.

In an autoassociator, f and g are equivalent and the equations become

\[ f = A f \]  
\[ e = t - f \]  
\[ \Delta A = \alpha e f^T \]

Once the system has learned its lexicon of category terms, a mechanism has to be provided to label the conceptual interpretation produced by an exemplar. In other words, how can we retrieve f", given f'? We saw earlier
that the activation on the map reflects several conceptual regions with graded levels of activation. Therefore, to be labelled correctly, this complete conceptual interpretation has to be disambiguated by some process. In order to do so, the interesting dynamical properties of the BSB model will be used here (Anderson & Murphy, 1986; Anderson et al. 1977; Kawamoto, 1988).

The BSB model is a dynamical system that evolves to stable states, or attractors, by implementing a gradient-descent algorithm (Golden, 1986, 1987). Our intuition about dynamical systems of that sort can be strengthened by referring to the landscape metaphor. Imagine an energy landscape, with hills and valleys where height corresponds to regions of local energetic maxima and minima. The actual state of the dynamical system can be thought of as a ball rolling on this landscape. The ball follows the gradient of steepest descent until it reaches the bottom of its nearby valley and stays there. Formally, we will say that the activity of the network has fallen to a state corresponding to a local energy minimum (Hopfield, 1982, 1984). On its way to a stable state, a corner, an attractor, or an energy minimum, the dynamical system disambiguates the conceptual representation and names it. In the meantime, it meanders around in polymorphous meanings.

Formally, the dynamics of the system are described with

\[ x(t+1) = \lim(\alpha x(t) + \gamma A x(t)) \]  

where \( \lim(x_i) = 1 \) if \( x_i > 1 \),

\[ -1 \text{ if } x_i < -1, \]

\[ x_i \text{ otherwise,} \]

and the equation to measure energy is the simple quadratic function of (11)

\[ E(x) = - \frac{1}{2} x^T A x \]  

At time \( t = 0 \) of the retrieval process, the state vector \( x(t) \) of the dynamical system is given an initial value; \( \gamma A x(t) \) computes the amount of feedback that \( x \) receives at time \( t + 1 \). This feedback information should eventually saturate the components of the state vector by driving it to a particular attractor. Two parameters are of importance in this iterative scheme: \( \alpha \) is a state-decay parameter that enables the system to leave a stable state and eventually go to another; \( \gamma \) is a factor that weights the total amount of feedback \( x(t + 1) \) receives. \( \lim(x_i) \) is a limiting function that prevents the activation of any unit \( x_i \) of the state vector \( x(t) \) from growing without bound. With these characteristics, the possible solutions of the dynamical search lie at the corners of a state space shaped as a hypercube. The limiting function presented in (10) is a discontinuous version of the sigmoid used by Grossberg (1976, 1980). It should be noted that the retrieval scheme implemented here is synchronous and deterministic (a probabilistic one is explained in Ackley,
Hinton, & Sejnowski, 1985). Equation (11) is the Liapunov energy function described by Hopfield (1984) for the class of neural networks whose limiting function is similar to a step function (see also Golden, 1986).

\( A \) represents the cross-connections between the elements of the different \( f \) the autoassociative memory knows about. At test phase, to retrieve a particular category name, the conceptual interpretation \( f' \) is the initial state of the dynamic search, that is, at \( t = 0 \), \( x(t) = f' \). Because the information about the name, \( f'' \), is stored in the cross-connection matrix \( A \), the search strategy repeatedly passes this vector through \( A \) to reconstruct the missing information. In the energy-landscape metaphor, the state vector composed solely of \( f' \) provides the initial state of the dynamical system. By repeatedly adding new information to the state vector, the state of the system "rolls" incrementally to a minimum. If the state vector at this minimum corresponds closely to \( f - f' | f'' \), this amounts to correct retrieval of the category name.

**Experiment 2**

Experiment 2 presents the naming of exemplars through the labeling of conceptual interpretations. Experiment 1 showed that conceptual interpretations of prototypes were, at the same time, less ambiguous and more contrasted than conceptual interpretations of exemplars. This characteristic of categorical judgments provides the basis for a prototype effect in naming.

**Stimuli.** The \( f \) input vectors had dimensionality 132; \( f' \), the \( 10 \times 10 \) conceptual interpretation on the map, was stored in the 100 first units and \( f'' \) in the 32 last units. In order to keep the label representation simple, "dog," "cat." and "bird" were coded with an ASCII binary code. Each character was represented by its ASCII number on eight units, with 1 and \(-1\) standing, respectively, for the binary values 1 and 0. When a label had three characters, a hyphen was added to it (e.g., "cat-").

**Learning Phase.** After 700 iterations of the first experiment, when the conceptual maps were formed, the conceptual interpretations and the correct category names of the exemplars were composed on \( f \) as described previously. This hybrid vector was then autoassociated with a Widrow-Hoff (1960) learning rule. This scheme was repeated for 300 learning iterations.

**Testing Phase.** An input pattern, either a prototype or an exemplar, was presented to the system. The resulting categorical judgment was written as the \( f' \) component of the \( f \) vector. I recorded, as a measure of reaction time, the number of iterations required by BSB to fill the \( f'' \) component, that is, to label the conceptual interpretation.
**Results and Discussion**

Once the labels were associated to the conceptual interpretations, the testing phase was performed. Mean reaction times were computed for a hundred exemplars of each category. As expected, the mean reaction times of exemplars were longer than those of the prototypes. For category dog, 12 iterations were necessary to name the prototype whereas a mean of 14.38 iterations was required to name a hundred exemplars. For category cat, the figures were, respectively, 8 and 13.93, and for category bird, 11 and 12.82. These results agree qualitatively with those observed in human categorization experiments in which prototypes that were never presented to subjects were more readily identified than the learned and the new exemplars of categories (Homa & Chambliss, 1975; Homa, Cross, Cornell, Goldman, & Schwartz, 1973; Knapp & Anderson, 1984; Posner & Keele, 1968).

The reason for the faster naming of the prototype over the exemplar is found in the cross-connection matrix and the characteristics of BSB dynamic search. BSB is a gradient-descent algorithm, it minimizes a Liapunov energy function until it reaches an attractor point. A critical aspect of a gradient-descent search is the initial state. In general, the closer the initial state is to an attractor on the energy landscape, the faster the state of minimum energy is reached.

Table 1 shows that for each category, the energy value of the prototype is closer to the attractor than the average energy value of the exemplars. The numbers in the third column represent the energy values at which the category names are retrieved. The last column shows the asymptotic values of the energy function, that is, the energy of the state vector when it has reached a corner and all its components saturate. Depending on the purity of the conceptual interpretation f', the dynamical system starts either close to an attractor, as is the case with prototypes, or a little bit further from the attractor, as is the case with an exemplar, or somewhere between two attractor points, as can be the case with ambiguous conceptual interpretations. Therefore, the process responsible for the prototype effect is nested in the naming
module; its performances are modulated by the clarity of the conceptual judgments of the categorizing module.

I argued earlier that by reducing the complexity of the outside world to some classes of equivalence, concepts also constrain the content of our thoughts. In other words, we can be blinded to some categories or subcategories of the world because we have a limited number of concepts to grasp the complexity of reality. The next simulations show how a learning system can partially overcome these conceptual constraints. In each of these experiments, the recurring theme will be the notion of specific constraints imposed by conceptual development on lexical acquisition.

**RESTRUCTURING CONCEPTUAL KNOWLEDGE**

Recent research on early lexical acquisition has concentrated on the initial assumptions that constrain word meaning (An & Gluskin, 1990; Markman, in press). These initial constraints give a broad framework to interpret a phenomenon such as the overextension of category terms. With the techniques available nowadays, it would be presumptuous to pretend to model such complex biases completely. However, we will see that the modular model presented here can account for some determinants of these constraints.

When a child participates in ostensive learning, when he or she experiences objects and hears the label applied to the objects, the child faces a difficult problem: What does the label refer to? The novel term could refer to an object category, but it could also refer to a part of the object, or its color, or its weight, or its substance, and so on. One way in which children could initially constrain word meaning is by assuming that the novel label refers to the whole object rather than to its parts, substance, or any other properties. This initial constraint on word meaning is referred to as the whole-object constraint (Markman, in press). Another important early constraint on word meaning is the mutual exclusivity constraints. Mutual exclusivity states that children constrain word meaning by assuming at first that each object has only one label. Put differently, children would initially assume that words refer to mutually exclusive sets of referents.

Consider, as a simple illustration of mutual exclusivity and the whole-object constraints, the situation where two objects are presented, one of which has a known label, and the other one does not. If a new label is then mentioned, on the whole-object assumption, the child should look for a whole object as a first hypothesis on word meaning. Then, on the mutual exclusivity assumption, the child should assume that the novel object is referred to by the new label (Markman, in press). Another example of mutual exclusivity occurs when a second label is given to an already-named exemplar. On the mutual exclusivity assumption, children sometimes narrow down the extension of the old label by excluding the object just named as the member of a new category (Markman, in press).
At least two accounts of mutual exclusivity are possible in the modular framework presented here. The first one is a bottom-up account. It states that distinct conceptual interpretations of categories precede, and are required for, the mutual exclusivity of category terms. The second account is grounded on the argument that a new category name applied to an already-named set of objects should force the child to segment the new category from the initially overextended one. In this version of mutual exclusivity, category names force the encoding of categories on the conceptual map by constraining its state of activation. These two versions of mutual exclusivity are more complementary than contradictory. Although the processing assumptions are different—the first account is a bottom-up process whereas the second one is a top-down process—both versions of the constraint concentrate on the emergence of nonoverlapping conceptual interpretations. In the following experiment, I will present a model of the bottom-up account; the implementation of the top-down one will be discussed afterwards.

Experiment 3
This experiment is decomposed into three parts. In the first phase, the system, loaded with the three concepts and the three labels learned in the previous experiment, will be briefly exposed to exemplars from a new category, wolf. This category will be quite similar to the dog category. During the second phase, the unsupervised module of the network will have extensive experience with exemplars from the three known categories plus wolf. When a reorganization of its conceptual knowledge is realized, the system will learn to associate correctly the new category term, "wolf." At different phases of the experiment, an overextension and a narrowing down of category names will be observed. These effects will be related to the bottom-up development of conceptual codes.

Stimuli. The stimuli were exemplars computed, as explained before, from the prototypes of four categories. The new prototype, wolf, was more similar to the prototype of dog than to any other prototype. The interprototypes' vector cosines were 0.5 for wolf and dog, 0.14 for wolf and cat, and 0.02 for wolf and bird.

Relearning Phase. The network was initially loaded with the knowledge acquired in the previous experiment; it knew three concepts and three category names to label them. The relearning phase was carried out in two steps. First, for 250 iterations of relearning, the categorizing module was exposed to exemplars of the four equiprobable categories. No name was associated with the conceptual interpretations; neither new nor old names. Second, after relearning, and for 300 iterations, the four names were associated to the new conceptual interpretations.
Testing Phase. Two tests were undertaken in two different states of lexical development. First, before any new label was learned, snapshots of the weight portraits and snapshots of the evolution of the conceptual interpretations were taken after 10, 40, and 250 iterations of relearning. The naming performance of the system was also measured after 250 iterations. Second, after the new symbol "wolf" was learned, the capacity to name categories was tested as in Experiment 2.

Results and Discussion
In Figure 7, the conceptual interpretation of wolf's prototype—the right column—is compared with the conceptual interpretation of dog's prototype—the left column—after 10, 40, and 250 iterations of relearning. We saw earlier that the lower right region of the map was responding preferentially to instances of dog. Because the prototype of wolf is more similar to the prototype of dog than to any other, a wolf is interpreted as an instance of dog at the beginning of relearning (see Figure 7a, right column). If BSB attempts to name this conceptual interpretation, "dog" comes out of the naming module, an expected effect. This result illustrates the constraints that concepts can impose on the interpretation of the outside world. Because the new category has not yet been represented in the system's conceptual knowledge, all of its exemplars are regarded as instances of the closest concept that can interpret the new category. As the network gains more and more experience with wolf, it eventually learns how to distinguish this category from the others. This is illustrated on Figure 7, the right column, where a new conceptual region gradually emerges from the dog region to interpret the new category as a separate piece of knowledge.

The weight portraits of the system after 10, 40, and 250 iterations of relearning are shown in Figure 8. We can see, after 10 iterations, that new weight vectors superimpose on some of the weight vectors that were dedicated to represent bird and dog. Eventually, as the weights underlying the new conceptual region incrementally pick out the relevant features that characterize the new category, its prototype is represented distinctively. The prototype of wolf is represented in the weight vectors of the lower left corner of Figure 8.

It should be noted that the new conceptual region is close to the region that responds preferentially to dog (the lower right region both of the conceptual interpretations and the weight portraits of Figures 7 and 8). This is not accidental. The self-organizing learning algorithm updates the weights by first selecting a winner. We have seen that the region for dog is the one that responds the most to wolves in the beginning of learning. Therefore, the weights that will first be updated to represent wolves will be in the dog region. With learning, the new region will migrate to a corner where the competition for space allocation is less intense. A more detailed description
of the mechanism underlying this kind of migration on the map due to a repelling effect will be given later.

After 250 iterations of relearning, the naming of the conceptual interpretations resulting from the four prototypes gives three labels. Then four category names are provided for 300 more iterations. Importantly, once all labels are learned, a one-to-one relationship between concepts and labels is achieved. The BSB reaction time for each prototype is shown in Table 2. The left part of Table 2 shows the reaction times before wolf was learned,
Much like children’s overextension errors and their eventual corrections, in this model, when the new category wolf is learned, conceptual interpretations of its exemplars are labelled as “dog.” At this stage, the lexical item “dog” is overloaded. Its set of referents is overgeneralized because it comprehends not only the dogs, but also the wolves. However, the conceptual map interprets wolves distinctively from dogs, so a new label can be associated to the wolf area of the map. When this new category term is learned, the initially overextended category name “dog” narrows down to the correct set of referents, whereas “wolf” refers to the remaining referents. These results are presented on the right part of Table 2.

Chapman et al. (1986) and Quinn and Eimas (1986) showed that a new category term could follow, rather than precede, the acquisition of a new
TABLE 2
BSB Reaction Times for the Retrieval of Names in the Two Conditions of Experiment 3

<table>
<thead>
<tr>
<th>Prototype</th>
<th>Before the New Category Name is Learned</th>
<th>After the New Category Name is Learned</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Name Retrieved</td>
<td>Reaction Time</td>
</tr>
<tr>
<td>dog</td>
<td>&quot;dog&quot;</td>
<td>14</td>
</tr>
<tr>
<td>cat</td>
<td>&quot;cat&quot;</td>
<td>11</td>
</tr>
<tr>
<td>bird</td>
<td>&quot;bird&quot;</td>
<td>8</td>
</tr>
<tr>
<td>wolf</td>
<td>&quot;dog&quot;</td>
<td>27</td>
</tr>
</tbody>
</table>

concept in a kind of bottom-up category learning. By explicitly grounding the acquisition of a simple lexicon on the development of conceptual knowledge, the modular architecture of the current research proposes a specific implementation of Chapman et al.’s and Quinn and Eimas’s proposal. By making the adequate learning and correct usage of a new category term ultimately contingent upon the development of a distinct concept, the model provides a possible implementation of the mechanism by which the early lexical categories of children narrow down to the adult ones.

Although conceptual development can occur independently of lexical acquisition, developmental studies have shown that the presentation of new exemplars together with a new symbol facilitates the extraction of a new category (Baldwin, 1989; Landau, Smith, & Jones, 1988; Markman, 1989). This phenomenon is interpretable as the top-down instance of mutual exclusivity alluded to earlier. Because mutual exclusivity constrains each object to be named with a unique label, a violation occurs when a new label denotes exemplars of an already-named category. Consider, as an illustration, the case where a child has represented two similar categories, that is, dog and wolf, with the same concept. On the mutual exclusivity assumption, hearing the new category term “wolf” applied to an object that was before labelled with “dog” should force the child to extract the relevant features that segment wolves from dogs. This second facet of mutual exclusivity is a specific instance of top-down processing in concept learning.

The top-down influence of labels on conceptual interpretations is easily implemented with the conceptual map scheme if the map integrates two different sources of input. The first source of input is the bottom-up conceptual interpretation of exemplars presented earlier. The second source of input is the top-down conceptual interpretation associated with a particular item of the lexicon. The top-down influence takes advantage of the bidirectional link between conceptual interpretations and lexical items (see Figure 1). This link not only allows the retrieval of a category name from a conceptual interpretation, but also the reconstruction of a conceptual interpretation associated with a particular lexical item. If a new category name is presented
to the system, the top-down interpretation it generates may be used to expedite concept learning. The next experiment explores the facilitation that top-down symbolic influences produce on the bottom-up extraction of a new concept presented earlier.

Experiment 4
In order to isolate the top-down effect on concept extraction, the system first reproduced the initial stage of the previous experiment by learning three concepts and three labels ("dog," "cat," and "bird"). Then, the architecture was presented with exemplars from the new category wolf, as in the relearning phase of Experiment 3. Here, however, the conceptual map was computed by considering both bottom-up categorical judgments and top-down symbolic constraints. The two sources of input were integrated with a weighted sum of bottom-up $\beta$ and top-down $\tau$ influences (see Equation 12).

$$o_i = \beta [w_i^T i] + \tau \left[\lim(\alpha x + \gamma A x)\right]$$

Learning Phase. A top-down conceptual map was reconstructed from a symbol with BSB dynamic search. The search was initialized by clamping a particular category name in the $f'$ component of the BSB state vector, that is, $f = 0|f'$ (see Equations 9 and 10). It was stopped either when one of the units of $f'$ saturated (equal to 1.0 or -1.0), or, by default, after 10 iterations through the weight matrix. The state of activation $f'$ was then fed to the conceptual map as the top-down component of (12). The bottom-up contribution was computed as before according to the first part of equation (12): $\beta$ and $\tau$ were set to 0.5.

Testing Phase. The bottom-up and top-down contributions were recorded for the first exemplar of wolf presented to the system. The reconstructed conceptual interpretation from the symbol "dog" was also recorded, and a snapshot of the weight portrait was taken after 150 iterations of learning.

Results and Discussion
Figure 9a, left column, shows the conceptual interpretation reconstructed from the symbol "dog," with the top-down process, previously explained. As expected, the conceptual code synthesized covers the region that responds mostly to dog. This result isn’t surprising: It simply illustrates the true bi-directional nature of the link learned through repeated associations between complete conceptual interpretations and lexical items. In this case, the top-down, symbol-driven expectation is confirmed by the bottom-up, exemplar-driven contribution (see Figure 7a, left column). Consequently, the weighted summation achieved by (12) assigns the highest activation to a unit in the
Concept Acquisition 491

Figure 9. Bottom-up account of the prototypes of dog (a, left) and wolf (a, right). Top-down contribution of the symbol “wolf” (b). Weight portrait of the categorizing module after 150 iterations of relearning (c, left). Weighted summation of bottom-up and top-down contributions for an exemplar of wolf and the symbol “wolf.”

dog region. As illustrated by this example, when top-down expectation and bottom-up interpretation coincide, the exemplar is correctly named, and its conceptual code is what it should be.

If bottom-up and top-down contributions do not coincide, mutual exclusivity is violated and a new category must be encoded on the map. We saw earlier that a bottom-up process could progressively encode a new category by incrementally extracting a new conceptual region from the one that represents the overextended category. The top-down influence achieves a similar
result, but faster. Consider the example presented on Figure 9a and b, right column. The first map shows the bottom-up interpretation of *wolf*. As in the previous experiment, a wolf is categorized as a particular instance of *dog*. The second map reflects the top-down pattern synthesized in response to the unknown label “*wolf*.” Because “*wolf*” has not yet been grounded on a particular conceptual code, the pattern constructed doesn’t match with the bottom-up interpretation: It cannot be directly related to a specific concept, as was the case for “*dog*.” Technically, the pattern synthesized from the new label is a linear combination of the stored conceptual interpretations, weighted by the pseudocorrelations between the new name and the known category terms. In the following discussion, I assume that the new symbol and the overextended category term have no, or a small, correlation.

When mutual exclusivity is violated, the top-down contribution to the map can be conceived of as additive noise. This noise helps splitting conceptual representations because it repels the winner unit out of the critical perimeter of interference of the overextended category’s representation. At the same time, the bottom-up constraint attracts the winner in the critical perimeter. A suitable integration of these two conflicting “forces” guarantees that the nice property of the conceptual coding scheme is still preserved: Similar categories are represented close to one another on the conceptual map. As a matter of fact, on Figure 9c, last column, the winner unit to *wolf* is located at the periphery of the region that responds mostly to *dog* (at (9, 5) on the map in x and y coordinates). In the previous experiment, with bottom-up processing only, the winning unit was located in the *dog* region (at (10, 7) in x and y coordinates). The addition of top-down influences facilitate the emergence of a new conceptual code. This happens because the weight vectors allocated to the representation of a new category need not be pulled out of the overextended region as is the case with a bottom-up process (the new code emerges roughly 110 iterations earlier with a top-down contribution; a more detailed explanation of this mechanism will be given later). Figure 8c and Figure 9c, left column, show weight portraits of the unsupervised module once the encoding of the overextended category has been achieved, respectively, with bottom-up, and bottom-up and top-down processing. Thus, much like children’s concept learning, the addition of top-down influences speeds up the bottom-up process of concept extraction presented earlier.

In general, mutual exclusivity states that children start off with the assumption that each object in the world is referred to by only one label (Markman & Watchel, 1988). Although different names might be applied to the same object in different phases (e.g., an overextended label such as “*dog*” preceding an appropriate label such as “*wolf*”), only one name is applied at any one phase. The architecture presented here naturally honors mutual exclusivity through the bias of contrasted conceptual interpretations. Category terms are mutually exclusive if they are mapped onto mutually exclusive sets
of objects. Thus, nonoverlapping internal codes provide the basis for the mutual exclusivity of category terms. However, mutual exclusivity is really honored if the one-to-one association of internal codes to category names is an internal constraint of the naming module. By internal, I mean a constraint that ensues solely from the dynamics of the module, as opposed to an external constraint such as a teacher that provides one name per category. If mutual exclusivity were violated, that is, if one code were accidentally associated with two different labels, the naming system wouldn't work correctly. Technically, the autoassociator would most likely learn a category name that results from the linear combination of the two labels associated with the same conceptual interpretation. Moreover, even if this problem were solved with a more powerful learning scheme, another problem would still occur if mutual exclusivity were violated as explained earlier. Because BSB is a deterministic search algorithm, how could it retrieve one word sometimes and the second word some other times, given the same conceptual interpretation as initial state? Thus, mutual exclusivity is a fundamental internal constraint on the proper working of the naming module presented here.

HIERARCHICAL RELATIONSHIPS BETWEEN CONCEPTS AND EXPERTISE ACQUISITION

The Development of Hierarchical Dependencies Between Concepts

In the previous section, I described the categorization and naming of simple categories. However, it is well known that real-world categories do not have such an elementary structure. We know, for example, that dogs, cats, and birds are animals, not furniture. We also know that specific kinds of dogs are characterized as German shepherd, Doberman pinscher, and so on. In other words, we can organize object categories into hierarchies. Within a hierarchy, a relation of inclusion prevails so that more specific categories (e.g., dog) are included into more general ones (e.g., animal). A particular level of the hierarchy, known as the basic level, is preferred in classification, presumably because it is the level at which categories are discriminated most easily (Murphy & Smith, 1982; Rosch, Mervis, Gray, Johnson, & Boyes-Broem, 1976). More general levels are usually called "superordinates." and more specific levels "subordinates."

Learning the hierarchical organization of categories—learning their inclusion relationships—is a difficult stage children have to go through to reach a mature conceptual organization (Au & Glusman, 1990; Callanan, 1985, 1989; Callanan & Markman, 1982; Inhelder & Piaget, 1964; Markman, Horton, & McLanahan, 1980; Taylor & Gelman, 1989). As alluded to earlier, in modular architecture, the relative encapsulation of knowledge makes one subsystem blind to the other's processes and representations. However, when confronted with structured categories, the naming module
should be able to label correctly exemplars at the different hierarchical levels. For example, a *German Shepherd* should not only be labelled as a *German Shepherd*, but also as a *dog* and an *animal*. Therefore, the inclusion relationships among categories that are represented in the categorizing module should be made accessible to the naming module. In the modular design presented here, a sensible solution to this problem is to develop, at the interface between the modules, an internal code of the inclusion relationships between concepts. In this case, the naming module could tag internal codes with hierarchies of distinct symbols, where each symbol in the hierarchy would correspond to a specific code on the conceptual map. With this kind of organization, the inclusion relationships of category terms would be grounded on the inclusion of concepts reflected by the conceptual map's internal codes.

**The Development of Conceptual Expertise**

It has been observed that people often differ dramatically in the knowledge they have about different domains in the world. Even in domains where no certified experts can be found (e.g., the domain of can openers), expertise may be gained as a side effect of a greater experience with the objects of the domain. What kind of differences can one expect to find between the knowledge of an expert and the knowledge of a novice? More specifically, how do the representations of categories eventually reflect expertise, and how does expertise affect categorization?

Rosch et al. (1976) noticed that experts segment their domain of expertise into finer and more specific categories than novices. This result suggested that the relationship between a categorizer and the objects to categorize was contingent on the categorizer's level of expertise (see also Chi, Feltovich, & Glaser, 1981). For example, when an airplane mechanic sees an airplane, he or she may not only see an airplane, as a novice would, but also a particular model of airplane, with a particular kind of engine, manufactured by a particular company. Therefore, the airplane mechanic not only uses the basic concepts used by novices, but also has at his or her disposal a vast amount of specific knowledge novices do not possess.

Concepts of experts and novices may have the same principles of organization, but experts simply have more low-level concepts to deal with the vast number of details they know. In a recent study, Tanaka and Taylor (1991) gave some empirical support to this theoretical proposal by showing that extensive domain-specific experience increases distinctive information at the subordinate level. They showed that the basic level—the preferred level for categorical discriminations—of experts in one domain (*bird* or *dog*) was equivalent to the subordinate level of novices; basicity was “going down” with expertise. To illustrate Tanaka and Taylor's results, consider the category *bird*. If it is represented with the attributes *has wings, flies, has feathers.*
a maximal distinction between, for example, birds and dogs is possible. More specific, low-level features are required to distinguish between, for example, robin and sparrow. Tanaka and Taylor demonstrated that the acquisition of expertise was made at the subordinate level of abstraction. The acquisition of low-level relevant features, as Rosch et al. (1976) suggested, enables the expert to make finer categorizations in his or her domain of expertise because these features are part of the low-level concepts necessary to segment the domain finely. This characterization of expertise is very similar to the acquisition of hierarchical relationships in children's concepts. Both suggest that high-level concepts are broken down into low-level ones by adding low-level relevant perceptual features to crude concepts as the context of categorization changes.

**Experiment 5**

This experiment investigates how the modular architecture can naturally develop expertise and hierarchical relationships between concepts as a by-product of gradually experiencing a structured external world. I will show how this conceptual knowledge can be associated with hierarchies of symbols. Once again, representing structured categories and gaining lexical acuity will be considered two separate operations.

The development of expertise in two symmetric systems will be compared. The first system will experience more exemplars from dog, then it will experience exemplars of bird, and it will be exposed to all the names of the different sorts of dog it knows about. The second system will see many instances of bird and a few instances of dog. However, in this case, its linguistic environment isn't rich for bird, and names of the four subtypes were not presented to the model.

**Stimuli.** I constructed a taxonomy of two different categories, each composed of four subcategories. Each category was characterized by common features (see Figure 10a). The prototypes of the subcategories were created by adding some features to the common features characterizing the category (see Figure 10b). The prototypes of the subcategories were quite similar to one another: The average vector cosines in each category was roughly 0.5.

**Learning Phase.** Each system was exposed 80% of the time to its category of expertise; the other category was presented 20% of the time. Once a category was randomly selected according to this distribution, the prototype of a subcategory was chosen with a random equiprobable distribution. The exemplar presented to the network was computed as before, by adding noisy features to the prototype of the selected subcategory. This scheme was repeated for 1,000 iterations. After 700 iterations of the preceding phase, and for 300 iterations, three symbols were associated to the conceptual
interpretations of exemplars. These symbols corresponded to the three levels of the hierarchy of labels, for example, ANimal, DOg, PIInscher (see Table 3). For each of these labels, the ASCII representation of the two first letters (see Table 3, capitals) were clamped in specific slots of the $f$ component of $f$, where $f$ had dimensionality 148. In this example, $f'$ would look like AN DO PI. Thus, name length was equal across category levels. Note that mutual exclusivity is violated when a hierarchy of labels is associated with the same object (Markman, in press, Taylor & Gelman, 1989). Because the naming module honors mutual exclusivity, I had to represent the different labels in different slots to be able to learn the hierarchies. For the dog

Figure 11. Weight portraits of the expert in bird (left) and the expert in dog (right) after 40 (a) and 1,000 (b) iterations of learning.

Testing Phase. Snapshots of the weight portraits were taken, for both systems, after 40 and 1,000 iterations of learning. Once the labels were learned, the naming capacities of the two experts were tested for their category of expertise and the category they were novices of. The time taken by the dynamical BSB scheme to fill each slot of \( f \) was recorded.

Results and Discussion

After 40 iterations of learning, the weight portraits of both “experts” reveal that the relevant features characterizing basic-level categories have been picked out. The vertical bar is salient in the weight portrait of the expert in dog (see Figure 11a, right), whereas the diagonal bar is salient in the weight portrait of the expert in bird (see Figure 11a, left). At this stage of conceptual
development, no subordinate relevant features are clearly represented in the weight portraits of both systems. The fact that medium-level concepts emerge first has also been observed in the conceptual development of infants by Mervis (1980). In the system presented here, basic-level concepts constrain the schematized external world to be understood in a crude way. As we will see in what follows, a crude mode of interpretation is a necessary prerequisite to the acquisition of a finer mode of grasping the external world.

When several categories have to be represented in the categorizing module, they have to share a finite amount of representational resources, that is, the weight vectors underlying the output units of the map. In Figure 11b, on the weight portrait of both experts after 1,000 iterations, we can see that the most frequent category has received approximately 80% of the total number of weight vectors, whereas the least frequent category is allocated 20% of the representational resources. This result instantiates an interesting property. The amount of resources allocated to represent a category depends on the category's probability of occurrence. In other words, the most experienced categories receive more representational resources. Now, for each category, the amount of internal structure that can be represented in the weights depends on the amount of resources available to encode this category. This property is also observable on the weight portraits of both systems (see Figure 11b). After 1,000 iterations, all the subcategories of the category of expertise are represented with distinct groups of weight vectors. Conversely, the novice categories are not as finely represented. For each expert, the novice category is confined in a small space and only one or two of its subcategories are represented without the interference of other subcategories. Take, for example, the weight portrait of the expert in dog (Figure 11b, right). The prototype of the subcategory Robin—arrow pointing up has been fully represented in the weights. In contrast, the prototype of Crow—arrow pointing down—is only partially represented because it suffers the interference of the features of dog's prototype: the vertical bar. The other vectors devoted to the representation of the novice category have encoded only the prototype of bird: the diagonal bar. These first results illustrate the intimate relationship between this scheme and k-means estimation of the categories' probability density. As explained earlier, areas of the input space with high densities receive a large portion of the mean vectors available, whereas areas with a low density receive fewer weight vectors.

To summarize the important features of the categorizing module, it can be observed that high-level concepts are captured first and constitute a constraint on what will be learned next. Once high-level concepts have been acquired, low-level relevant features are added to high-level concepts. Typically, frequent categories have more structure represented than less frequent categories, leading to a spectrum of expertise in different conceptual domains (Schyns, 1990).
Expert in Bird.

Figure 12. Conceptual interpretations of the prototypes from different subcategories of the expert in bird. (a) and (b) have been elicited by the prototypes of two subcategories of dog; (c) and (d) have been elicited by the prototypes of two subcategories of bird.

So far, no inclusion relationship between concepts has been directly observed. The categorizing module has just represented different low-level concepts in its weights. I will now show that hierarchical relationships are explicitly coded on the conceptual map to enable the description of categorical judgments with hierarchies of labels.

Figure 12 shows conceptual interpretations of the prototype of four subcategories of the expert in bird. Figure 12a and b show the conceptual interpretation of the prototypes of two subcategories of bird. As noted before, the typicality measure is higher over the region that encodes the two subcategories of bird. Because the subcategories of each category were similar, the different prototypes elicited a strong typicality measure over the region that defines the category as a whole. Therefore, the code for hierarchical relationships can be stated in these terms: A category (e.g., dog) delimits a region of high-typicality response on the conceptual interpretation. Subregions tesselate this region to reflect the typicality of the exemplar according to each subcategory (e.g., pinscher, German shepherd, etc.).
TABLE 4
Mean BSB Reaction Times to Label the Prototypes of the Subcategories with the Different Names of the Hierarchy in Experiment 5

<table>
<thead>
<tr>
<th></th>
<th>Basic</th>
<th>Subordinate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expert in dogs</td>
<td>10</td>
<td>60</td>
</tr>
<tr>
<td>expert in birds</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Novice category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expert in dogs</td>
<td>12</td>
<td>31</td>
</tr>
<tr>
<td>expert in birds</td>
<td>8</td>
<td>58</td>
</tr>
</tbody>
</table>

Conceptual nesting makes the coding of hierarchical relationships explicit. The number of subregions inside a region depends on the number of subcategories the afferent weights have represented. The number of levels in the nesting, that is, the number of levels in the hierarchy, depends both on the complexity of the categorical structure and the amount of resources allocated to represent this category. Consider, as an illustration of this point, the conceptual interpretations of the dog category of the expert in bird displayed on Figure 12c and d. Figure 12c shows a clear conceptual judgment of the prototype. If we refer to Figure 11b, we can see that the expert in bird represents this subcategory of dog in its weights: a T-shaped pattern. However, on Figure 12d, the conceptual interpretation of another subtype of dog elicits a categorical judgment that spreads over all the regions responding preferentially to dog, but which are not highly saturated anywhere, as if the input sample was a dog, but not a specific kind of dog. That is, subordinate categories are not as well distinguished in the conceptual interpretations of novice categories.

Thus, the conceptual map provides a rich code where the input exemplar is evaluated with respect to a hierarchical conceptual structure. Low-level concepts enable the system to cut the external world at its joints more finely, and the internal code reflects this knowledge refinement by richer conceptual interpretations of objects. If this hierarchical structure is truly in the conceptual map, it should be revealed in the naming performance. When the hierarchies of symbols presented in Table 3 were learned, the naming abilities of both experts were tested with the prototypes of the subcategories. As expected, for its category of expertise, the expert in dog could name the prototype of each subcategory it had represented with the adequate superordinate, basic, and subordinate labels (see Table 4). We saw earlier that the expert in bird could categorize exemplars from different subcategories of bird with contrasted conceptual interpretations. However, as it had a label for just one subcategory of bird (i.e., robin), the naming device generalized the instances of all subcategories to one subcategory label, realizing a many-to-one mapping from conceptual interpretations to subordinate labels.
For the novice category, we saw earlier that each system could only represent one or two subcategories. Therefore, their conceptual interpretations were labelled with the correct subordinate label. Because the three other subcategories were not represented clearly in the weight vectors, the conceptual interpretation of their prototypes couldn't be distinguished: The internal codes of the inclusion relationships were not distinct. In these cases, the naming device labelled the exemplars correctly with the superordinate and the ordinate labels. The slot of the subordinate label on f" was either filled with the label of the only represented subcategory of the novice category, or with nothing sensible. Table 4 shows only BSB reaction times of correct subordinate labellings of the ordinate and subordinate categories. The labellings of the superordinate category are not given because the category name was the same across conditions. It should be noted that preliminary experiments have been conducted with categories that were named with different superordinate labels. In these cases, BSB could also retrieve the superordinate symbols correctly. The naming module functions the way it does because it learned to associate a specific conceptual interpretation—a specific internal code—to each symbol in the hierarchy. Subordinate labels were associated with the conceptual interpretations of the exemplars of subordinate categories. The basic-level name was most likely attached to a conceptual interpretation corresponding to the interpretation of the prototype of dog and bird. The superordinate label was simply associated to the entire conceptual map. Note that as it stands, the model presented here doesn't attempt to account for the basic-level effect.

Ultimately, the results presented in all the experiments depend on a mechanism that implements conceptual acquisition by gradually allocating resources—(technically, mean vectors)—as a function of the probability structure of categories. To understand better the functioning of this mechanism in the Kohonen maps context, consider two output units a and b. Assume that these output units are close enough to each other on the map so that they are locally connected with a strong Gaussian strength. Suppose also that a responds maximally to exemplars from Category A and b maximally to exemplars from Category B. After a few iterations of learning, each time output-unit a has its afferent weight vector updated with a proportion of the sample input from A, b is updated with a proportion of the exemplar from A as well. Also, a can be updated with a proportion of the sample input from B. As the weight vector afferent to a and b tend to represent, respectively, A and B, the interference of the updating just described prevents the accomplishment of this goal. For interference to be minimized, that is, for the mean weight vectors underlying output-units a and b to map correctly distinct areas of the data set probability density, the output-units a and b should move out of a critical perimeter of interference. The effect of the interference perimeter can be observed in Figure 8, where the weights...
representing wolf were repelled out of the region encoding dog to a region of the map where interference between categories was less intense. A faster relocation of the region encoding the new category was the essential contribution of top-down influences in Experiment 4.

Now, because the allocation of mean weight vectors is a function of the probability of a particular category, the less frequent categories will be confined in a small region where the subcategories will suffer the each other's updating without being able to move away from the critical perimeter of interference. Conversely, if a large amount of resources is allocated to represent a particular category, as is the case for an expert category, the weight vectors will pick out more and more of the category's internal structure as the neighborhood size diminishes gradually. The depth of the category's structure that can be represented will depend on how quickly interference begins in the representation of subcategories at any level of a hierarchy. This depends on the number of mean descriptor vectors available to represent, or map the distribution of, a category, which in turn depends on how likely the category is.

To summarize, this experiment has illustrated how expertise and hierarchical relationships between concepts could incrementally emerge as a natural consequence of experiencing a structured external world. The pattern of emergence was qualitatively similar to the results of Mervis (1980): Basic-level categories had to be represented before low-level concepts could be acquired. I presented how the categorizing module would give rise to a coding scheme, which would propagate important information to the naming module, and it was seen how that the latter could name different levels of a hierarchy, according to specific constraints. These results agree qualitatively with those of Tanaka and Taylor (1991), and the theoretical hypothesis put forward in Rosch et al. (1976), that experts have more low-level concepts than novices do.

**GENERAL DISCUSSION**

In the article, I have presented a modular model of conceptual development. In order to satisfy the label-concept independence constraint, two functionally independent operations, categorizing and naming, were implemented by two structurally different modules. In each module, specific processes and representations realized the categorical judgments of exemplars and the labelling of conceptual interpretations. In modular systems, knowledge is informationally encapsulated and the modules communicate at their input and output stages. Thus, one of the major goals of this article was to show how a self-organized code—the complete conceptual interpretations—could develop to allow the informational flow to propagate bottom-up from one module to the other one. However, it is clear that top-down influences play an important role in concept learning. Experiment 4 presented an example
of such influence, and the facilitation effect it led to in the encoding of new
categories. The organization of the concept-learning mechanism presented
here can be understood as a data-driven process. When no category name is
available, the bottom-up mode of processing operates autonomously to
extract concepts. When other sources of input are present, for example,
top-down ones, they are integrated on the conceptual map to facilitate the
extraction of a new category. This organization has many characteristics of
modular designs. It should be noted, however, that strict informational
encapsulation is somewhat altered if the output of the categorizing module
is affected by top-down expectations.

Representational issues were of prime concern in this model. We saw that
the conceptual map leads to complete conceptual interpretations of exem-
plars, and those were a rich enough code to represent category membership,
hierarchical dependencies, and a spectrum of expertise in different con-
ceptual domains. This code and the constraints it imposes could be used by
a simple naming device to label exemplars correctly and demonstrate some
interesting psychological effects. These experiments illustrated the theoreti-
cal position that, quite often, concepts may be extracted without external
feedback about the exemplars' category. With this view, symbols either
have the status of indicators of knowledge or facilitators of concept extrac-
tion. In both cases, they are ultimately grounded on conceptual knowledge.

My overall approach to cognitive modelling was resolutely developmental
with an empiricist flavor. I think this kind of approach is particularly fruit-
ful because, among other things, it suggests new forms of representations,
and it allows one to investigate how and why representations constrain
learning in specific ways. For example, the model demonstrated how the
conceptual code built by the categorizing module was constraining correct
learning in the naming module. It also showed how crude concepts were
prerequisites to more specific concepts and the constraint this dependency
would impose on the conceptual code, and consequently, on learning in the
naming module.

The modular architecture presented here raises the issue of capacity and
scaling. Capacity and scaling refer, respectively, to the number of items that
may be learned, and the increase in processing demands that results from
the architecture's expansion. An important argument for the preference of
Kohonen maps over a backpropagation based architecture is the poor scaling
performances of the latter (Blum & Rivest, 1989; Hinton & Becker, 1990;
Judd, 1987). Due essentially to local processing, Kohonen maps scale up
very well because the organization of a large map, or the organization of a
small map, doesn't depend on a global error measure that is backpropagated
to update each and every weight of the network: A small number of weights
—the winner unit's weight vector, as well as the weights underlying the local
neighbors—are updated with a small proportion of the input in the competi-
tive map scheme.
The issue of capacity needs to be considered for the two components of the overall architecture. Kohonen maps are often easier to analyze in their simplest version, when the output units are organized as a one-dimensional string. Consider a set of $n$ orthogonal vector patterns made of $0$s and $1$s that must be encoded on a string of $n$ output units. If the neighborhood size is allowed to shrink to $0$, each orthogonal pattern is recognized with a different output unit. Because the input patterns form an orthogonal set, they elicit $n$ orthogonal output patterns on the string. These orthogonal output patterns may, in turn, be learned exactly with a linear associator. This "best case" argument shows an optimal use of resources: A network of size $n$ can learn $n$ patterns and label them with different category names. When the learning scheme forces neighboring units to be strongly correlated (when the asymptotic value of the neighborhood size is $>0$), and when the data set is not orthogonal but strongly correlated, the capacity of the architecture declines rapidly to some proportion of $n$. However, the arguments already presented should make it clear that, with a large amount of resources, many different concepts and category names can be acquired at a reasonable computational cost.

This simple scheme doesn't allow anything like theory-driven category learning. The reason is that the former results are all based on the assumption that members of the same category are similar to one another. However, important similarity might be in human concept learning, it doesn't seem to have the necessary character presupposed by the architecture I presented (Murphy, in press; Murphy & Medin, 1985). Humans form concepts to represent things that have no a priori similarity to each other. Consider for an example, the category of things to do to be in good health: Having a good diet, getting some exercise, and sleeping enough are activities that have very few properties in common (see Barsalou, 1983). How is it that they are grouped together to give "to be healthy" its meaning? Murphy and Medin argued that people may form categories according to a theory they have about the objects in these categories. These theories would highlight the information that is and is not important to take into account in different situations, thereby maintaining the conceptual coherence of the category. So far, no model of concept learning takes these high-level constraints into account. In the future, more work needs to be done in the direction of finding a way to relate concepts to theories that, in turn, would provide the glue that maintains conceptual coherence by prescribing how to weight features according to the situation considered. However, this is not an easy problem, for our theories are made out of concepts. For example, how would it be possible for a system to have the theory of animacy without the concepts of alive and animal? We have a long way to go before being able to give a non-circular account of high-level knowledge.

However, my model was not intended to account for such complex phenomena. Rather, it was aimed at showing how simple object categories
could be extracted, conceptually refined, and named, much in the way young infants seem to do it when they deal with concrete objects.

REFERENCES


Clark, E.V. (1973). What’s in a word? On the child’s acquisition of semantics in his first
New York: Academic.


Hilledale, NJ: Erlbaum.

example of extensional programming. In N.E. Sharkey (Ed.), Advances in Cognitive


Journal of Mathematical Psychology, 30, 73-80.

proach. Unpublished doctoral dissertation, Psychology Department, Brown University,
Providence, RI.

to classify sonar targets. Neural Networks, 1, 75-89.

development and coding of neural feature detectors. Biological Cybernetics, 23,
121-124.

87, 1-51.


Gruendel, J. (1977). Referential extension in early language development. Child Develop-
ment, 48, 1567-1576.


J.A. Anderson (Eds.), Parallel models of associative memory. Hillsdale, NJ: Erlbaum.


Carnegie Mellon University, Pittsburgh, PA.

in random-dot stereograms. Proceedings of the International Joint Conference on
Neural Network, 1, 218-222.


information on the abstraction from ill-defined categories. Journal of Experimental
Psychology: Human Learning and Memory, 1, 351-359.

and classification of new instances as a function of the number of instances defining the

Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computa-

Hopfield, J.J. (1984). Neurons with graded response have collective properties like those of
two-states neurons. Proceedings of the National Academy of Sciences, 81, 3088-3092.


