Machine Learning, 16, 121–155 (1994)

Acquiring and Combining Overlapping Concepts

JOEL D. MARTIN martin@cs.pitt.edu
Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15260

DORRIT O. BILLMAN billman@pravda.cc.gatech.edu
Department of Psychology, Georgia Institute of Technology, Atlanta, GA 30332

Editor: Michael Pazzani

Abstract. This article presents OLOC, an incremental concept formation system that learns and uses overlapping concepts. OLOC learns probabilistic concepts that have overlapping extensions and does so to maximize expected predictive accuracy. When making predictions, OLOC can combine multiple overlapping concepts.

Keywords: concept formation, overlapping concepts, conceptual combination, incremental algorithms

1. Introduction

With little supervision, people can divide their world into categories and use those categories to predict missing information or to comprehend novel inputs. Furthermore, people classify instances into overlapping categories: Fido is both a pet and a dog (Osherson & Smith, 1982); soccer is both a game and a sport (e.g., Hampton, 1987). Existing unsupervised concept formation methods (e.g., COBWEB, Fisher, 1987) capture much of the flavor and qualitative specifics of human category learning. However, such methods often assume that every instance is a member of exactly one category at each level of a hierarchy of categories. In this paper, we present an unsupervised learning algorithm that learns about and uses overlapping categories.

Several well-known concept formation systems learn hierarchies or flat sets of categories by classifying each instance into only one category at each level of the hierarchy (e.g., Fisher, 1987; Anderson, 1990). In these systems, a *category* is any set of instances and a *concept* is an internal representation of that category. The learner classifies instances into categories and manipulates the corresponding concepts by creating, changing, or deleting them. Existing concept formation systems were designed around results from cognitive psychology, such as descriptions of a basic, or maximally predictive, level of categories (see Gluck & Corter, 1985). Inspired by psychological research, they were then tested as psychological models and produced impressive quantitative matches with many studies of human category learning (Anderson, 1990).

A major assumption of these approaches, that instances are classified into one category at any level in the hierarchy, is somewhat counterintuitive and inconsistent with other psychological research. People appear to have overlapping categories and, as an example, can classify the dolphin as both an aquatic animal and a mammal. Two or more categories are said to *overlap* when they share instances but neither category is a subset of the other. The `mammal` category and the `aquatic animal` category overlap because

they share instances; yet, not all mammals are aquatic and not all aquatic animals are mammals.

People readily combine information from overlapping concepts, that is concept representations whose corresponding categories overlap (e.g., Murphy, 1990; Hampton, 1987, 1988; Jones, 1982). Indeed, Osherson and Smith (1981, p. 55) state that "the ability to construct thought and complex concepts seems to lie near the heart of human mentation." Humans appear to combine concepts by computing an internal representation of the combined concept using the internal representations of the constituent concepts (e.g., Smith & Osherson, 1984). This fundamental ability is used for many related purposes: understanding noun phrases (e.g., Osherson & Smith, 1982), understanding categories in context (Roth & Shoben, 1983), deriving ad-hoc categories that are relevant to current goals (Barsalou, 1983), and determining the point of view of other individuals (Barsalou & Sewell, 1984).

Besides the evidence that humans use overlapping concepts, there are also computational reasons both to learn and to use such concepts. First, with overlapping concepts, the same knowledge can often be represented more succinctly. This is the traditional advantage of multiple classification for the representation of knowledge. For example, if we know about 100 animal species that can be either pets, wild animals, or performers, we would prefer to represent this information with 100 distinct animal concepts, a pet concept, a wild animal concept, and a performer concept. Without overlapping concepts, a system would need 300 separate concepts to represent the distinctions between all possible combinations.

A second computational reason to use overlapping concepts is that doing so accelerates learning when overlapping concepts best represent the knowledge in question. A learner who can recognize overlapping categories can divide a complex learning problem into two or more simpler learning problems and thereby learn using fewer examples. For example, the world may contain pets, wild animals, performers, cats, dogs, and horses. Without overlapping concepts, the learner must see a few examples of every combination of a role (pet, wild, performer) and animal type to have an adequate model of the world. Alternatively, when overlapping concepts are possible, the learning problem can be decomposed. Now, the learner must still see a few examples of pets, wild animals and performers to have an adequate model of the roles animals can play. The learner must also see a few examples of each animal type to have a model of those animals; however, the same example can be used as an instance of both an animal role and an animal type. Thus, when learning overlapping concepts, the learner need not see examples of every combination. For example, the learner need never have encountered a pet horse to have a model of such a thing.

Despite these potential benefits, learning overlapping concepts has not been widely addressed in the concept formation work from the machine learning community (cf. Lebowitz, 1987). Many models of concept formation do not form overlapping concepts at all. We present OLOC (One-level for Learning Overlapping Concepts), a concept formation system that learns overlapping concepts, and we specify a simple but informative rule for combining such concepts once learned.

Table 1. A probabilistic concept

	Probability = 0.4					
Attributes	*Values with Probabilities*					
Found	inside	(0.83)	outside	(0.17)		
Food-Type	packaged	(0.83)	fresh	(0.17)		
Mobility	swims	(0.42)	walks	(0.29)	flies	(0.29)
Covering	scales	(0.42)	hair	(0.29)	feathers	(0.29)
Legs	zero	(0.42)	two	(0.29)	four	(0.29)
Reproduction	fertilizes	(0.33)	produces-eggs	(0.67)		
Appearance	plain	(0.5)	pretty	(0.5)		

In the next two sections, we will present OLOC. In Section 2, we describe the representation and organization of concepts. Also, we describe how OLOC uses such concept structures for classification and prediction. Section 3 then presents the learning method that OLOC uses to construct structures of overlapping concepts. Section 4 demonstrates that OLOC can learn successfully in both artificial and natural domains and can outperform learning systems that do not acquire overlapping concepts. Section 5 briefly presents an interesting, though limited, set of findings from human conceptual combination and shows how OLOC exhibits the same qualitative pattern of findings. Section 6 summarizes the design of OLOC, assesses its promise as a cognitive model, and presents directions for future research.

2. Concept representation and use

OLOC's organization of overlapping concepts is novel, as are its methods for using and learning such concepts. Each concept in this structure is probabilistic and those concepts representing mutually exclusive categories are grouped together into sets called *contrast sets*. Concepts in different contrast sets represent overlapping categories. OLOC classifies instances into one or more overlapping categories and can use the corresponding concepts to generate predictions.

2.1. Representation of concept structures

Concepts in OLOC specify the information needed to compute the utility those concepts. A concept in OLOC is probabilistic, such that predictive associations between categories and attribute values are represented as probabilities (cf., Smith & Medin, 1981; Fisher, 1987). Gluck and Corter (1985) and Fisher (1987) used a probabilistic measure of the utility of a contrast set of categories. This measure required that the representation of a category, C_k, contain at least the probability that an instance is classified into the corresponding category, $P(C_k)$, and, for each attribute value, $A_i = V_{ij}$, the conditional probability of that attribute value given classification into the corresponding category $P(A_i = V_{ij}|C_k)$. Concepts in OLOC have the same form as those used by Fisher (1987). An example of a concept is shown in Table 1. As in Fisher's (1987) system COBWEB, OLOC's concepts explicitly store observed frequencies, e.g., the number of

instances classified into a category. These frequencies are used to estimate probabilities. In particular, the estimated probability of classification into a category, $P(C_k)$, is the number of instances classified into that category, $F(C_k)$, divided by the total number of instances, N.

$$P(C_k) \approx \frac{F(C_k)}{N} \tag{1}$$

With continued random sampling, this estimate converges on the expected value of the probability of classifying an arbitrary instance into the category.

Furthermore, the estimated conditional probability, $P(A_i = V_{ij}|C_k)$, of a value given classification into a category is a function of the number of instances classified into that category, $F(C_k)$, and the number of those instances that contained the value in question, $F(A_i = V_{ij}|C_k)$. In COBWEB (Fisher, 1987), this function is a simple division of frequencies as for the $P(C_k)$; however, with small samples, such a calculation can be grossly misleading. We use a more conservative approximation (cf., Anderson & Matessa, 1991) namely,

$$P(A_i = V_{ij}|C_k) \approx \frac{1 + F(A_i = V_{ij}|C_k)}{n_{A_i} + F(C_k)} \tag{2}$$

In this equation, n_{A_i} is the number of values of the attribute, A_i. This estimated probability converges on the expected value of the true probability. Further, this approximation provides a way of exploiting additional knowledge about the values for a particular attribute. OLOC does not require these values, but when they are known, better estimates of probabilities are possible.

OLOC's probabilistic concepts are organized into multiple contrast sets. The categories represented in one contrast set are mutually exclusive, such that no instance can be classified into more than one category in the set. The concepts in different contrast sets, i.e., overlapping concepts, are independent. In other words, knowing to which category an instance was classified in one contrast set does not provide any new information about which category is best in another contrast set. By assuming independence, OLOC attempts to minimize redundant categories between contrast sets.

OLOC allows learning of contrast sets, sets of non-overlapping concepts, for two reasons. First, classification into multiple categories can be more efficient if OLOC knows which categories contrast with a given one. Once an instance is classified into one category, OLOC need not consider adding the instance to any of its contrasting categories. Second, there is evidence that humans build and maintain contrast sets of concepts and use those sets for prediction (e.g., Rosch, 1978).

2.2. *Multiple classification and concept combination*

OLOC, like many other concept formation systems, supports two basic functions on concept structures: classification and prediction. Classification consists of choosing one or more appropriate categories for a given instance. For example, a description of

a goldfish might be classified into a `fish` category instead of a `mammal` category because the description of the goldfish is a better match to the description of `fish`.

Once the instance is classified into one or more categories, the system may use statistical information from the corresponding concepts to predict characteristics of a novel instance. For example, a description of a particular shark may not explicitly state that that shark has a 2-chamber heart. To predict the type of heart the shark has, the system first classifies the shark based on what it does know, such as that this animal lives in the water, etc. Once the instance is classified as a `fish`, the system can then predict that the shark has a 2-chamber heart based on the knowledge that most known fish have two-chamber hearts.

Classification of instances into overlapping categories and the combination of corresponding concepts for prediction requires different methods than those used for trees of categories. Trees of non-overlapping categories are a familiar part of biology and other sciences. Living things are classified into exactly one kingdom (e.g., animal or plant), one phylum (e.g., mammal, bird, fish, etc.), etc. In a tree of categories, classification is based on the familiar idea of a best-match: A bluejay is a better match to other birds than it is to reptiles so it is classified as a `bird`. Unfortunately, a single best-match is not a sufficient means for allowing multiple classification.

Furthermore, the path of increasingly specific concepts in a tree (e.g., Animal, Mammal, Cow, etc.) can be combined simply based on the familiar idea of inheritance: a bluejay breathes like other animals, lays eggs like other birds, and sings like other songbirds. Whenever characteristics inherited from different concepts conflict, the more specific of the two concepts overrides the more general. For example, birds fly, but a penguin is a bird that cannot. In this case, the more specific category `penguin` overrides the more general, `bird`. The combination of overlapping concepts is not so simple, because the concepts to be combined are not ordered explicitly by any relation such as specificity. For example, when an activity is classified as both a sport and a game, it is not clear, without an analysis of content, which of the two concepts should dominate, in which respects, when they are combined.

2.2.1. Classification

In OLOC, multiple classification is performed by finding several best-matching concepts and doing so sequentially. An instance is first classified into the category whose concept representation best matches the instance, then a new best match is determined. Later best-matches are influenced by earlier classifications as described below. This process continues until a combination of all the selected concepts is sufficient to determine all attribute values (algorithm in Table 2). For example, a shark may be an overall better example of a `fish` than it is of a `mammal` or a `performer` or a `pet`. It is first classified into the category `fish`. This classification permits predictions about breathing method and other gross physiological characteristics. However, several characteristics, such as behavior, habitat, and food, are less predictable. Given that it is a `fish`, it cannot also be a `mammal` but it can be a `performer` or a `pet`. The shark is then

classified into whichever of the remaining categories whose concept representation best matches the instance.

In OLOC, the determination of which of several categories is the best match is determined by a Utility score. This utility score is a probabilistic measure that is correlated with the probability of a category given an instance. In other words, the more probable the category given the instance, the higher the utility score and the better the match. The exact measure used is motivated by issues encountered when learning concept structures and is discussed in detail when those issues are addressed below.

```
Function M-Classify (Instance, Root); Returns Combination
   LET Combination be a copy of Root concept;
   Let Sets be the list of contrast sets;
   WHILE not PREDICTABLE(Instance, Combination) OR
         not IMPROVE(Instance, Combination)  DO
       FOREACH ContrastSet in Sets
           FOREACH Concept in ContrastSet
                   Add Instance to Concept;
           Calculate Utility using Combination;
                   Remove Instance from Concept
           END
       END;
       LET BestConcept be Concept producing the highest Utility;
       Remove the contrast set containing BestConcept from Sets;
       LET Combination be COMBINE(Combination, BestConcept);
   END WHILE;

Function COMBINE (Concept1, Concept2) ; Returns Result
   /* Distribution Maximizing */
   LET Result be a copy of Concept1
   FOREACH Attribute
      IF most-probable-value(Attribute, Concept2) is more proba-
ble than
         most-probable-value(Attribute, Concept1)
      THEN
         replace the distribution for Attribute in Result
         with the distribution for Attribute in Concept2.
```

Table 2. The algorithm for multiple classification

2.2.2. *Combining concepts*

A significant component of both classification and prediction in OLOC is the combination of multiple overlapping concepts. OLOC must calculate the probability of an attribute value given a subset of concepts, $P(A_i = V_{ij}|C_1, C_2, \ldots)$. If concepts in different contrast sets are independent, this quantity can be calculated as follows:

$$P(A_i = V_{ij}|C_1, C_2, \ldots) = P(A_i = V_{ij}) \prod_k \frac{P(A_i = V_{ij}|C_k)}{P(A_i = V_{ij})} \quad (3)$$

Although OLOC does assume independence, concept combinations should be robust when this assumption is not precisely true. Equation 3 does not meet this requirement. First, when the concepts are not independent, this calculation will yield estimated probabilities for each value of some attribute that do not sum to one. Second, when the concepts are interdependent, the probability of a value given the combined concept varies widely. It is either considerably larger or smaller than the probability of that value given either of the concepts that were combined. We have chosen to use a measure without these two characteristics. When concepts are independent, our calculation produces the same result as does Equation 3.[1] However, when concepts are not independent, it yields a more stable estimate.[2]

OLOC combines two concepts by considering each attribute separately and allowing the concept with the greatest certainty about the attribute's value to dominate in the combination. For example, if one concept predicts that body covering of an animal is `feathers` with a certainty (probability) of 0.99 and another concept predicts that body covering is `hair` with a certainty of 0.7, OLOC accepts the description of the more certain concept. It then expects the body covering to be `feathers` with a certainty of 0.99.

More specifically, for each attribute, OLOC compares the existing distributions from the two concepts and adopts one of the two distributions unchanged. A *distribution* is a list of the probabilities for each observed value of an attribute. For example, the distribution for the attribute 'body covering' in one concept might be (feathers, 0.99);(hair, 0.01);(scales, 0.0). For another concept, the corresponding distribution might be (feathers, 0.2);(hair, 0.7);(scales, 0.1). OLOC determines which of the two concepts has the distribution with the largest maximum probability (the one with the most certainty about some attribute value) and adopts that concept's distribution for the resulting combination. In this example, OLOC observes that the maximum probability for any value in the first distribution (0.99) is greater than the maximum probability for any value in the second distribution (0.7). As a result, it adopts the first distribution unchanged. OLOC's combination method is called *distribution maximizing* and is further described in the lower half of Table 2.

The distribution maximizing approach is similar but not identical to that advocated by Holland, Holyoak, Nisbett, and Thagard (1986). They suggest that a property with a low "degree of variability" for a concept should more highly influence the combined concept than should a property with a high degree of variability. A property whose value does not vary much will also have a high, maximum probability. The color of fire engines has a low degree of variability, meaning that the probability of the most common color (red) is nearly 1.0.

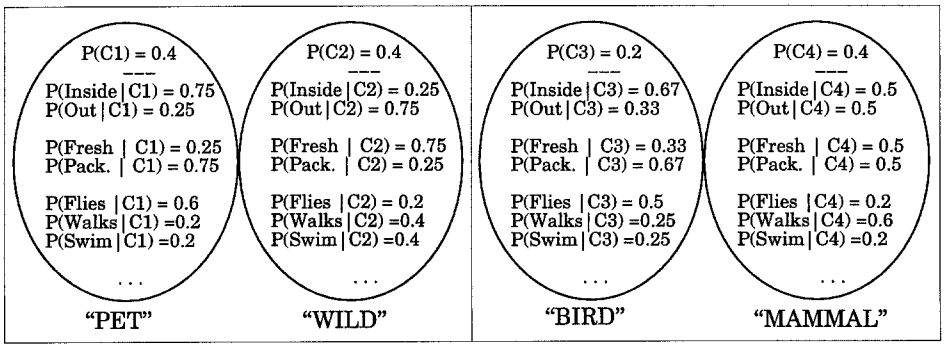

Figure 1. An overlapping concept structure with two contrast sets and four concepts.

2.3. *An example of classification and prediction*

We now consider an example classification of an instance and the subsequent prediction of missing attribute values. Figure 1 depicts an overlapping concept structure. The two rectangles represent contrast sets and the ovals represent concepts. The statistical information stored in the concepts is printed in the ovals. These concepts are not sharply defined but they approximate categories with the labels shown in quotes beneath each oval: pets, wild animals, birds, and mammals. These labels are not provided to OLOC.

OLOC multiply classifies an instance by first classifying it into the best category overall and then in light of the first classification, classifies it into the best remaining category. Consider a hypothetical robin that lives outside, has feathers, two legs, fertilizes eggs, and appears pretty. The utility scores that result from adding the instance to each category are as follows: 1.105 for adding it to the `bird` category, 1.053 for adding it to the `mammal` category, 0.925 for adding it to the `wild` category, and 0.911 for adding it to the `pet` category. The most probable category given the instance is the `bird` category. The instance is first classified into that category and the `mammal` category is no longer considered, because the concept structure states that an instance cannot be both a `bird` and a `mammal`. Moreover, a combined concept description is calculated. Because the instance has only been classified into one category, the combined description is the same as the `bird` concept (see Table 3).

Of the two remaining categories, `pet` and `wild`, the most probable one given the instance and the prior classification into `bird`, is the `wild` category. The utility scores that result from adding the instance to each category are as follows: 1.162 for adding it to the `wild` category and 1.083 for adding it to the `pet` category. The instance is classified into the wild category and a combined concept description is calculated. OLOC temporarily combines the two concepts to increase the chances of a correct prediction (see Table 4).

Table 3. The predictive description of the robin after classification as a "bird."

Attributes	*Values with Probabilities*					
Found	inside	(0.67)	outside	(0.33)		
Food-Type	packaged	(0.67)	fresh	(0.33)		
Mobility	swims	(0.25)	walks	(0.25)	flies	(0.5)
Covering	scales	(0.25)	hair	(0.25)	feathers	(0.5)
Legs	zero	(0.25)	two	(0.5)	four	(0.25)
Reproduction	fertilizes	(0.67)	produces-eggs	(0.33)		
Appearance	plain	(0.67)	pretty	(0.33)		

Table 4. The combined description of the robin after classifying it both as a "bird" and as a "wild" animal.

Attributes	*Values with Probabilities*					
Found	inside	(0.25)	outside	(0.75)		
Food-Type	packaged	(0.25)	fresh	(0.75)		
Mobility	swims	(0.25)	walks	(0.25)	flies	(0.5)
Covering	scales	(0.25)	hair	(0.25)	feathers	(0.5)
Legs	zero	(0.25)	two	(0.5)	four	(0.25)
Reproduction	fertilizes	(0.25)	produces-eggs	(0.75)		
Appearance	plain	(0.25)	pretty	(0.75)		

An important characteristic of this combined description is that the result preserves extremes. The `bird` concept strongly predicts `flies` and `feathers` and that information was preserved in the combination. Similarly, the `wild` concept strongly predicts `outside` and `fresh` food and that information was also preserved in the combination.

Once the robin is classified as a `bird` and a `wild` animal and the combined concept description has been calculated, OLOC can predict the values of unspecified attributes. In this instance, the values of neither `Food-Type` or `Mobility` are specified. Using the combined description in Table 4, OLOC chooses the most probable value for each unknown attribute. It predicts that `Food-Type` is `fresh`, as for most wild animals, and that `Mobility` is by `Flying`, as for most birds.

In general, then, OLOC multiply classifies an instance and as it does so, it generates a combined concept description relevant to the instance. This combined concept description can then be used to predict the values of unspecified attributes.

3. OLOC: Learning overlapping concepts

OLOC is an unsupervised, incremental concept formation system. It is incremental, because instances are incorporated into the concept structure as they are observed. It is unsupervised, because there is no teacher directing learning to enhance prediction of one or a few specific attributes.

We describe OLOC's learning as a hill-climbing search through possible concept structures (cf., Fisher, 1987). In OLOC's hill-climbing search, there are *operators*; i.e., processes that can transform one concept structure into another. The learner chooses which operator to perform based on the Utility score described above, a heuristic evaluation

function. The Utility score is probabilistic and estimates the *predictive accuracy*; i.e., the probability of producing correct predictions of missing attribute values. Therefore, OLOC searches for a structure of overlapping concepts that maximizes predictive accuracy.

In this section, we describe OLOC's learning algorithm by presenting:

- The measure of predictive accuracy that is to be maximized, and
- The operators and control strategy for altering concept structures.

This description is followed by a detailed example of its operation in section 3.3. Some readers may wish to examine the example before reading about the algorithm in general.

3.1. *A measure of predictive accuracy*

The goal of OLOC is to find the concept structure, θ, that maximizes the probability of correct predictions, $P(correct|\theta)$. OLOC considers concept structures as described above such that instances are organized into contrast sets, and concepts in one contrast set overlap with those in other contrast sets.

The overall estimated probability of correct predictions is calculated as a geometric average of the probability of being correct given each contrast set. In turn, the probability of being correct given one contrast set is calculated by performing a weighted sum across the concepts in that contrast set. In particular, the estimated probability of being correct is calculated with the following measure,

$$U(\theta) = P(correct|\theta) = \sqrt[M]{\prod_{l=1\ldots M} SetUtility(S_l)} \tag{4}$$

$$SetUtility(S_l) = \sum_{\{C_k \in S_l\}} P(C_k|\theta) \sum_i \sum_j P(A_i|\theta) P(A_i = V_{ij}|C_k, \theta)^2 \tag{5}$$

In this equation, M is the total number of contrast sets and $P(A_i|\theta)$ is the probability that the learner will need or want to predict the value of attribute, A_i, given a particular concept structure.

In OLOC, this latter probability is a constant equal to $\frac{1}{number\ of\ attributes}$. In Equation 5, the term $P(C_k|\theta)$ is the probability that an arbitrary instance will be classified into the k-th category in a particular concept structure θ. Finally, the term $P(A_i = Vij|C_k, \theta)$ is the probability that an instance classified into the k-th concept will contain the j-th value of the i-th attribute.

To calculate this Utility score, OLOC need only have access to the probability of concepts and to the conditional probabilities of values given concepts. Both types of quantity are stored in concepts. In the Appendix, the Utility score in Equation 4 is shown to be equal to the estimated probability of being correct given certain assumptions.

3.1.1. *Advantages of the utility score*

Besides being a direct measure of predictive accuracy, the Utility score has three additional desirable aspects. First, because the Utility score calculates a geometric average of a utility score for each contrast set, concept structures with differing numbers of contrast sets can be productively compared.

Second, the contribution of the individual contrast sets to the overall measure can be calculated and updated independently. This means that local changes to a single contrast set do not require a complete recalculation of the Utility score. Additionally, it means that given an appropriate architecture, the Utility score can be calculated in parallel.

Third, the Utility score favors contrast sets that balance the traditional goals of clustering: high within-class similarity and low between-class similarity (cf., Fisher, 1987). In probabilistic terms, a concept has high within-class similarity if the probabilities of attribute values given the concept, $P(A_i = V_{ij}|C_k)$ are high. If these quantities are high, then most instances of the category will share the same attribute values. Conversely, concepts have low between-class similarity if the probabilities of concepts given attribute values, $P(C_k|A_i = V_{ij})$, are high. If these quantities are high, then most instances in one category will have different attribute values than do other concepts. Fisher (1987) demonstrates that one measure for balancing these quantities,

$$\sum_{k=1}^{n} \sum_{i} \sum_{j} P(A_i = V_{ij}) P(C_k|A_i = V_{ij}) P(A_i = V_{ij}|C_k) \tag{6}$$

can be rewritten as,

$$\sum_{k=1}^{n} P(C_k) \sum_{i} \sum_{j} P(A_i = V_{ij}|C_k)^2 \tag{7}$$

(cf., Gluck & Corter, 1985). Without the normalizing constant $P(A_i|\theta)$, Equations 5 and 6 would calculate the same quantity.

3.2. OLOC*'s operators and algorithm*

To describe learning as a hill-climbing search, we must specify the operators that can transform one concept structure into another. At each point in the search—as each instance arrives—the learner chooses the operator that most increases the Utility score. OLOC can transform a concept structure in three ways[3]: a) updating the representation of multiple overlapping concepts by adding an instance to a subset of those concepts; b) creating a new concept in an existing contrast set; c) and creating a new contrast set with one new concept. At each point in learning, OLOC attempts to find the operator or operators that will most improve the Utility score.

Ideally, OLOC would tentatively add the instance to every subset of concepts to determine the best. However, this is not practical, because the number of subsets of concepts is an exponential function of the number of concepts.

Instead, OLOC performs a greedy search for one set of overlapping concepts, such that if the instance were classified into all the concepts in this subset, the Utility score would be high.

For each instance encountered, OLOC does the following:

1. Considers all concepts in all contrast sets,
2. Adds the instance to the one best concept (across all remaining contrast sets), creates a new concept, or creates a new contrast set,
3. Removes from consideration the contrast set containing that best concept, and
4. Iterates through steps 2 and 3 until there are no gains in predictive accuracy.

In this algorithm, no gain in predictive accuracy means that the number of attribute values successfully predicted does not increase. An attribute value in an instance is predictable if that attribute value is the most probable value of the attribute given a concept or set of concepts. Table 5 outlines this algorithm in more detail.

3.2.1. Maintaining independent contrast sets

The above algorithm is not sufficient to ensure that the structure learned consists of independent sets of mutually exclusive concepts. Both step 2 and step 4 must be enhanced. First, step 2 does not guarantee contrast sets with multiple concepts. To address this problem, OLOC requires that the second instance added to a contrast set be used to create a new, second concept in the contrast set. In some cases, this approach will lead to similar instances being classified into different categories. Although this problem can produce unnecessary concepts, in practice, it has not lead to poorer prediction performance. Below, we discuss an extension to OLOC that directly addresses this problem.

Second, the iteration in step 4 does not guarantee that the concepts in different contrast sets will be independent. To best approximate independence, OLOC records how well attribute values are predicted by a subset of concepts and favors classification into those additional concepts that would improve the poorly predicted attribute values. Specifically, after the instance has been classified into a concept, that concept's description is combined with all other concepts before determining the next classification (Table 2 describes the combination algorithm). Whenever the Utility score is recalculated to determine additional overlapping concepts, it is calculated using the combined concept description. For example, a particular shark may be first classified as a fish. Subsequent classification decisions are then made in light of that first classification. If the shark might be classified as either a performer or a wild animal, OLOC first combines the fish concept with the performer concept and combines the fish concept with the wild animal concept. These combinations are only temporary and help determine classification. Classification is then made into `performer` or `wild` based on the better of the two combined concepts.

```
Function OLOC (Instance, Root); Returns Combination
   Update Root concept using Instance;
   LET Combination be a copy of Root concept;
   IF there are zero contrast sets
   THEN    Create a contrast set with one new Concept;
           Update Concept using Instance;
           LET Combination be COMBINE(Combination, Concept);
   ELSE    Let Sets be the list of contrast sets;
           WHILE not PREDICTABLE(Instance, Combination) OR
                 not IMPROVE(Instance, Combination) DO
              LET BestConcept be BEST-CONCEPT(Sets, In-
stance, Combination);
              LET CreateConcept be new concept in a new con-
trast set;
              IF BestConcept gives a higher Util-
ity than CreateConcept
              THEN    IF BestConcept is new THEN create it;
                      Update BestConcept using Instance;
                      Remove the contrast set containing BestCon-
cept from Sets;
                      LET GlobalBest be BestConcept
              ELSE    Create a contrast set with one new Concept;
                      Update CreateConcept using Instance;
                      LET GlobalBest be CreateConcept
              END IF;
              LET Combination be COMBINE(Combination, GlobalBest);
           END WHILE
   END IF;

Function BEST-CONCEPT (Sets, Instance, Combination); Re-
turns BestConcept
   FOREACH ContrastSet in Sets
      IF there is more than one concept in ContrastSet
      THEN FOREACH Concept in ContrastSet
              Add Instance to Concept;
              Calculate Utility using Combination;
              Remove Instance from Concept
      END
   END;
   FOREACH ContrastSet in Sets
      Create a new concept in ContrastSet;
      Add the Instance to the concept;
      Calculate Utility using Combination;
      Remove the new concept
   END;
   LET BestConcept be Concept or new concept producing the high-
est Utility
```

Table 5. OLOC's algorithm

Table 6. Instance animal descriptions

Name	*Found*	*Food-Type*	*Mobility*	*Covering*	*Legs*	*Reproduction*	*Appears*
Finch	inside	packaged	Flies	feathers	two	produces-eggs	plain
Angel-Fish	outside	fresh	Swims	scales	zero	produces-eggs	pretty
Macaw	inside	packaged	Flies	feathers	two	fertilizes	plain
Hamster	inside	packaged	Walks	hair	four	fertilizes	plain
Leopard	outside	fresh	Walks	hair	four	produces-eggs	pretty
Pigeon	outside	fresh	Flies	feathers	two	fertilizes	plain
Goldfish	inside	packaged	Swims	scales	zero	produces-eggs	pretty
Guppy	inside	packaged	Swims	scales	zero	fertilizes	plain
CatFish	outside	fresh	Swims	scales	zero	produces-eggs	plain
Gopher	outside	fresh	Walks	hair	four	fertilizes	plain

3.3. An example

Figure 2 shows the process of learning four instances from Table 6. Each snapshot shows the multiple classification of each instance and the concept structure that is learned. Each illustration of a concept structure consists of a collection of contrast sets (the rectangles) and concepts within the contrast sets (the ovals). Under each concept is a list of the instances that have been classified into the corresponding category.

For this example, we assume that OLOC begins with no concept structure. After encountering the first two instances in Table 6 ('finch' and 'angel-fish'), OLOC's concept structure is as in Figure 2a.

In Figure 2a, there is only one contrast set, represented in the figure as a rectangle. There are two concepts, represented by ovals. Under each concept is a list of instances that have been classified into the corresponding category. Each concept in the contrast set has a probability, which is the probability that an instance will be classified into the corresponding category. The concept also has conditional probabilities of attribute values given the concept. For simplicity, these probabilities are omitted from the figure.

Figure 2b shows the process of learning about the 'macaw.' The dotted arrows represent the action of attempted classification and the solid black arrows represent the action of updating a concept structure. Learning in OLOC follows the algorithm in Table 5. For each of the two concepts in the only contrast set, the instance is temporarily added to the concept and the Utility score is calculated. Next, a new concept (dotted oval) is temporarily created in the contrast set and again the Utility score is calculated. Finally, a new contrast set is created with a new concept and once more, the Utility score is calculated. The operator that yields the highest Utility score is then adopted. In this example, the Utility score for adding to the first concept, C_1, is 0.513; the Utility for adding to the second concept, C_2, is 0.469; the Utility for creating a new concept in the contrast set is 0.478; and the Utility for creating a new contrast set is 0.225. Therefore, the best operation is to add the instance to the first concept, C_1. Hence, the 'macaw' instance is classified into the same concept as was 'finch.'

After adding the instance to one concept, OLOC tests whether the number of predictable attributes is not rising or that each attribute value in the instance is the most probable value for the concept. In the first case, additional classification of the instance does not

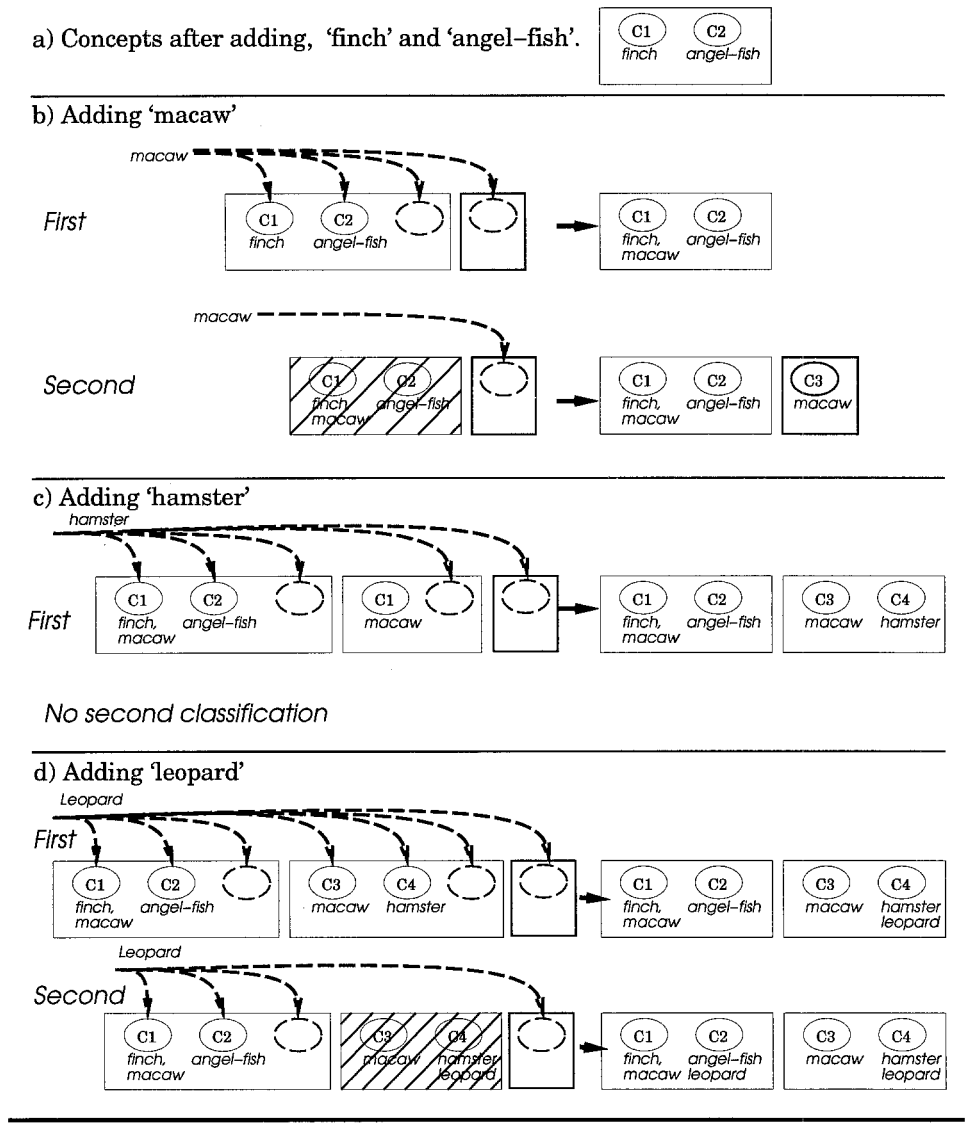

Figure 2. A graphical description of learning in OLOC.

improve predictive utility. In the second case, additional classification of the instance is unnecessary for accurate prediction. In this example, predictive accuracy is rising and some attribute values, reproductive role and appearance, are not predictable using C_1 alone. Therefore, OLOC continues processing the instance. Since there are no existing contrast sets to which the instance has not already been added, OLOC's only option is to create a new contrast set with a new concept, (Utility = 0.308). Note that in the figure, contrast sets with hash marks have been removed from consideration.

Figure 2c shows the process of learning about the 'hamster.' Now there are two existing contrast sets, so OLOC must do more work. OLOC must consider adding the instance to any concept in the first contrast set, creating a new concept in either set, and creating a new set. It does not consider adding the instance to the third concept, C_3, the one by itself in the second contrast set, because the second instance added to a contrast set must become a new concept. This requirement was imposed to ensure there is more than one concept in each contrast set.

In this example, the Utility for adding 'hamster' to the first concept, C_1, is 0.249; the Utility for adding to the second concept, C_2, is 0.243; the Utility for creating a new concept in the first contrast set is 0.245; the Utility for creating a new concept in the second contrast set is 0.303; and the Utility for creating a new contrast set is 0.176. Therefore, the best operation is to create a second concept in the second contrast set.

Again, OLOC tests whether the attribute values in the instance are predictable. In this case, all attribute values are predictable, because the instance is the only known instance of the new concept. All the attribute values of the instance are the most probable values for the new concept. When a new concept is created either in an existing set or in a new set, the instance will be predictable and processing will stop.

Figure 2d shows the process of learning about the 'leopard.' There are still only two contrast sets. OLOC must again consider adding the concept to any concept in either contrast set, creating a new concept in either set, and creating a new set.

In this example, the Utility for adding 'leopard' to the first concept, C_1, is 0.267; the Utility for adding to the second concept, C_2, is 0.282; the Utility for creating a new concept in the first contrast set is 0.278; the Utility for adding 'leopard' to the third concept, C_3, is 0.290; the Utility for adding to the fourth concept, C_4, is 0.297; the Utility for creating a new concept in the second contrast set is 0.297; and the Utility for creating a new contrast set is 0.178. Therefore, the best operation is to add the instance to the fourth concept along with 'hamster.' When there are ties for Utility score, OLOC prefers to add an instance to an existing concept rather than to create a concept. All attribute values are not yet predictable. In particular, the leopard does not live indoors and does not eat packaged-food. Therefore, OLOC continues processing the instance. It must now consider adding the instance to any concept in the first contrast set, creating a new concept in the first set, and creating a new set. Utility scores are calculated based on the combined concepts.

In this example, the Utility for adding 'leopard' to the first concept, C_1, is 0.339; the Utility for adding to the second concept, C_2, is 0.355; the Utility for creating a new concept in the first contrast set is 0.348; and the Utility for creating a new contrast set is 0.210. Therefore, the best operation is to add the instance to the second concept along

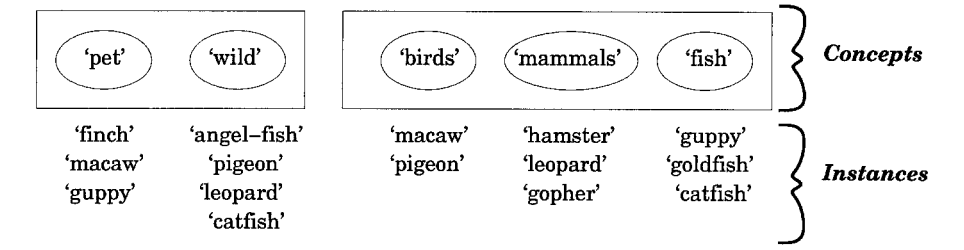

Figure 3. The concept structure after learning all instances from Table 6.

with 'angel-fish.' All attribute values are now predictable and processing of the 'leopard' instance stops.

Figure 3 shows the results of learning all 10 instances from Table 6. The figure includes explanatory labels for the learned concepts (not created by OLOC) and the instances that were classified into each concept (listed beneath each concept). This figure highlights a characteristic of OLOC learning using small sample sizes. Although the internal descriptions of the concepts between contrast sets describe overlapping sets of instances, some of the instances were not classified into more than one concept, i.e., 'finch', 'angel-fish', 'hamster', 'goldfish', and 'gopher.' There are two reasons for this phenomenon. First, early instances are encountered when there is only one contrast set and hence cannot be classified into more than one category. In the example, the second contrast set was learned while processing the third instance. Second, when a new contrast set is learned, the new concepts have only one or a few instances. As a result, these concepts are usually sufficient for predicting all attribute values of an instance. Hence, processing of the instances stops after the instance has been classified into only one category. Generally, after several instances have been classified into the categories in OLOC, subsequent instances are reliably classified into multiple categories.

The concept structure that OLOC learns from a set of instances is dependent on the order in which it encounters those instances. The order of instances in Table 6 was chosen because it provided a clear example of learning. We ran OLOC on 1000 different orders of the instances in Table 6. For most, OLOC learned all the concepts shown in Figure 3. However, with some orders OLOC learned a third contrast set to attempt to predict the values of the attributes: appearance and reproductive-role. Further, with some orders, learning does not happen as quickly and some instances are misclassified, such as putting 'macaw' in the concept that evolves into the 'fish' concept.

3.4. Extensions

Here we describe how OLOC can be extended to delete concepts and learn hierarchical structures. These extensions are noted to illustrate the flexibility and power of our

approach. However, they are not to be used in any example or empirical test in this paper. The extended system will be called OLOC+.

3.4.1. Deleting concepts

OLOC+, like other incremental learners, is hindered by inopportune orders of its input instances, if, for example, the first few instances are very similar. To combat order effects, incremental learners need some learning operator that can reduce the effect of such orders. For example, COBWEB uses the splitting and merging of concepts to combat overly general or overly specific concepts formed early in learning. These operators are applied to a concept in a tree of concepts if they result in improvements to a Utility score. Splitting a concept is an operation that replaces a concept by its children concepts, those one level lower in the tree. In essence, the operator promotes concepts to a higher level in the tree. Merging, on the other hand, combines two concept descriptions into a single one.

In contrast, OLOC+ uses a 'delete-concept' operator that can be applied to remove a concept from a contrast set. It does not use splitting and merging because of the complexities introduced by non-tree hierarchies of overlapping concepts. Every time OLOC adds an instance to a contrast set, it considers deleting the one concept from that set that contributes the least to the Utility score. It only performs the deletion if the Utility score after deletion is higher than that before deletion.

One difficulty with a deletion operator is that information stored in the deleted concept is lost. OLOC+ provides an option to reclassify instances that were previously classified into the deleted category. There is a parameter α that determines the number of instances that will be reclassified. For example, if $\alpha = 2$, then the last two instances classified into each category will be saved and reclassified if the concept is deleted. In many cases, this approach will not result in any lost information and can actually mimic splitting and merging in early learning.[4] In general, α need not be large because the problems that the deletion operator alleviates are most evident in early learning, with small sample sizes.

3.4.2. Hierarchical structures

Overlapping concepts can be arranged in hierarchical structures to represent concepts ordered by specificity. Just as for trees of concepts, more general concepts are linked to successively more specific concepts. In contrast to trees, however, a given specific concept may have several more general parent concepts. For example, a `wooden spoon` concept is more specific than a `wooden object` concept and a `spoon` concept. In such a hierarchy, the more specific concept may be necessary to specify special exceptions about wooden spoons that are not generally true of either wooden objects or spoons, such as their large size. In other words, the more specific concepts allow OLOC+ to override the default combination of the two more general concepts.

OLOC+ learns and uses a hierarchy of levels of overlapping concepts. Here a level is a collection of contrast sets as in Figure 1. Each concept within a level may have a 'child'

Table 7. Comparisons of Concept Formation Systems

Feature:	COBWEB	Anderson AUTOCLASS	Segen	Unimem, Galois	PI	OLOC
Overlapping	X	X	√	√	√	√
Contrasting	N/A	N/A	X	√	√	√
Combination	N/A	N/A	√	X	√	√
Uncertainty	√	√	√	X	X	√

link to a more specific level or collection of contrast sets. For example, the `spoon` concept on one level of the hierarchy may be linked to other contrast sets such as one containing the concepts `wooden-spoon` and `metal-spoon`. After an object is classified as a `spoon`, it can subsequently be classified as a `wooden-spoon`.

Classification and learning occur as in OLOC and occur at several levels of the hierarchy. OLOC+ classifies an instance into categories at one level until adding the instance to a concept at the next more specific level yields a higher Utility score than adding it to concepts at the current level. Similarly, learning updates or creates concepts at the next more specific level when doing so results in a higher Utility score than does any change at the current level.

Although there may be many 'child' links to follow to more specific levels, OLOC+ only follows one: the first such link encountered during classification. As in the example of the wooden spoon, both the concepts `spoon` and `wooden-object` may have more specific levels, but only one of those is used by OLOC+. If the instance is first classified as a `spoon`, OLOC+ will use the child link of that concept and not the child link of `wooden-object`. OLOC+ follows only one child link to more specific levels to avoid the combinatorial explosion possible by following all such links. As a result of this strategy, some more specific concepts that would be relevant to a current instance may be not accessed if they lie along other untraversed child links. An alternative, but more expensive, solution is to follow some larger but constant number of links.

3.5. Related work

Our approach to learning overlapping concepts is similar to existing methods of concept formation but differs along four dimensions. First, OLOC learns overlapping concepts, unlike some concept formation methods. Second, although it learns overlapping concepts, it permits some concepts to be mutually exclusive. Third, it permits the combination of multiple concepts. Finally, OLOC handles uncertainty or noise in a domain using a principled probabilistic model. Table 7 presents these four features and indicates which of several representative systems share those features with OLOC.

Consider each feature in turn. There are several well-known concept formation systems that do not permit overlapping concepts (e.g., Fisher, 1987; Anderson, 1990; Cheeseman et al., 1988). They assume that every instance belongs in exactly one category at each level of the hierarchy. Despite this assumption, two of these systems, Anderson's (1990) system and AUTOCLASS (Cheeseman et al., 1988), do allow instances to be classified into multiple categories, giving the appearance of overlapping concepts. However, both

systems assume that all concepts are mutually exclusive. Instances are multiply classified only because the learner is uncertain where to classify the instance. As a result, the learner partially assigns an instance to each category based on the probability that the instance is a member of that category. A particular instance of a small dog may be an example of the `dog` concept with a probability of 0.9 and an example of the `cat` category with a probability of 0.1.

However, this technique does not produce overlapping concept descriptions in practice. This technique attempts to learn maximally distinct concepts. However, overlapping concepts are not maximally distinct, because they have shared instances. For example, the concepts `dog` and `pet` are not as different as are `dog` and `cat`. Therefore, learning methods that seek disjoint concepts will tend not to learn overlapping ones.

A second point of difference with earlier systems is that OLOC allows some concepts to be contrasting (i.e., "negatively" interdependent), while other systems only permit independent overlapping concepts (e.g., Segen, 1988). In building a probabilistic system for concept formation, it is especially tempting to assume that all concepts are independent of all other concepts. This greatly simplifies the mathematics and allows classification and learning to be tractable. However, it is unusual for a domain to require only independent concepts. Fortunately, it is possible to achieve tractability by assuming that concepts are organized into contrast sets (cf., Ben Bassat, 1980; see Appendix). This implies that at least some of the concepts are not independent (i.e., in this case, mutually exclusive).

Third, unlike other systems (e.g., Lebowitz, 1987; Carpineto & Romano, 1993), OLOC provides a method by which overlapping concepts can be combined to provide a consistent basis for prediction and other concept use. It is not enough that an instance be multiply classified if the resulting concepts do not interact to produce behavior. In GALOIS (Carpineto & Romano, 1993), for example, an instance is multiply classified and then one overlapping concept is chosen as the basis for prediction. Fourth, unlike some systems, OLOC explicitly represents uncertainty by using probabilistic concepts and a corresponding learning method. Such concepts have the advantage of being more sensitive to probabilistic structure in the world and less sensitive to noise and missing values.

Despite OLOC's value with respect to these four features, some of the comparison systems do implement further desirable features that are not implemented in OLOC. For example, unlike OLOC, both Anderson's (1990) method and AUTOCLASS (Cheeseman et al., 1988) are based on more extensive Bayesian analyses. For example, they probabilistically classify instances into concepts in one contrast set. This provides an effective means of addressing some uncertainty that OLOC ignores.

Furthermore, Segen's (1988) minimal representation method allows each concept to store only the attributes that are relevant to the concept. In OLOC, all concepts maintain summary information about all attributes. With overlapping concepts this is often unnecessary. If a learner has a `mammal` concept and a `pet` concept, the pet concept does not need to store any information about whether a pet bears live young. Segen presents one method for eliminating such unnecessary storage.

4. Evaluation of OLOC

A primary purpose of concept formation is to allow correct predictions about novel instances. A model of concept formation must at least show successful learning. Furthermore, to be a model of human concept formation, a learning system must show that it is robust and that in the face of degraded input, its performance degrades gradually.

4.1. An overlapping dataset

To show learning in the presence of noise, we applied OLOC to a base dataset for which overlapping concepts are appropriate. An *overlapping dataset* is one in which there are two or more independent sets of interrelated attributes. The attributes can be partitioned into disjoint subsets such that the values of attributes in any one subset are interpredictive and the values of attributes in different subsets are independent. For example, a dataset may have attributes corresponding to animal type that can be separated from the attributes corresponding to the animal's role. The attributes for animal type are interpredictive, but are independent of the other attributes.

For this experiment, we also varied the amount of noise of the attribute values. Here, the amount of noise refers to the probability that some attribute value will be randomly changed to some incorrect value of that attribute.

Also to show the advantage of OLOC over systems that do not permit overlapping concepts, we applied two other systems to the same datasets. One system was an incomplete version of OLOC that does not permit overlapping concepts. We will refer to this system as SINGLE. The other system was COBWEB (Fisher, 1987). We compare with COBWEB because it is an effective concept formation system that does not permit overlapping concepts but does acquires a hierarchy of concepts instead of a single set of concepts.

4.1.1. Dataset design

Each instance from the base dataset is described using nine attributes, where each attribute has one of four possible values. The first three attributes (1-3) are all perfectly correlated, as are the second set of three attributes (4-6) and the third set of three attributes (7-9). Two attributes are said to be perfectly correlated if the value of one attribute is sufficient to predict the value of the other. All 64 possible instances in this domain are shown in Table 8.

4.1.2. Procedure

In this experiment, each system (OLOC, COBWEB, and SINGLE) was tested at each of four levels of noise. The noise parameter N represents the probability that each attribute value for an instance from the base dataset will be changed to some incorrect value of the

Table 8. Sixty-four instances from Overlapping Animal Dataset, nine four-valued attributes. Values are represented by digits from 0 to 3 and attribute is represented by position.

000000000	111000000	222000000	333000000
000000111	111000111	222000111	333000111
000000222	111000222	222000222	333000222
000000333	111000333	222000333	333000333
000111000	111111000	222111000	333111000
000111111	111111111	222111111	333111111
000111222	111111222	222111222	333111222
000111333	111111333	222111333	333111333
000222000	111222000	222222000	333222000
000222111	111222111	222222111	333222111
000222222	111222222	222222222	333222222
000222333	111222333	222222333	333222333
000333000	111333000	222333000	333333000
000333111	111333111	222333111	333333111
000333222	111333222	222333222	333333222
000333333	111333333	222333333	333333333

same attribute. So if $N = 0.1$, the probability that any attribute value of the instance will be changed randomly to an incorrect value of that attribute is 0.1. Noise is introduced to mimic the effects of nonlogical (probabilistic) relations between attribute values. We varied this noise parameter across the values 0, 0.1, 0.2, and 0.3.

For each system at each level of noise, we performed 10 runs, for a total of 120 learning runs. For each of these 120 runs, a new run dataset was created which was a random order of the instances in the base dataset with attribute values changed in accordance with the noise level. A learning run consisted of presenting one system with the first 54 (training) instances from the run dataset followed by a prediction test for each of the remaining 10 (test) instances from the run dataset. The training and test instances were chosen randomly without replacement from the run dataset.

A prediction test for an instance consisted of, for each attribute, removing that attribute from the instance, classifying the partial instance into a category or categories, and predicting the value of the removed attribute. The score on the prediction test was the proportion of correct predictions across all attributes and across all test instances.

4.1.3. Results

The means are displayed in Table 9 and Figure 4. As expected, OLOC does learn well and its performance degrades gracefully as the input becomes noisier. Furthermore, for this dataset, OLOC does markedly better than either COBWEB or SINGLE. In turn, for low noise, COBWEB does better than SINGLE. SINGLE does particularly poorly because there is no single adequate set of mutually exclusive concepts for this dataset. COBWEB performs better because it can build a hierarchy of concepts to compensate for an inadequate initial partition. OLOC performs the best because it directly identifies overlapping concepts.

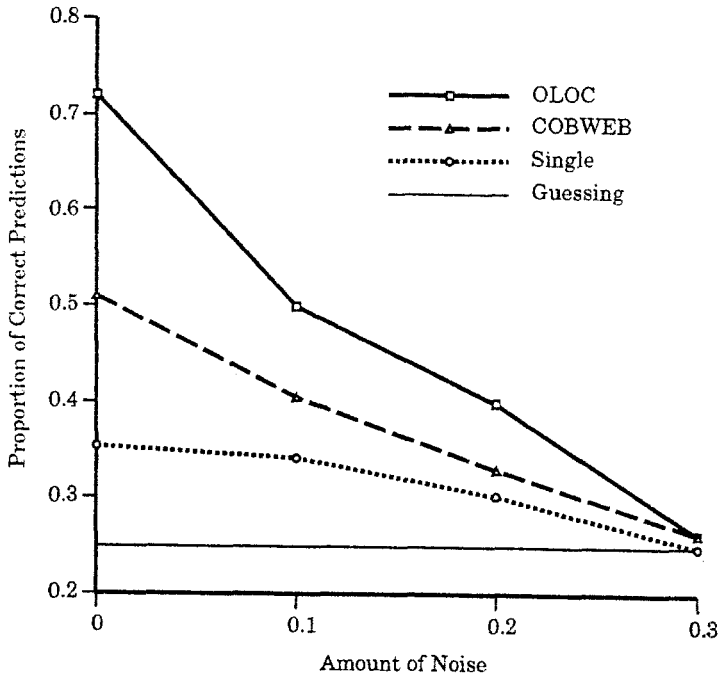

Figure 4. Prediction performance for overlapping dataset for OLOC and two other methods for varying amounts of noise.

Table 9. Proportion of correct predictions by learning system and amount of noise for the Overlapping dataset.

Noise	0.0	0.1	0.2	0.3
OLOC	0.721	0.501	0.400	0.262
COBWEB	0.510	0.406	0.331	0.262
SINGLE	0.353	0.342	0.303	0.250

A two-factor ANOVA applied to the data yielded a significant main effect for system ($F(2, 108) = 39.79$, $p \ll 0.01$), a significant main effect for noise ($F(3, 108) = 61.05$, $p \ll 0.01$), and a significant interaction effect ($F(6, 108) = 9.075$, $p \ll 0.01$). Planned comparisons between OLOC and each of the other two systems for each level of noise (eight planned comparisons) ($S(6, 108, p=0.01) = 4.212$) showed that OLOC was significantly better than both systems for no noise ($F(1, 108)=105.45$, $p < 0.01$; $F(1, 108)=34.739$, $p < 0.01$) and for a noise level of 0.1 ($F(1, 108)=19.683$, $p < 0.01$; $F(1, 108)=7.110$, $p < 0.01$). Furthermore, OLOC was significantly better than SINGLE for a noise level of 0.2 ($F(1, 108)=7.289$, $p < 0.01$). No other planned comparisons were significant.

Table 10. Sixty-four instances from Non-Overlapping Animal Dataset, nine four-valued attributes. Values are represented by digits from 0 to 3 and attribute is represented by position.

111111100	222222200	333333300	444444400
111111101	222222201	333333301	444444401
111111102	222222202	333333302	444444402
111111103	222222203	333333303	444444403
111111110	222222210	333333310	444444410
111111111	222222211	333333311	444444411
111111112	222222212	333333312	444444412
111111113	222222213	333333313	444444413
111111120	222222220	333333320	444444420
111111121	222222221	333333321	444444421
111111122	222222222	333333322	444444422
111111123	222222223	333333323	444444423
111111130	222222230	333333330	444444430
111111131	222222231	333333331	444444431
111111132	222222232	333333332	444444432
111111133	222222233	333333333	444444433

4.2. A non-hierarchical, non-overlapping dataset

The previous result suggests that at the least, OLOC performs the task for which it was designed. However, it does not address whether OLOC performs as well as alternative methods for non-overlapping datasets. To show learning in a non-overlapping dataset, we applied OLOC, COBWEB, and SINGLE to a base dataset and we varied the amount of noise. A non-overlapping dataset is one in which the attributes cannot be partitioned into independent sets of interpredictive attributes.

4.2.1. Dataset design

Again, each instance in the base dataset is described using nine attributes, where each attribute has one of four possible values. In contrast to the previous dataset, the first *seven* attributes differentiate between mammals, birds, reptiles, and insects; the remaining two attributes vary randomly. All 64 possible instances are shown in Table 10. The procedure for this experiment is identical to the previous one except that a different dataset was used.

4.2.2. Results

The mean proportion of correct predictions is displayed in Table 11 and Figure 5. As expected, OLOC does learn well in non-overlapping domains and, like alternative learning systems, its performance degrades gracefully as the input becomes noisier. Furthermore, even for this dataset, both OLOC and SINGLE perform better than COBWEB for high

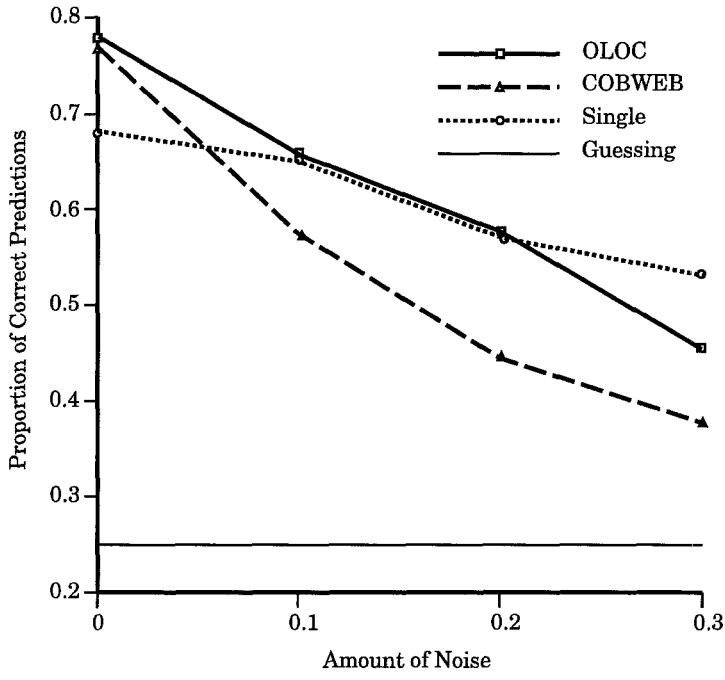

Figure 5. Prediction performance for non-overlapping dataset for OLOC and two other methods with varying amounts of noise.

Table 11. Proportion of correct predictions by learning system and amount of noise for the Non-overlapping dataset.

Noise	0.0	0.1	0.2	0.3
OLOC	0.792	0.663	0.582	0.458
COBWEB	0.778	0.585	0.435	0.382
SINGLE	0.689	0.668	0.584	0.53

noise conditions. COBWEB does particularly poorly because in the high noise conditions it attempts to improve predictions by deepening its hierarchy. As a result, the concept structure overfits the data and performance suffers. Therefore, when OLOC performs better than COBWEB, it may be because of inherent overlapping structure in the domain or the lack of hierarchical structure in a noisy domain.

For a noise level of 0.3, OLOC shows a (non-significant) tendency to learn more poorly than SINGLE. It is possible that OLOC too is overfitting the instances because it learns many contrast sets to attempt to predict the noise.

A two-factor ANOVA applied to the data yielded a significant main effect for system ($F(2, 108) = 9.519$, $p \ll 0.01$), a significant main effect for noise ($F(3, 108) = 61.03$, $p \ll 0.01$), and a significant interaction effect ($F(6, 108) = 4.261$, $p \ll 0.01$). Planned

Table 12. Proportion of correct predictions by learning system and dataset.

	COBWEB	OLOC
ZOO	0.807	0.762
Soybean	0.817	0.732
Congress	0.727	0.778

comparisons between OLOC and each of the other two systems for each level of noise (eight planned comparisons) showed that OLOC did better than SINGLE for a noise level of 0.0 ($F(1, 108) = 6.546$, $p < 0.05$). Further, OLOC did significantly better than COBWEB for a noise level 0.2 ($F(1, 108) = 13.400$, $p < 0.01$). No other planned comparisons were significant. Post hoc Scheffé comparisons ($S(6,108,0.05) = 3.608$) were conducted to determine whether the unexpected differences between COBWEB and SINGLE were significant. SINGLE is significantly better than COBWEB for a noise level of 0.2 ($t(108) = 3.713$) and a noise level 0.3 ($t(108) = 3.6661$).

4.3. Datasets from the UC Irvine repository

In the two previous experiments OLOC performed well for two artificial domains. However, this does not address whether OLOC would also perform well for domains with other patterns of relationship between the attributes. To test its success in other domains, we applied OLOC and COBWEB to three datasets with nominally-valued attributes from the UC Irvine machine learning repository.

The datasets used were the Soybean, ZOO, and Congress datasets. The Soybean dataset is a collection of 307 cases of 19 different soybean diseases. In this dataset, there are 35 attributes besides the disease attribute. The ZOO dataset is a collection of 101 animal types distinguished by 17 carefully chosen attributes. The Congress dataset records how 435 U.S Congressmen voted on 16 key issues in 1984. In this dataset, there are 17 attributes: 16 are votes and the other is the political party affiliation.

All the datasets were treated in the same manner. There were 10 runs for each pairing of a system and a dataset. For each run, an 80-instance training set and a 20-instance test set were randomly chosen without replacement. A learning run consisted of presenting one system with the entire training set followed by a prediction test for each of the test instances.

A prediction test for an instance consisted of, for each attribute in the dataset, removing that attribute from the instance, classifying the partial instance into a category or categories, and predicting the value of the removed attribute. The score on the prediction test was the proportion of correct predictions across all attributes and across all test instances. The prediction scores were averaged over the 10 runs.

Table 12 shows the results for the application of both systems to the three datasets. All differences between systems are significant (ZOO, $t(18) = 4.003$; Soybean $t(18) = 6.937$; Congress, $t(18) = 2.677$; $P<0.05$). For both the ZOO dataset and the Soybean dataset, COBWEB performed significantly better than OLOC. Likely, this advantage is

the result of hierarchical structure in the datasets. The ZOO dataset, in particular, has a clear hierarchical structure.[5]

Conversely, for the Congress dataset, OLOC performs significantly better than COBWEB. As already noted, an advantage for OLOC may be caused either by inherent overlapping structure or by the absence of overfitting.

To summarize, OLOC excels when learning about domains that have inherent overlapping structure; i.e., domains in which there are several good sets of mutually exclusive concepts. Furthermore, OLOC does at least as well as COBWEB for artificial and natural domains unless there is an inherent hierarchical structure. For such domains, OLOC must be extended to permit hierarchical concepts.

5. Modeling conceptual combination

Little research in psychology has investigated human *acquisition* of overlapping concepts, so it is difficult to compare OLOC's learning processes with those of people. In contrast, there is considerable research on the way established concepts are combined and used. The generative combination of concepts pervades cognition and is critical to productivity of thought, from language comprehension to novel problem solving (e.g., Osherson & Smith, 1981; Holland, Holyoak, Nisbett, & Thagard, 1986). OLOC addresses the combination of overlapping concepts. Combinations of overlapping concepts are those that can sensibly be described as "An X that is also a Y" (e.g., a sport that is also a game, such as soccer). The two concepts in a *combined concept* (i.e., `sport and game`) are called *constituent concepts*. In this section we sketch OLOC's promise as a psychological model of the combination and use of established concepts. We describe the phenomena of selective inheritance and typicality, OLOC's account of these, and some limitations of this account.

5.1. The overlapping concept phenomena

Selective Inheritance Phenomena When people are asked to list properties of a combined concept (e.g. a pet that is also a bird), the properties they list are inherited from the constituent concepts (Hampton, 1987). The challenge is to predict when each concept will contribute the inherited value. Hampton suggested two principles that characterize this selection. First, the importance of a property to the combined concept is an average of its importance in the constituent concepts. Hampton called this phenomenon *compensation*. Second, property importance does not seem to average for properties with extremely high or extremely low importance in a constituent concept. In such cases, the extreme value is more highly weighted. In other words, if a property is necessary in either constituent concept, it must be inherited and if it is impossible in either, it cannot be inherited. Hampton labeled this phenomenon *impossibility/necessity*.

Typicality Phenomena When people are asked to judge how typical an instance is of a combined concept (e.g. guppy of the concept `petfish`), they judge that some

instances are much more (or much less) typical of the combined concept than of either constituent concept (Osherson & Smith, 1981; Hampton, 1987). In addition, Hampton noted that one of the two constituent concepts tended to have more weight, or to *dominate*, the other concept in predicting typicality of the combined concept.

Osherson and Smith (1981) and Hampton (1987) outline the inability of instance and prototype models to account for inheritance and typicality phenomena. First, the inheritance phenomena are incompatible with instance models. Instance models represent a concept as its extension, that is, the instances that are its members. Instance models construct combined categories by operations on instances, such as identifying the instances that are members of both constituent categories and designating those as members of the combined category. However, combined categories are not like this. For example, some instances are judged to be members of the combined concept but are not included in either constituent (Hampton, 1987). It appears that the combined concept must be constructed from its intension (attributes), rather than its extension (instances).

Second, typicality phenomena are incompatible with simple prototype theories (i.e., one prototype per concept). Prototype theory predicts that typicality is based on similarity to the prototype. They would thus compute the typicality of an instance to the combined concept as an increasing function of its typicality to the constituent prototypes. But a typicality function that only considers the global typicality of an instance to the constituent concepts does not accurately predict that instance's typicality to the combined concept. In contrast, typicality judgments for a combined concept must be determined attribute by attribute.

5.2. OLOC's *account of the phenomena*

OLOC accounts for selective inheritance with its distribution maximizing rule. In OLOC, the distribution for a particular attribute in a new combined concept is inherited from a single constituent concept. The maximizing rule is used to choose which constituent concept will supply the distribution, and does so independently for each attribute. For example, virtually all fish have scales, while pets vary in body covering. Hence, the concept `pet fish` inherits the distribution for body covering from the concept `fish` not from the concept `pet`. Conversely, having-an-owner is invariably true for pets, but fish vary in ownership-status. Hence, the concept `pet fish` inherits ownership-status from the concept `pet`.

OLOC provides a unified account of both inheritance phenomena that Hampton described: compensation and impossibility/necessity. Hampton proposed an averaging rule to account for compensation. OLOC's maximizing rule is a different form of compensation, but is consistent with Hampton's reported data. Hampton also proposed that the averaging rule does not apply to necessary or impossible properties, but rather that extreme values dominate. In OLOC, properties such as Hampton discussed are represented as binary attributes (two values). If a property is necessary or impossible, its attribute distribution will indicate that one of its two values ('present' or 'absent') will

have the maximum possible probability (1.0). Hence, the maximizing rule will select this distribution for the combined concept (barring ties).

Hampton makes one further observation about inheritance. He found that people list a very small number of properties (e.g., lives in a cage) that applies only to the combined concept (`pet bird`), not to its constituent concepts. His informal content analysis suggested that direct experience with combined concepts (i.e., observing pet birds) was the source of such properties. Since Hampton did not examine learning, his data cannot directly confirm this. However, that is exactly what a hierarchical OLOC would do. As new instances are classified into the combined concept, its attribute distributions are updated as for any other concept.

Besides inheritance effects, OLOC also models the typicality effects. OLOC measures typicality of an instance to a concept as the probability of the instance given the concept, $P(I|C)$. If a concept predicts many of the values that an instance actually has (and does so with high certainty) the instance is highly typical of that concept. For OLOC, a guppy is more typical of `pet fish` than of either constituent, because many features of the guppy are probable for `pet fish` but are relatively improbable for fish in general or pets in general. Indeed, if there were not instances that could be better predicted from the overlap concept, OLOC would not construct the concept or classify a particular instance into it. OLOC, and we believe people, use combined concepts precisely when neither constituent concept does an adequate job of characterizing an instance.

Finally, Hampton suggested that concept dominance might derive from the way attributes are selectively inherited, specifically that the dominant concept would have a larger number of more important properties. OLOC explains dominance as a byproduct of constituent concepts that differ in their numbers of highly probable attribute values. The constituent concept with the most highly probable attribute values dominates in the combined concept.

In sum, OLOC provides qualitative accounts for existing data about the combination of overlapping concepts. However, there are many instances of conceptual combination that are not cases of overlapping concepts and that OLOC does not address. Dog food, oil money, and beach houses are not combinations of two categories. An instance of dog food is not an example of a dog. In these cases, the modifier noun (e.g., 'dog') functions as the value of an attribute possessed by the head noun (e.g., "food that is eaten by a dog"). Interpretation of such combinations depends on several factors. The patterns of language specify that examples of dog food must be food, but need not be dogs. Familiar expressions (bus fare) may specify interpretation of novel analogs (bicycle fare). Or interpretation may be unspecified and require extended bouts of reasoning from context (table shoe) or learning about special cases. Interestingly, information available to OLOC would allow it to identify many cases where a combined concept should not be treated as overlapping. Highly certain but conflicting values of the same attribute would provide one valuable clue that two concepts do not overlap.

6. Discussion

OLOC is an effective concept formation system that was inspired by human cognitive abilities. Here we discuss the source of OLOC's success, its value as a cognitive model, and future directions.

6.1. OLOC *as a machine learning system*

OLOC incrementally learns one layer of overlapping concepts. The concepts are organized into multiple contrast sets—sets of mutually exclusive concepts. When learning or performing predictions, OLOC seeks overlapping concepts, which are then combined to form a composite concept.

Because of its successful performance for many datasets, OLOC is a contribution to machine learning models of concept formation. It is the only system to provide a probabilistic means for learning, while using both contrasting and overlapping concepts.

Our results show that when the domain of instances has multiple good sets of mutually exclusive categories, OLOC learns more effectively than do alternative systems. Furthermore, for other domains, without overlapping categories, OLOC performs comparably to alternative systems.

OLOC leads to superior predictive performance primarily because it can directly represent multiple sets of mutually exclusive concepts; i.e., overlapping concepts. A system that permits hierarchical organizations of non-overlapping concepts can theoretically achieve the same performance as OLOC by progressively extending the depth of its concept tree. For example, where OLOC learns the concepts `pet`, `wild`, `dog`, and `cat`, COBWEB would eventually learn the concepts `pet-cat`, `wild-cat`, `pet-dog`, and `wild-dog` at some level of its hierarchy. Although COBWEB can learn every possible combination of the overlapping concepts, this process is much slower than is OLOC's approach, and it does not organize information about overlapping concepts in easily generalized ways.

There are two additional reasons why OLOC learns successfully. First, as OLOC learns overlapping concepts, it is not only performing a hill-climbing search, it can also be seen as performing a limited form of a beam search for a single best contrast set. For concept formation, a beam search considers several alternative concept structures simultaneously. OLOC can simultaneously entertain many alternative contrast sets, and it therefore has a greater chance than systems like COBWEB, of finding the single best contrast set.

Second, OLOC incorporates two methods that help avoid overfitting the input data. When learning from noisy, erroneous input, the learner must be careful to base predictions on consistent regularities, not on random variation. If the learner relies on random variation present in training instances, it is overfitting the data and is more likely to produce incorrect predictions.

Because small samples are often unrepresentative, OLOC does not trust such samples when estimating conditional probabilities. Hence, for small samples, OLOC is more conservative than is COBWEB when deciding what variation should be considered consistent and what should be considered random.

OLOC also avoids overfitting the data because it only continues classifying an instance until all features are predictable or the number of predictable features does not increase. If OLOC were to continue classifying past this point, as can happen in COBWEB, it would attempt to learn consistency in what appear to be noisy attributes.

6.2. OLOC *as a psychological model*

OLOC was designed as a concept formation system, not to address specific psychological findings; it was designed primarily for computational efficiency, not necessarily as a model of what people do. Nevertheless, it provides a good account of one form of the phenomena of human conceptual combination.

We consider the main successes of the model in qualitatively matching human performance to be the following. 1) OLOC correctly predicts that an instance can be included in a combined concept even if it is not included in both constituents. These and similar cases are not predicted by many models of concept learning or representation, particularly those that depend on instances or 'average' prototypes. 2) OLOC matches key characteristics of how humans allow a combined concept to inherit properties selectively, some from one and some from the other constituent concept. OLOC uses a single mechanism, the distribution maximizing rule, to model the effects of compensation, impossibility/necessity, and dominance.

6.3. *Future work*

In future work, we will extend OLOC's capabilities both as a machine learning algorithm and as a model of human concept learning. Although successful "as is," OLOC's capabilities can be extended. First, OLOC can be applied recursively to build a hierarchy of overlapping concepts. We will design several alternative approaches to hierarchical learning, including that outlined above, and determine the conditions under which each approach is warranted. Second, OLOC does not explicitly acknowledge the possibility of positively inter-dependent overlapping concepts. Such concepts are common: sports and games overlap, but most sports are also games. Therefore, OLOC might be improved if an efficient Utility score can be found that does not assume independence among overlapping concepts.

OLOC can also be improved as a model of concept formation and use by people. First, we will assess the extent to which the current model is accurate. If people use the methods underlying OLOC, then OLOC can be used to generate several predictions about human behavior. With respect to typicality judgments, OLOC predicts that the best instances (Gretchen the Guppie) of a combined concept (`Pet Fish`) will be more typical of that concept than will the best instances (Tim the Trout) of its constituent concepts (`Fish`). The combined concept is only formed if neither constituent concept is adequate for characterizing an instance, such as Gretchen the Guppie.

Moreover, OLOC predicts that, if people learn overlapping concepts, they would generalize differently than if they do not. Suppose that a person encounters several examples

of objects that can be classified into two contrast sets, say `pets`, `wild`, `performer`, and `canine`, `feline`, `equine`. Furthermore, suppose that the person sees every combination of the concepts in the two contrast sets, except performing-felines. When finally presented with a partial instance of a performing feline, the person will use the learned concepts to make predictions. If, like COBWEB, the people learn a tree of concepts, they will be wrong consistently in these predictions, because all known examples of performing animals are different from felines and vice versa. However, if, like OLOC, the person learns overlapping concepts, the performing feline will be just a novel combination of well-learned concepts. As a result, the predictions will be correct consistently for this artificial domain.

OLOC's account of conceptual combination can also be improved by extending it to other types of combination. There are circumstances in which a weak but universal composition method is incomplete. First, people do not always need to combine concepts explicitly. Their understanding of the concept pet-fish can be based on direct experience with 'exceptions'; i.e., a particular pet fish. Conceptual combination is critical for dealing with novel cases, but some related phenomena may be based on the direct learning of combined concepts (cf., Hampton, 1987, on "extensional feedback"). If made to be hierarchical, OLOC would learn about combined concepts directly from exceptions.

Second, people have extended knowledge (or theories) beyond particular concepts, and this knowledge is important for conceptual combination as well (see Murphy & Medin, 1985). People can reason about the size of a pet alligator to conclude it is probably smaller than the usual alligator, even if size were more important for alligator than pet. This reasoning is particularly important for cases in which some feature, such as 'alive,' is both necessary and impossible in the combined concept, such as stone lions. In general, we expect that a more local, automatic procedure, such as that used by OLOC, is used and preferred over open, extended reasoning.

Finally, OLOC does not use information specified in the syntactic form of language to guide combination, and this is clearly important. The head noun gets more importance than do modifiers; a dog house is a house and a house dog is a dog. This is one major cause for asymmetry or non-commutativity effects. OLOC could use such information to guide combination, but does not currently include any sensitivity to 'external' factors such as language or theory.

Acknowledgements

We would like to thank Michael Pazzani, Nancy Smith Martin, Anandeep Pannu, Foster Provost, Jonathan Rubin, and three anonymous reviewers for help with earlier versions of this paper. This research was partially supported by an Equifax fellowship to the first author.

Appendix A

In this appendix, we derive the Utility score introduced in section 3. The derivation is based on six assumptions that concern how concepts, contrast sets, subsets of concepts, and predictions are interrelated. These assumptions are,

1. An instance is classified into the elements of exactly one subset of concepts.
2. The concepts in one contrast set are independent of those in other sets.
3. A subset of concepts, Z_s, consists of M concepts, one from each of M contrast sets.
4. Each prediction event requests the value of one attribute.
5. The values of each attribute are disjoint.
6. The probability of guessing value, V_{ij}, when an instance is classified into a category, C_k, is the probability that value is true given that classification, $P(A_j = V_{ij} \mid C_k)$.

To derive the Utility score, we begin with a simple observation and gradually transform this observation into the Utility score. For each subset of concepts, there is a probability that an instance was classified into all the categories in the subset after we know that the learner's prediction was correct and we know that the learner has the concept structure θ. Because each instance is assigned to exactly one subset of categories (Assumption 1), the sum of these probabilities is 1, as in Equation A.1.

$$1 = \sum_s P(Z_s|correct, \theta) \tag{A.1}$$

We assume that each concept in one contrast set is probabilistically independent of all concepts in all other contrast sets (assumption 2), i.e., $P(Z_s|correct, \theta) = \prod_{\{k|C_k \in Z_s\}} P(C_k|correct, \theta)$. As a result of this assumption, Equation A.1 becomes Equation A.2.

$$1 = \sum_s \prod_{\{k|C_k \in Z_s\}} P(C_k|correct, \theta) \tag{A.2}$$

The sum in Equation A.2 has an exponential number of terms. However, it can be simplified to have only one term per concept. To do so, we use the distributive principle of multiplication over addition and assume that an instance is classified into one category from each contrast set (Assumption 3).

As a result of this assumption, Equation A.2 becomes Equation A.3.

$$1 = \prod_l^M \sum_{\{k|C_k \in S_l\}} P(C_k|correct, \theta) \tag{A.3}$$

In Equation A.3, M is the number of contrast sets. We next apply Bayes' theorem to produce Equation A.4.

$$1 = \prod_{l}^{M} \sum_{\{k|C_k \in S_l\}} \frac{P(correct|C_k,\theta)P(C_k|\theta)}{P(correct|\theta)} \tag{A.4}$$

Because the term, $P(correct|\theta)$, does not depend on the subscripts, k or l, Equation A.4 can be rewritten as,

$$P(correct|\theta) = \sqrt[M]{\prod_{l} \sum_{\{k|C_k \in S_l\}} P(correct|C_k,\theta)P(C_k|\theta)} \tag{A.5}$$

We next specify the probability of being correct given a concept. We assume that on any prediction trial, the learner will be asked to predict the value of exactly one attribute (Assumption 4). Therefore, we can write,

$$P(correct|C_k,\theta) = \sum_{i} P(A_i|\theta)P(correct|A_i,C_k,\theta) \tag{A.6}$$

We assume that the values of one attribute are mutually exclusive (Assumption 5) and that the learner makes predictions about an attribute value based on the probability of that attribute value given that it was classified into a category (Assumption 6). Equation A.6 becomes,

$$P(correct|C_k,\theta) = \sum_{i}\sum_{j} P(A_i|\theta)P(A_i = V_{ij}|C_k,\theta)^2 \tag{A.7}$$

Finally, by substituting this result into Equation A.5, we obtain the Utility measure used by OLOC:

$$U(\theta) = P(correct|\theta) = \sqrt[M]{\prod_{l} SetUtility(S_l)} \tag{A.8}$$

$$SetUtility(S_l) = \sum_{\{k|C_k \in S_l\}} P(C_k|\theta) \sum_{i}\sum_{j} P(A_i|\theta)P(A_i = V_{ij}|C_k,\theta)^2 \tag{A.9}$$

Notes

1. When the concepts are independent, only one concept can have a $P(A_i = V_{ij}|C_k)$ that differs from $P(A_i = V_{ij})$. Using the Utility score, the values for $P(A_i = V_{ij}|C_k)$ will be larger than $P(A_i = V_{ij})$. Therefore, that concept's distribution for that attribute will also have the largest maximum probability.
2. Empirically, Equation 3 also leads to lower predictive accuracy. For the domain described in Section 4.1, when using Equation 3, there is significantly poorer prediction performance ($t(11) = 4.156$, $p < 0.01$).
3. There are many other possible operators such as the deletion of concepts or contrast sets (Martin, 1992) or merging and splitting concepts (Fisher, 1987). OLOC is able to find highly accurate concept structures without these operators, so they were not used.
4. Splitting can be achieved because, as the instances are reclassified, two or more new concepts may be created.
5. When OLOC is extended to permit hierarchies, performance on both the ZOO dataset and the Soybean dataset is indistinguishable from COBWEB's.

References

Anderson, J. R. (1990). *The Adaptive Character of Thought.* Hillsdale, NJ: Lawrence Erlbaum.

Anderson, J. R. & Matessa, M. (1992). Exploration of an incremental, Bayesian algorithm for categorization. *Machine Learning, 9,* 275–308.

Barsalou, L. W. (1983). Ad hoc categories. *Memory and Cognition, 10,* 82–93.

Barsalou, L. W. & Sewell, D. R. (1984). Constructing representations of categories from different points of view. Emory Cognition Project Report #2, Emory University.

Ben Bassat, M. (1980). Multimembership and multiperspective classification: Introduction, application, and Bayesian model. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-10,* 331-336.

Carpineto, C. & Romano, G. (1993). GALOIS: An order-theoretic approach to conceptual clustering. In *Proceedings of the Tenth International Conference on Machine Learning,* (pp. 33–40). Amherst, MA: 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.

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

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: Academic Press.

Hampton, J. A. (1987). Inheritance of attributes in natural concept conjunctions. *Memory and Cognition, 15,* 55–71.

Hampton, J. A. (1988). Overextension of conjunctive concepts: Evidence for a unitary model of category typicality and class inclusion. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 14,* 12-32.

Holland, J.H., Holyoak, K.J., Nisbett, R.E., & Thagard, P.R. (1986). *Induction: Processes of inference, learning, and discovery.* Cambridge, MA.: MIT Press.

Jones, G. V. (1982). Stacks not fuzzy sets: An ordinal basis for a prototype theory of concepts. *Cognition, 12,* 281–290.

Lebowitz, M. (1987). Experiments with incremental concept formation: UNIMEM. *Machine Learning, 2,* 103-138.

Martin, J. D. (1992) Direct and indirect transfer: Investigations in concept formation. (Technical Report GITAI-92-58, Atlanta, GA: Department of Computer Science, Georgia Institute of Technology).

Murphy, G. L. (1990). Noun phrase interpretation and conceptual combination. *Journal of Memory and Language, 29,* 259-288.

Murphy, G. L. & Medin, D. L. (1985). The role of theories in conceptual coherence. *Psychological Review, 92,* 289–316.

Osherson, D. N. & Smith, E. E. (1981). On the adequacy of prototype theory as a theory of concepts. *Cognition, 9,* 35–58.

Osherson, D. N. & Smith, E. E. (1982). Gradedness and conceptual combination. *Cognition, 12,* 299–318.

Rosch, E. H. (1978). Principles of categorization. In E. Rosch & B. Lloyd (Eds.), *Cognition and Categorization,* (pp. 27–48). Hillsdale, NJ: Erlbaum.

Roth, E. M. & Shoben, E. J. (1983). The effect of context on the structure of categories. *Cognitive Psychology, 15,* 346–378.

Segen, J. (1988). Conceptual clumping of binary vectors with Occam's razor. In *Proceedings of the Fifth International Conference on Machine Learning,* (pp. 47–53). Ann Arbor, MI: Morgan Kaufmann.

Smith, E. E. & Osherson, D. N. (1984). Conceptual combination with prototype concepts. *Cognitive Science, 8,* 337–361.

Smith, E. E. & Medin, D. L. (1981). *Categories and Concepts.* Cambridge, MA: Harvard University Press.

Received November 16, 1992

Accepted January 25, 1994