Formal Approaches in Categorization

Emmanuel M. Pothos and Andy J. Wills

Editors

CAMBRIDGE
UNIVERSITY PRESS

11 COBWEB models of categorization and probabilistic concept formation

Wayne Iba and Pat Langley

Description of the model

In this chapter, we describe a family of integrated categorization and category learning models that process and organize past experience to facilitate responses to future experience. The COBWEB system (Fisher, 1987) and its descendants CLASSIT (Gennari, 1990), OXBOW (Iba, 1991), LABYRINTH (Thompson & Langley, 1991), DÆDALUS (Langley & Allen, 1993), and TWILIX (Martin & Billman, 1994) comprise a family of models that share a genealogy, a search strategy, and a heuristic to guide that search. We will often refer to this entire family as COBWEB when the intended meaning is clear from the context.

These systems grew out of machine learning and cognitive science research that explored methods for acquiring concepts in an unsupervised context. In that setting, a teacher does not provide explicit category information for instances as the learner encounters them; instead, the learner must decide how to group or categorize a collection of instances. In contrast to most clustering methods, instances are encountered *incrementally*; the learner must make appropriate adjustments in response to each one as it comes. The COBWEB family of models view categorization as a conceptualization process or as the formation of ontologies.[1] That is, these models provide answers to the question, 'How does one form conceptual representations of similar experiences and how might those representations be organized?' However, these models also address the *use* of the acquired concepts to handle future situations. The framework embodies a collection of assumptions that together provide constraints on the design and implementation of computational systems for ontology formation and use. Such implemented models make it possible for agents to efficiently process new experiences and respond in an effective manner (for some definition of effectiveness).

[1] We use ontology in its narrow philosophical sense and do not intend the connotations associated with common knowledge frameworks and agent interchange languages.

Assumptions

The COBWEB models we describe here have emerged within the context of several assumptions. These assumptions may be grouped as philosophical, psychological and computational.

Philosophically, a conceptualization or ontology provides a means for agents to make sense of their experiences. We assume that conceptualizations conforming to the actual world will provide performance advantages over alternative conceptualizations. This emphasis on pragmatic benefits suggests that the contents or substance of concepts, as well as their organization and relationships, are important issues for these methods. Note that, although a particular conceptualization represents certain ontological commitments on the agent's part (Gruber, 1995), the systems need not be concerned with the metaphysical situation as long as the adopted ontology provides pragmatic value when responding to novel stimuli.

These models also grew out of several psychological assumptions. Following a grand tradition in artificial intelligence, we treat human cognition as an inspiration for the design of computational processes that exhibit intelligence. For example, if we believe that humans employ partial matching in categorization, then we would be wise to include a similar mechanism in our computational models. Likewise, cognitive limitations observed in humans should provide constraints on implementations in computer systems. If humans learn incrementally (subject to their ability to reprocess previous stimuli), these models should do the same. Although alternative methods not subject to such constraints may perform in some respects more effectively than methods that are subject to them, we know that, at a minimum, the constraints found in humans do not preclude intelligent behaviour or, in this case, the formation and use of conceptualizations. As stated by Fisher and Pazzani (1991, p. 26), 'principles which dictate preferred concepts in humans are good heuristics for machine concept formation.'

Several computational assumptions have influenced the design and development of COBWEB models. First, we recognize that nearly all of the concepts that humans form exhibit an imprecision that reveals itself, for example, when trying to pin down the meaning of words. This imprecision makes exact reasoning and communication a challenging task, but this imprecision also allows concepts considerable flexibility in their application to novel situations. So from our existence proof in humans we conclude that imprecise reasoning is sufficient for intelligence. From the way humans powerfully use the flexibility of such imprecision, we assume that such reasoning might prove to be necessary for intelligence.

Computationally, partial matching enables flexible reasoning about the membership of given objects with respect to alternative categories. Probability theory provides support for some types of partial matching, as well as an account for some of the imprecision in our language. Thus, COBWEB models share a commitment to probabilistic representation of concepts.

Second, we note that hierarchies are efficient data structures for storing and retrieving information and that humans frequently think about concepts in terms of 'is-a' or 'a kind of' hierarchies. In order to access one of n nodes organized in an appropriately structured hierarchy, we need only incur $\log_b(n)$ comparisons for uniform branching factor b. Even if it proves to be the case that humans do not organize concepts within hierarchies but only introspect as if they did, the computational benefit remains. So COBWEB models also assume that the probabilistic concepts they form are organized into some form of hierarchical structure. Furthermore, we assume that such hierarchies have the property that, when branches relate a parent to various children, the parent generalizes the more specific child concepts.

Finally we remember that conceptualizations provide value only in so far as they facilitate the processing of new experiences in an effective manner. It is only within the context of some performance task that a concept hierarchy proves its value. In one form or another, COBWEB models all rely on *flexible prediction* as their performance task (Fisher, 1987). Generally, we may think of flexible prediction as a measure of the average predictive power derived from a conceptualization. For a given object, flexible prediction measures the average predictive accuracy on unseen attributes. When evaluating learned concept hierarchies, we take a test instance and withhold each attribute, comparing the model's prediction of the missing attribute to the withheld value.[2] The average accuracy on these individual predictions provides the measure of flexible prediction. These models implicitly seek to maximize this average predictive ability.

Details of the model

We are now ready to describe the common elements of this family of methods and point the reader to other sources that explain significant variations. The model's core consists of three elements: the internal representation of concepts, search operators for exploring the space of

[2] In the limit, we withhold every possible non-empty proper subset of attributes and ask the model to predict all missing ones.

conceptualizations, and the heuristic evaluation function for guiding the search. These three elements are employed in a uniform manner for both performance and learning. COBWEB uses the same process for forming conceptualizations as it does for categorization or flexible prediction. The difference between learning and performance is the object of interest – the new conceptualization as a whole in the first case, and the values of a specific concept in which a test instance is classified in the second.

Representing conceptualizations Concepts, organized within hierarchies, serve as the basis for categorization in COBWEB models. A given concept subsumes or summarizes a number of child concepts that are each more specific than it. For example, the concept 'bird' may have subconcepts 'robin', 'goldfinch', and 'turkey'. Although some COBWEB models maintain a strict tree hierarchy and others allow a directed graph structure,[3] each concept is a child of a more general concept and in turn has some number of more specific child concepts below it. The exceptions are the root of the concept hierarchy, which has no parent, and leaves, which are maximally specific concepts corresponding to single instances or collections of indistinguishable instances.

These models represent individual concepts probabilistically. Each node in a conceptualization summarizes the previously observed objects deemed to have been members of that particular category. The system summarizes these instances by estimating the probability distributions for each attribute under an assumption of conditional independence. The respective values observed for a given attribute form the estimate of the probability distribution for that attribute. If the attribute is numeric instead of symbolic, one approach is to model it as a mean and standard deviation under the assumption that its range of values satisfies a normal distribution. As will be seen, the model also needs to know the prior probability of a category; it stores a count of the number of objects classified at each category and estimates its base rate as this count divided by the total number of objects in its parent.[4]

All COBWEB models represent the attributes of concepts in a probabilistic fashion along these lines. However, two of the methods, LABYRINTH and OXBOW, additionally model the components that comprise structured stimuli and treat these components as some form of attributes along with the regular attributes. For example, OXBOW forms concepts

[3] For example, TWILIX allows a given concept to participate in multiple contexts, effectively giving it multiple parent concepts (Martin & Billman, 1994).

[4] Technically, this is the conditional probability that an instance belongs in a particular class given that it belongs in the parent class.

of movement skills and includes probabilities on components of the skill, treating a component as a kind of attribute. Alternatively, LABYRINTH might represent chairs as having a seat component and one or more legs that support the seat. These particular models point to the variety of representations that are possible within COBWEB's framework of probabilistic hierarchies.

Searching for conceptualizations The manner in which these models form hierarchies of probabilistic concepts may be viewed as a search through the space of such hierarchies. Specifically, COBWEB conducts this search in an incremental hill-climbing manner, locally maximizing the quality of the hierarchy as described below. Steps between states involve either modifications to the particular concepts, modifications to their hierarchical organization, or a combination of both. Updates to concepts reflect changes to probability estimates, and revisions to the hierarchy itself reflect the addition, merging, or splitting of concepts. Drawing inspiration from the incremental character of human concept learning, COBWEB models consider local revisions to the current concept hierarchy based on single instances.

For a given training stimulus, i, and a node into which the stimulus has been categorized (initially the root), COBWEB evaluates several alternatives regarding the concepts c_k that are the immediate children of the given node. First, the system considers adding the new instance, i, to the child concept, c_k, into which it fits best according to the search heuristic described below. Second, if the new instance is sufficiently unlike all of the existing children, the system creates a new singleton concept with the instance and adds the new concept as a sibling of the existing child concepts.

Because order effects can sometimes cause COBWEB to mistakenly add new singleton concepts or classify new instances as existing concepts, the model employs two structural revision operators that simulate a form of backtracking. Thus, as a third alternative, the system considers merging the two best-fitting child concepts and adding the new instance to this merged concept. Finally, it considers splitting the best child concept, promoting its children to the current set of children, and adding the new instance to the best over the resulting set of child concepts. These two structural revision operators do not reprocess previously encountered examples but instead depend on probabilistic summaries stored at the current level of the hierarchy. Given the four alternative conceptualizations arising from these search operators, the system selects among them according to its evaluation function, as discussed shortly.

Recursively, the system repeats this comparison between competing classifications until it creates a leaf node with the new instance (second alternative above). Thus, a new instance triggers a sequence of updates to probabilistic representations of concepts, possibly interspersed with revisions of the local hierarchical structure, finally culminating in the addition of a new singleton concept.

Guiding the search for conceptualizations The manipulations described above occur under the guidance of a heuristic evaluation function that measures the quality of the children of a given concept in the hierarchy. Gluck and Corter (1985) propose *category utility* as a numeric measure of how good or useful a particular partition is with respect to the parent concept. Fisher (1987) generalized Gluck and Corter's formula to multiple categories as:

$$\mathrm{CU}(C) = \frac{1}{K}\left[\sum_{k=1}^{K} P(C_k \mid C)\sum_{i}\sum_{i} P(A_i = V_{ij} \mid C_k)^2 - \sum_{i}\sum_{j} P(A_i = V_i \mid C)^2\right] \tag{1}$$

where K is the number of classes at the current level of the hierarchy, $P(C_k \mid C)$ is the base-rate probability that an instance belongs to the child class given it belongs to the parent class C, $P(A_i = V_{ij} \mid C_k)$ is the probability that attribute i will have value j given that we know the instance belongs to category C_k, which is summed first over the possible values of attribute i, and then over all attributes i. Similarly, $P(A_i = V_i \mid C)$ is the same summed quantity when we only know that the instance belongs to the parent class. This formula applies to nominally valued attributes; for continuous attributes, the corresponding generalization (Gennari, 1990) is defined as:

$$\mathrm{CU}(C) = \frac{1}{K}\left[\sum_{k=1}^{K} P(C_k \mid C)\sum_{i}\frac{1}{\sigma_{ik}} - \sum_{i}\frac{1}{\sigma_{ip}}\right] \tag{2}$$

where σ_{ik} is the standard deviation of attribute i in child class C_k and σ_{ip} is the standard deviation of the attribute in the parent class C. Using this metric, COBWEB evaluates the quality of alternative revisions to this part of the hierarchy and selects the best alternative. Unless a new singleton class is created, the process is repeated with the selected node as the nominal root and its children as the partition over the objects that have previously been classified here.

Again, note that COBWEB models tightly couple performance and learning. The process of classifying an instance updates the probability distribution of the class where the instance ultimately comes to rest, as well as all the distributions of concepts along the path back to the root concept. Additionally, the process of classifying the instance may trigger local revisions to the hierarchy along that path. Thus, we might think of classification (performance) taking place by incorporating (learning) the instance to be predicted, and learning taking place by classifying a newly observed instance (at each step of the process, alternative classifications are considered and the best forms the basis for updates).

Motivation

Most generally, a simple recognition that most human learning takes place in non-teaching environments underlines the need to understand unsupervised category learning. By the time children master several hundred words, they have learned a vast quantity of information about the way their world works. The knowledge they accumulate spans objects they have experienced through one or more (often all) of the senses, motor skills they have refined for moving through and manipulating their environment, and basic reasoning skills for accomplishing their goals (or more often getting others to fulfil them). Infants and toddlers acquire most of this knowledge through what corresponds to the unsupervised learning process described above. Thus, we want to explore methods by which such accumulation and organization of knowledge can take place. By actually implementing such methods, we verify their practicability and utility. The model and assumptions we have described above grow out of prior inclinations or sensibilities that situate and motivate our model-building activities.

Philosophical sensibilities

The conceptualization process fundamentally entails the formation of a world model. Philosophically, we assume the value of such conceptualizations lies solely in their predictive advantage. For two alternative conceptualizations of prior experience, an intelligent agent should prefer the one that makes relevant predictions more accurately. Thus, the ontological commitments implicit in a conceptualization refer to the utility of the model rather than to the way the world 'actually is'.

Note that reference to utility carries with it a notion of value. Clearly, the ability to predict some features is more valuable than others. For example, seeing a four-legged animal, we have much higher utility from accurately predicting the animal's likelihood of attacking and harming us than in accurately predicting its body temperature. Most of the COBWEB models treat all attributes as having equal utility. However, Martin and Billman (1994) have explored relaxing this assumption by maintaining weights that they associates with features as a representation of their respective values.

Psychological sensibilities

Human behaviour and cognition provide an unending source of phenomena to understand and explain, but conceptualization is particularly important. A long tradition of artificial intelligence work has sought insights from psychology to fuel the implementation of computational models and research in the COBWEB family is a clear instance of this tradition.

We assume that insights from human concept formation will tend to provide useful implementation constraints. Humans, as examples of intelligent agents, motivate and justify this assumption. Note that humans process experiences incrementally, have a limited capacity to remember and reprocess these experiences, and reason imperfectly about them. If we consistently and faithfully designed with respect to the patterns and constraints observed in humans, then our models should exhibit important aspects of intelligent behaviour. However, we also assume that intelligence is not a uniquely human characteristic. Thus, we view human cognition as providing soft constraints that may be ignored as needed.

Computational sensibilities

In addition to the sensibilities identified above, we also expect that if COBWEB captures characteristics of human intelligence, then it can also function as a practical tool. That is, we want it to process data and provide results in a reasonable amount of time. Our choice of modelling concepts with probabilities provides an example of tensions between sensibilities. Probability theory nicely models certain aspects of intelligence. Dretske (1999) has argued that probability and information theory provides a formal basis for understanding knowledge. However, we know that humans do not reason according to Bayes-optimal principles. Thus, our models employ a probabilistic representation without a commitment to such reasoning. Given our primary interest in forming reliable models that allow agents to operate effectively, probabilistic representations provide a natural, efficient and well-grounded foundation.

In the case of our choice to organize concepts in hierarchies, our psychological and computational sensibilities work in harmony. In computer science applications, the tree serves as a common and well-used data structure. Although other structures have superior characteristics along particular dimensions, the tree excels in its combination of

Table 11.1 *Generic algorithm for the family of COBWEB models*

```
// for parent concept $p$ and new instance $i$
classify($p$, $i$)
 if $p$ is a leaf and has no children,
  create copy of $p$ as new-$p$;
  create singleton node from $i$ as new-$i$;
  update $p$'s probabilities based on $i$;
  add new-$p$ and new-$i$ as children of $p$;
 else with each child $c_k$ of $p$
  select best of these four according to CategoryUtility($p$)
   1. add $i$ to each child $c_k$ of $p$;
     rank order their resulting partitions;
   2. add $i$ as new singleton child of $p$;
   3. merge the best and second-best children from 1;
     add $i$ to merged result;
   4. promote children of best child to be children of $p$;
     add instance to best of children;
  unless 2 is selected, continue with classify($c_k$, $I$)
   where $c_k$ is the child that gets $i$.
```

flexibility and efficiency. For example, a hash map provides faster access to stored items, but it would not store two related concepts near each other. On the other hand, a tree organizes concepts so that similar concepts are close and provides efficient access to them. Furthermore, this structure increases the expressivity of the simple conditional probability scheme described above by embedding implicit conditional assumptions based on a concept's chain of parents.

Implementing COBWEB

Having described the motivations and the general outlines for the model, we can provide the details necessary for a rational reconstruction. To implement a COBWEB model, one must define: (1) the data structures for representing probabilistic concept hierarchies, (2) a function for updating the estimate of the probability distribution in response to a new instance, (3) the category utility function for deciding among the alternatives that are generated by (4) the four search operators that revise the hierarchy. Table 11.1 presents the schematic algorithm for the family of models we have discussed.

Representing and updating probabilistic concepts

Hierarchies consist of concept nodes, which in turn consist of prior probabilities for the concept itself, symbolic names, attributes and their values, conditional probability distributions over these attributes and values, and a collection of concept nodes representing the more specific concept nodes that are children of the given concept. Because COBWEB works incrementally, processing one instance at a time, it updates its probability distributions without reprocessing previous instances. The model represents base-rate probabilities by storing the counts of instances classified at each concept; the ratio of the count at the child to the count of the parent represents the base-rate probability of the child concept.

The treatment of probability distributions over attributes depends on whether the attribute is nominally or continuously valued. To maintain the probability distribution of a nominal attribute, one need only tabulate a total count for the attribute and counts for each observed value for that attribute. The ratio of a particular value's count to the number of times that attribute was observed provides the conditional probability of the attribute having that value given that the instance is a member of the class. We can have the model process continuous attributes either by employing some discretization method and treating values as nominal or by estimating the density as a normal distribution. When choosing the latter, the model maintains two running sums – one for the values of the attribute and the other for the sum of the squared values. With these two sums it can incrementally compute the standard deviation by expanding the expression from the traditional definition.

Revising the concept hierarchy

Table 11.1 identifies four alternative updates that a COBWEB model must generate and evaluate. The first two – adding the new instance to the best child and creating a new singleton concept – present no significant implementation issues. The operation described above for updating probabilistic concepts takes care of the first and a simple addition of a new concept node to the collection of child concepts handles the second. However, experience suggests that the merge and split operators require considerable development and debugging time.

The merge operator inputs two existing concepts and creates a new concept. The system determines the instance count for the new concept by adding the counts of the two concepts to be merged; similarly, the system computes the attribute models in the merged concept by adding the counts from corresponding attributes when nominally valued and by adding sums and sum of squares when continuously valued. The new merged concept contains a collection of child concepts created by combining the children of the two concepts being merged.

When splitting a concept, the system removes the concept to be split from the collection in which it occurs, then adds to that collection each child of the split node. Because these children comprise the summary information found in the deleted node, no counts or models need to be updated. We assume that, when computing category utility, the correct size of the collection is used as the discount factor, $1/K$, in Equation 1.

Other details

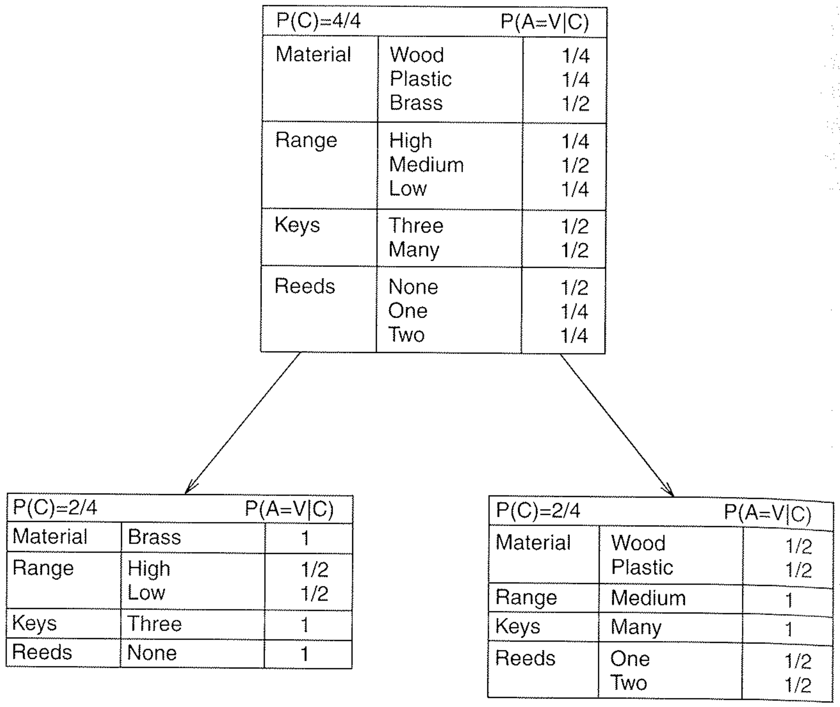
For both splitting and merging, the implementer should take care that, when creating the alternative local revisions to the concept hierarchy, she either maintains an original copy or provides reversibility of all operators. Creating copies of the hierarchy for each candidate might at first seem to be a memory-intensive operation. However, with the exception of the split operator, the revisions only impact the immediate context – the parent and its children. Thus, we need not copy more specific layers of the existing hierarchy as long as we handle the split operator properly.

Another detail arises in situations where one decides to estimate the probability distribution of a numeric attribute using a mean and standard deviation. For such cases, one must allow for singleton concepts where the standard deviation on attribute values will be zero and consequently the category utility will be undefined. A response employed by several COBWEB models introduces an acuity parameter, which represents the minimum variance for an attribute. This parameter bears similarity to the psychological notion of a *just noticeable difference*, the quantity by which two attribute values must differ in order to be distinguishable.

On a related note, the implementer may choose to include a cutoff parameter that serves as an alternative termination to the recursive algorithm presented in Table 11.1. Either for efficiency reasons (space or time) or because of cognitive limitations in humans, the classification process may reach a point where further subdivision into more specific concepts provides no additional predictive benefit. One natural approach to implementing the cutoff depends on the gain in information between the parent and the partition of children; when the gain falls below some predetermined cutoff value the system stops making sub-classes.

An example of COBWEB's operation

A simple example may clarify the operation of COBWEB models. Suppose our system is learning about musical instruments for the first time. In this scenario, assume the system has encountered four instruments:



P(C)=4/4		P(A=V|C)
Material	Wood	1/4
	Plastic	1/4
	Brass	1/2
Range	High	1/4
	Medium	1/2
	Low	1/4
Keys	Three	1/2
	Many	1/2
Reeds	None	1/2
	One	1/4
	Two	1/4

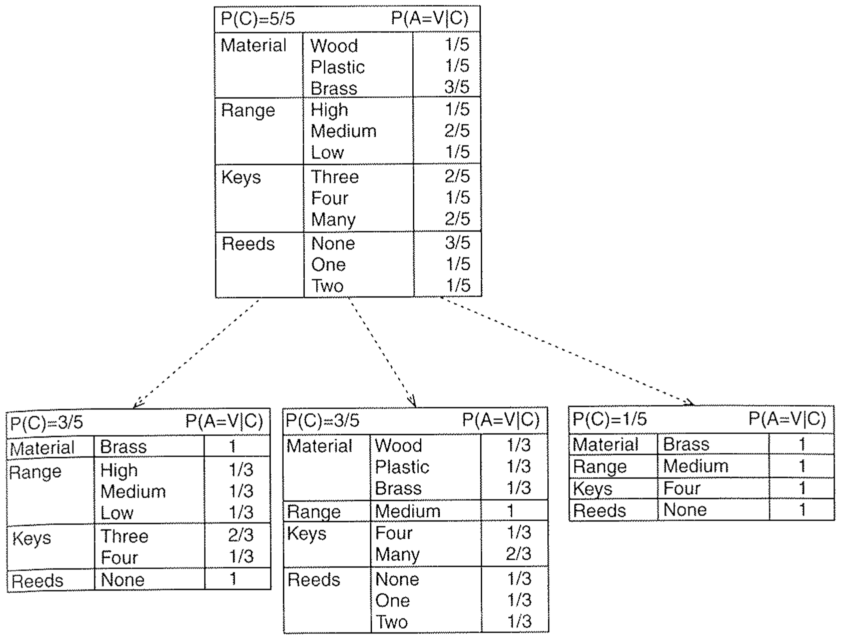
P(C)=2/4		P(A=V|C)
Material	Brass	1
Range	High	1/2
	Low	1/2
Keys	Three	1
Reeds	None	1

P(C)=2/4		P(A=V|C)
Material	Wood	1/2
	Plastic	1/2
Range	Medium	1
Keys	Many	1
Reeds	One	1/2
	Two	1/2

Figure 11.1 A possible concept hierarchy that captures probabilistic knowledge of musical instruments in two categories – brass and woodwind.

an oboe, a clarinet, a trumpet, and a tuba. Instruments are described by four attributes: the primary material of which they are constructed (wood, brass or plastic),[5] their musical range (low, medium or high), the number of keys for forming different pitches (three, four or many), and the number of reeds (none, one, two or many). The trumpet and the tuba are made of brass, have three keys, no reeds, and have a high and low musical range, respectively. The oboe and the clarinet both have many keys and a medium range, but the oboe is made of wood and has two reeds while the clarinet is made of plastic and has only one reed. Let us suppose that, based on these four musical instruments, the system has formed the concept hierarchy as shown in Figure 11.1, where the hierarchy consists of two classes corresponding to our traditional notions of brass and woodwind instruments.

[5] Here we use 'plastic' to stand for composite products that are occasionally used in the construction of less expensive woodwind instruments.



P(C)=5/5		P(A=V|C)
Material	Wood	1/5
	Plastic	1/5
	Brass	3/5
Range	High	1/5
	Medium	2/5
	Low	1/5
Keys	Three	2/5
	Four	1/5
	Many	2/5
Reeds	None	3/5
	One	1/5
	Two	1/5

P(C)=3/5		P(A=V|C)
Material	Brass	1
Range	High	1/3
	Medium	1/3
	Low	1/3
Keys	Three	2/3
	Four	1/3
Reeds	None	1

P(C)=3/5		P(A=V|C)
Material	Wood	1/3
	Plastic	1/3
	Brass	1/3
Range	Medium	1
Keys	Four	1/3
	Many	2/3
Reeds	None	1/3
	One	1/3
	Two	1/3

P(C)=1/5		P(A=V|C)
Material	Brass	1
Range	Medium	1
Keys	Four	1
Reeds	None	1

Figure 11.2 A rendering of multiple alternative updates to the concept hierarchy as a result of encountering a French horn. The root concept has been updated and one of the three alternatives (or merge or split) would be selected as the basis for a new partition consisting of the modified class and the original classes.

At this point, assume we show the system a particular French horn, which has the standard three valves plus a fourth rotary valve. COBWEB incorporates the new instrument (brass, medium, four, none) into the root of its conceptualization in a straightforward manner, updating the counts on attribute values. Next, the model decides between the various alternative operators described earlier. First, it considers adding it to the left-hand class corresponding to brass instruments. As a result, the prior probability on the class node becomes 3/5. COBWEB also updates the conditional probabilities for each attribute-value given the brass class. In the case of the material and reeds attributes, these probabilities do not change, as the French horn has the same values. However, it differs from the other brass instruments encountered in that the French horn's musical range attribute value is medium and it has four keys instead of three. We show the new probabilities for the category in the lower-left concept in Figure 11.2. Note this figure does not show an actual hierarchy but rather the alternative revisions to the respective classes that would result from incorporating the new instrument and the possible third class created from the single French horn. That is, the three child classes in Figure 11.2 represent the mutually exclusive updates for finding the best child and creating a new singleton (steps 1 and 2 in the algorithm given in Table 11.1). The system evaluates the partition that results from the respective updates together with the other original classes, having updated only their prior probabilities.

With the alternative updates in hand, COBWEB must determine which alternative to prefer according to the category utility metric. The computation[6] from Equation 1 reveals that the partition resulting from the addition of the French horn to the brass class has a score of 44/30; the new prior for the class, 3/5, times the sum of squared conditional probabilities 26/9, plus the score for the existing woodwind class (with updated prior) 2/5 times 3, all discounted by 1/2 for the number of classes in the partition. Similarly, the score for adding the new instance to the woodwind instruments is the score from the original brass class with the prior update (2/5 times 7/2) plus the updated woodwinds class shown in the middle-bottom of Figure 11.2 (3/5 times 20/9) for a total score of 41/30, after applying the discount of 1/2. Finally, the option for creating a third class consisting of just the French horn has an overall score of only 34/30. In the last case, note that although the singleton class has high conditional probabilities, the low base-rate for the new class 1/5 and the larger discount penalty for the extra class (1/3 instead of 1/2) lower this alternative's partition score substantially.

Because there are only two classes in the initial concept hierarchy, COBWEB would not consider merging them as described earlier. Given that the brass class is the best option, the system would finally consider promoting its two children, the singleton trumpet and tuba that it started with, and then check again to determine into which class it would be best to add the French horn. The overall score of this split option is 32/30, also a relatively low result. Consequently, the French horn would be placed with the other brass instruments (as we intuitively expected all along) and continue the process recursively.

This example portrays the concept formation process at an early stage of learning. As more examples are observed over time, the hierarchy becomes wider and deeper. Concepts near the top of the hierarchy summarize a large number and a wide variety of instruments observed

[6] We have omitted subtracting the parent's value of 42/25 because this same amount is subtracted from each option. The parent's value plays a significant role only when implementing a cutoff mechanism.

by the system; concepts near the leaves of the tree summarize a small number of very similar instruments. In the musical instruments domain, we could expect COBWEB to form two top-level concepts for brass and woodwinds, with the woodwinds concept having three children, for single-reed instruments like the clarinet and various types of saxophones, double-reed instruments such as the oboe, and non-reed woodwinds like the flute.

This simple example should convey the core method employed by this family of concept learning systems. However, several issues bear mentioning. First, some attribute information may be missing from instances. That is, the system may not initially perceive all of the features that describe observed objects. For example, many people would not take notice of the presence or absence of a spit-valve on brass instruments. But when the system starts noticing such a feature, COBWEB updates counts on the individual attributes and modifies the category utility function accordingly. Second, instead of simple attribute-value representations, objects may consist of components and these components may be structurally related. In general, this structural information cannot be compressed into an attribute-value format. In such cases, we must extend the representation and evaluation function employed by the model. Also, the control structure must be augmented in some way to confront the partial matching problem. Iba's (1991) OXBOW addressed both of these issues by forming classes of observed and practised motor skills. Likewise, Thompson and Langley's (1991) LABYRINTH system extended COBWEB to learn concepts that describe complex objects and relations among their parts. Finally, one may relax the constraint that a concept hierarchy must be a tree and instead allow a directed acyclic graph. In this context, parent concepts still summarize their children but a given child may be summarized by multiple parents. Martin and Billman (1994) employed this strategy in their TWILIX system, thereby allowing it to learn concepts that participate in multiple partitions. For example, the concept for 'bugle' might appear as a child of the 'brass' concept, and also as a child of a 'signal mechanism' class.

Relation to other models

Now that we have reviewed the COBWEB class of models, let us consider their similarities and differences from other approaches presented in this volume and elsewhere. We can view these relations along several dimensions: the representation and organization of categories, how categorizations are formed and used, and motivational commitments.

Representing and organizing categories

Most models for representing categories rely on either exemplars, prototypes, or some hybrid of them. We can view COBWEB as a hybrid, as it forms a hierarchy with concepts at higher levels that resemble prototypes and leaves of the hierarchy that serve as exemplars. The SUSTAIN model (Chapter 10) provides a similar hybrid. Like COBWEB, it starts with exemplars and lets them evolve into prototypes. Exemplars in SUSTAIN become more abstract in a context-sensitive manner guided by stimuli; this lets prototypes and exemplars co-exist and compete at the same level, potentially leading to interesting interactions. In principle, COBWEB could have both exemplars and prototypes in the same collection of concepts, but typically they are stratified by depth in the hierarchy. SUSTAIN represents categories in a connectionist framework but does not provide a hierarchy for them.

COBWEB'S probabilistic representation of concepts also bears similarities to Griffiths *et al.*'s (Chapter 8) mixture models for density estimation. In COBWEB, each concept provides an estimate of the probability distribution over the instances classified by it. Both systems support a unification of the prototype and exemplar models, COBWEB assigns instances to multiple categories along the classification path spanning multiple levels of the hierarchy, whereas the mixture models assign instances to multiple categories at the same level. The approaches also differ in their performance methods, with COBWEB using the category utility function grounded in information theory and the nonparametric models described by Griffiths *et al.* employing Bayesian methods.

COBWEB organizes its categories in a hierarchy of increasingly specific clusters. Most of the models presented in this volume do not address the organization of categories – hierarchical or otherwise. Pothos, Chater, and Hines (Chapter 9) describe the simplicity model, whose agglomerative method could be extended to hierarchies in a natural and straightforward manner. Griffiths *et al.* describe a hierarchical model in which clusters may participate in multiple categories, although this hierarchy has little resemblance to the multi-level concept hierarchy produced by COBWEB. Anderson and Matessa (1991) present a hierarchical version of their model that bears similarities to the COBWEB framework. Feigenbaum's (1961) EPAM system also forms hierarchical structures, although concepts appear at the leaves and internal nodes influence classification of novel stimuli.

Historically, Fisher's (1987) COBWEB grew out of ideas found in earlier systems such as UNIMEM (Lebowitz, 1982), CYRUS (Kolodner, 1983), and EPAM. To our knowledge, COBWEB was the first model of

categorization and category learning to employ probabilistic representations for incrementally learning concepts organized within a general-to-specific IS-A hierarchy. The synthesis represented by COBWEB and its direct descendants stimulated a number of exciting approaches that have fruitfully branched in other directions (e.g., Anderson & Matessa, 1991; Cheesman *et al.*, 1988; Griffiths *et al.*, Chapter 8).

Using and learning categories

Many models of categorization utilize measures of similarity between stimuli to guide their classification and, where applicable, their formation of conceptualizations. These similarity measures are often based on distance metrics. In contrast, COBWEB uses probability estimates stored in concepts at each level of its hierarchy to categorize a new example and updates the estimates for the concept to which it is assigned. The use of category utility in this estimation serves as a unifying feature of the COBWEB family of methods.

Like most of the models in this volume, COBWEB forms its conceptualization in an incremental fashion. Such methods must contend with order effects resulting from non-representative sequences of stimuli. In fact, order effects in humans are such a fundamental assumption that experimental regimes always attempt to control for them. Pure exemplar models (e.g., Nosofsky's GCM, Chapter 2) may be sensitive to a peculiar initial sample of stimuli, but in the long run order effects should not impact such methods. Likewise, prototype models (e.g., Minda & Smith, Chapter 3) would not notice order effects in the long run if applied in a supervised learning context. However, hybrid or prototype models that incrementally *form* categories (e.g., Ashby *et al.*'s COVIS, Chapter 4, and McDonnell and Gureckis's SUSTAIN, Chapter 10) will be susceptible to such effects. SUSTAIN provides mechanisms for adding clusters but apparently does not have an operator for backtracking from an ill-advised cluster. Presumably the impact of such choices can be eliminated over time by adjustment of weights.

An earlier relative of COBWEB, McKusick and Langley's (1991) ARACHNE, attempts to eliminate order effects by adopting an alternative search heuristic. Instead of using category utility to guide the search for a hierarchy of probabilistic concepts, it uses structural properties of the hierarchy to guide the addition of new categories or the revision of existing concepts via merging and splitting. In general, the resulting structural revisions are more extensive than COBWEB'S. Empirical results demonstrated that ARACHNE was less sensitive to noise in the domain and more adept at utilizing background knowledge. Which method provides a more realistic model of order effects in human learning remains a topic for future research.

Guiding motivations

The models presented in this volume span a range of commitments in both their details and their motivations. Some present a rational model for categorization irrespective of its value as a model of human category learning or performance (e.g., Griffiths *et al.*, Chapter 8). Many others focus on explaining specific phenomena from constrained recognition and choice selection tasks (e.g., Ashby *et al.*'s COVIS, Chapter 4). Models in the COBWEB family fall between, addressing problems that we might expect an agent to encounter when trying to make sense of, navigate, and manipulate its environment, while also attempting to explain high-level psychological phenomena. For example, COBWEB provides an account of the basic level (see also Pothos *et al.*'s simplicity model, Chapter 9), typicality effects, and the power law of learning. Taken together, the similarities and differences between COBWEB and other models underline areas of active research and areas where further work will be helpful.

Directions for future research

COBWEB models of categorization and category learning account for a wide range of psychological phenomena, including basic and typicality effects, fan effects, and the power-law of learning (Fisher & Langley, 1990; Iba, 1991). We have successfully applied these models in several domains. Thus, several directions of research hold promise for fruitful investigations and insightful results. Broadly, these involve two lines of work – improving the model itself and applying the model.

It is widely established that as concepts become well established they become more resistant to change. This raises the issue of how COBWEB reorganizes its concept hierarchy over time. Should the hierarchy become less plastic with increasing experience? Certainly, in clusters of concepts near the root of the hierarchy, new examples will be unlikely to become a new singleton concept. At the leaves, the structure will be more fluid regardless of how much experience is captured in the hierarchy as a whole. Because new examples trigger merging and splitting, and because the influence of a single instance near the top of the hierarchy will be minimal later in training, such reorganizations should take place more frequently near the leaves. Future work should evaluate and characterize the reorganizations according to level and timing.

It is also well established that the order in which a learner observes stimuli significantly impacts what is learned. Another line of exploration would look at order effects. Clearly, humans are influenced by the order of training stimuli and we should try to characterize the nature of those effects (Clapper & Bower, 2002; Langley, 1995). We can use COBWEB to articulate hypotheses regarding the order effects that we would expect to find in humans. Given a specific task with, say, three categories of stimuli, we could present examples for two of the categories and vary the onset for introducing instances from the third category. This evaluation would aim to characterize the influence of novel but meaningful stimuli as a function of their position in the training sequence.

As an implemented system, some components exist not as commitments of the model but to make things work; if such components cannot be integrated with the model they should be pruned. An example of this appears in MÆANDER, which Iba (1991) developed to acquire and improve motor skills through observation and practice. The approach employed inner concept hierarchies within each motor skill concept; these private hierarchies captured the temporal structure of a given motor program. This approach should be generalized so that a single hierarchy organizes both complete motor skills as well as their temporal components.

In addition to investigating model features, COBWEB systems can be applied to various problems of practical value. For example, MÆANDER has been employed to analyse telemetry from the NASA space shuttle (Iba, 1993) and to the early detection of faults in industrial pumps. Also, many extensions to the basic COBWEB system were designed to support abilities required by intelligent agents operating in physical environments, including categorization and learning with continuous attributes (Gennari, Langley, & Fisher, 1989), structured objects (Thompson & Langley, 1991), motor skills (Iba, 1991), and plan knowledge (Langley & Allen, 1993). We believe the application of models to real-world problems has the benefit of refining their explicit and implicit assumptions. The successful applications of COBWEB models to such tasks and the richness of their accounts of psychological phenomena, demonstrate the potential of the theoretical framework.

REFERENCES

Anderson, J. R., & Matessa, M. (1991). An incremental Bayesian algorithm for categorization. In D. H. Fisher, M. J. Pazzani, & P. Langley (eds.), *Concept Formation: Knowledge and Experience in Unsupervised Learning*. San Mateo, CA: Morgan Kaufmann.

Cheeseman, P., Kelly, J., Self, M., Stutz, J., Taylor, W., & Freeman, D. (1988). AUTOCLASS: A Bayesian classification system. In *Proceedings of the Fifth International Conference on Machine Learning* (pp. 54–64). Ann Arbor, MI: Morgan Kaufmann.

Clapper, J. P., & Bower, G. H. (2002). Adaptive categorization in unsupervised learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, **28** (5), 908–923.

Dretske, F. (1999). *Knowledge and the Flow of Information*. Palo Alto, CA: CSLI Press.

Feigenbaum, E. (1961). The simulation of verbal learning behavior. In *Proceedings of the Western Joint Computer Conference* (pp. 121–132). Reprinted in J. W. Shavlik & T. G. Dietterich (eds.) (1990). *Readings in Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, **2**, 139–172.

Fisher, D. H., & Langley, P. (1990). The structure and formation of natural categories. In G. H. Bower (ed.), *The Psychology of Learning and Motivation: Advances in Research and Theory* (Vol. 26). Cambridge, MA: Academic Press.

Fisher, D. H., & Pazzani, M. J. (1991). Computational models of concept learning. In D. H. Fisher, M. J. Pazzani, & P. Langley (eds.), *Concept Formation: Knowledge and Experience in Unsupervised Learning*. San Mateo, CA: Morgan Kaufmann.

Gennari, J. H. (1990). *An experimental study of concept formation*. Doctoral dissertation, Department of Information & Computer Science, University of California, Irvine, CA.

Gennari, J. H., Langley, P., & Fisher, D. H. (1989). Models of incremental concept formation. *Artificial Intelligence*, **40**, 11–61.

Gluck, M., & Corter, J. (1985). Information, uncertainty and the utility of categories. In *Proceedings of the Seventh Annual Conference of the Cognitive Science Society* (pp. 283–287). Irvine, CA: Lawrence Erlbaum.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, **43**, 907–928. Available on-line.

Iba, W. (1991). *Acquisition and improvement of human motor skills: learning through observation and practice*. Doctoral dissertation, Department of Information & Computer Science, University of California, Irvine, CA.

(1993). Concept formation in temporally structured domains. In *NASA Workshop on the Automation of Time Series, Signatures, and Trend Analysis*. Moffett Field: NASA Ames Research Center.

Iba, W., & Langley, P. (2001). Unsupervised learning of probabilistic concept hierarchies. In G. Paliouras, V. Karkaletsis, & C. D. Spyropoulos (eds.), *Machine Learning and its Applications*. Berlin: Springer.

Kolodner, J. L. (1983). Reconstructive memory: a computer model. *Cognitive Science*, **7**, 281–328.

Langley, P. (1995). Order effects in incremental learning. In P. Reimann & H. Spada (eds.), *Learning in Humans and Machines: Towards an Interdisciplinary Learning Science*. Oxford: Elsevier.

Langley, P., & Allen, J. A. (1993). A unified framework for planning and learning. In S. Minton (ed.), *Machine Learning Methods for Planning and Scheduling*. San Mateo, CA: Morgan Kaufmann.

Lebowitz, M. (1982). Correcting erroneous generalizations. *Cognition and Brain Theory*, **5**, 367–381.

Martin, J. D., & Billman, D. O. (1994). Acquiring and combining overlapping concepts. *Machine Learning*, **16**, 121–155.

McKusick, K. B., & Langley, P. (1991). Constraints on tree structure in concept formation. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence* (pp. 810–816). Sydney: Morgan Kaufmann.

Thompson, K., & Langley, P. (1991). Concept formation in structured domains. In D. H. Fisher, M. J. Pazzani, & P. Langley (eds.), *Concept Formation: Knowledge and Experience in Unsupervised Learning*. San Mateo, CA: Morgan Kaufmann.