Simulation Models
of the Influence of Learning Mode
and Training Variance on Category Learning

RENÉE ELIO
KUI LIN

University of Alberta

This article uses simulation as an empirical method for identifying process models of strategy effects in a category-learning task. A general set of learning assumptions defined a symbolic learning framework in which alternative simulation models were defined and tested. The goal was to identify process models that could account for previously reported data on the interaction between how a learner encounters category variance across a series of training samples and whether the task instructions suggested an active, hypothesis-testing approach, or a more passive learning mode. Descriptive characterizations of active and passive learning were mapped into complementary settings of parameters operating with the general learning framework. Alternative models, defined by different configurations of these parameters, were evaluated on their goodness of fit to the observed data. The signature differences between models that best fit the passive learning data and models that best fit the active learning data concerned a delayed versus immediate learning parameter and a degree-of-match parameter that determined which patterns were retrieved to make category decisions. A functional account of these parameters is given by considering the learning task as a search process and the role of these parameters in localizing the impact of learning mechanisms in certain areas of the search space. Issues related to simulation as an empirical method for identifying candidate process models are discussed.

There has been a continuing interest over the past two decades in identifying performance characteristics and processing mechanisms associated with different learning strategies. Much of this interest was generated by the distinction between implicit and explicit learning made by Reber and his colleagues...
(Reber, 1976; Reber & Allen, 1978) and by Brooks's (1978) seminal article on nonanalytic versus analytic concept formation. Since that time, a large body of empirical evidence has shown that learning performance on induction tasks can be affected by manipulating instructions (to suggest alternative strategies) or characteristics of the task itself (e.g., Kemler-Nelson, 1984; Medin & Smith, 1981; Nosofsky, Clark, & Shin, 1989). An open and interesting research question is how different strategies map into different computational processes and interact with learning-task features to impact performance. Such issues are relevant to current learning theories that distinguish among types of learning mechanisms (e.g., automatic vs. deliberate) and to efforts at combining symbolic frameworks that support a goal-directed approach to learning with the more implicit learning that characterizes connectionist models (Bechtel, 1988).

In this article, we describe our use of simulation as an empirical method for identifying process models to account for previously reported data on a learning-from-examples induction task. Specifically, we focus on one fairly complex set of findings in a category-learning study by Elio and Anderson (1984), who reported an interaction between a task feature—how category variance was introduced to learners across a series of training samples—and an instructional manipulation that encouraged learners to take either a more active or a more passive approach to the task. They found that transfer performance was better if learners initially studied a fairly homogeneous set of category members and gradually encountered more variance on later training samples, than if each training sample reflected the entire range of category variance. However, the reverse was true when the instructions encouraged a more active, hypothesis-evaluation approach. In this case, a slow introduction of category variance led to lower accuracy and typicality scores than did consistently representative samples that reflected the allowable category variance.

These were interesting results in their own right, because the interaction did not directly fall out of any computational model. They are particularly challenging from a simulation viewpoint because they require a consideration of the kinds of changes to computational processes that might correspond to descriptive notions of what constitutes a learning mode or strategy. The tasks and results are also important from the perspective of understanding how to model information-order effects in learning. Learning systems and algorithms are either designed to be order insensitive [e.g., Mitchell's (1982) version-space algorithms] or are known to be order sensitive [e.g., Lebowitz's (1986) generalization-based memory paradigms]. The particular set of findings that we examine here suggests something of a "first impression" bias that affects how subsequent information is processed, at least in the short term. Hence, it is interesting to explore what types of mechanisms give rise to this biasing effect.
We did not have process models for active or passive learning strategies specified a priori whose validity we wished to demonstrate or verify. Instead, we used simulation as an empirical method for identifying possible computational realizations of learning-strategy effects. Essentially, we did a constrained search for alternative models in a space defined by a set of fairly general theoretical constructs. Our aim was to identify models that fit the active learning data and models that fit the passive learning data and then to consider which parameter differences distinguished the two types of models. Our method was to conjecture how an active, hypothesis-testing approach might map into certain processing constraints and then to investigate whether changes to those processing constraints yielded the qualitatively different learning curves found with passive learning instructions on this task.

The outcome of this effort is twofold. First, we identified a configuration of parameters that, with complimentary settings, modeled either the passive learning or active learning data and hence, constitute candidate process accounts for passive and active learning on this particular induction task. Second, we better appreciated issues associated with using simulation as a generator of models and with how the general principles underlying a "successful" simulation model can be better understood. The implications of both these outcomes form the thrust of this article.

The remainder of the article is organized as follows: We first present the experimental task and the simulation goals that comprise the focus of this effort. Next, we describe the general architecture in which we explored possible models. Following that, we outline the processing parameters we investigated and present results of several alternative models, evaluating their fit to the observed data. We conclude with observations on our computational accounts of learning-mode effects and on simulation as a methodology for identifying such accounts.

**EXPERIMENTAL TASK, RESULTS, AND SIMULATION GOALS**

Category learning studies are aimed at understanding (a) a learner's representation of category information as a result of studying some training set of category exemplars, and (b) how that representation is used to make subsequent membership judgments about possible members. Here we are concerned only with artificially designed categories, not naturally occurring kinds. One important learning-task feature that has been investigated in several studies is the size and representativeness of the training set, as well as structural features of the categories themselves (Homa, 1984). If the to-be-learned category has a large number of members and there is a high degree
of variability in the features that define those members, then transfer performance is better when the training set reflects the allowable variation (e.g., Homa & Vosburgh, 1976; Posner & Keele, 1968). Such a finding can be accounted for by a variety of models. Our focus here is on the Elio and Anderson (1984) study, which was not concerned with contrasting the impact of representative versus nonrepresentative training material per se. Rather, that study's design held constant the amount of variance in the training material and varied only when a learner encountered different types of category members that illustrated the allowable range of within-category variation. In this section, we review the motivation for that study's manipulations and design, which is relevant to understanding the work reported here.

**Experimental Task**

**Stimulus Design and Rationale**

The Elio and Anderson (1984) experiments used two large artificial categories designed to have a high amount of variability among members within each category. The category exemplars were defined as five 4-valued dimensions. A given exemplar can be represented as a 5-digit number (e.g., 22134), where the position of each digit represents a particular dimension and the value of the digit represents a particular value on that dimension.

The goal in designing these categories was to create a high degree of variability while ensuring that some cohesive structure bound together the members within a category. This was done first by defining member types with a distinctive feature combination and then permuting certain features within the feature combinations to create the category exemplars. There was a structured overlap between these distinctive feature patterns that defined the member types so that the category members formed from these feature patterns also shared overlapping features. The general notion was to create a structured category similar, in some ways, to a category like "bird," in which there are many "robin-type" members, fewer "parrot-type" and "quail-type" members (who nonetheless share some feature values with the robin-type), and still fewer "ostrich-type" and "penguin-type" members (outlying members that have their own idiosyncratic feature patterns as well as some commonalities with all category members).

The top of Table 1 presents the defining patterns for the member types for one of the artificial categories from that study and used here as well. As Table 1 shows, the member types differed in which dimensions were fixed with either Value 1 or Value 2. The distinctive pattern for Type A members was arbitrarily defined to be a single feature fixed at Value 2; the distinctive pattern for Type B and B' members was that two features were fixed at Value 2; and the distinctive pattern for Type C and C' types was three
### TABLE 1
Category 1 Items for Empirical and Simulation Tasks

<table>
<thead>
<tr>
<th>Member Type</th>
<th>C</th>
<th>B</th>
<th>A</th>
<th>B'</th>
<th>C'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Member-type Patterns</td>
<td>22211</td>
<td>22111</td>
<td>21111</td>
<td>11122</td>
<td>11222</td>
</tr>
<tr>
<td></td>
<td>12211</td>
<td>12111</td>
<td>11211</td>
<td>1121</td>
<td>11112</td>
</tr>
<tr>
<td>Category Members</td>
<td>22231</td>
<td>22113</td>
<td>21113</td>
<td>31211</td>
<td>31221</td>
</tr>
<tr>
<td></td>
<td>22241</td>
<td>22114</td>
<td>21114</td>
<td>11214</td>
<td>41221</td>
</tr>
<tr>
<td></td>
<td>22213</td>
<td>22311</td>
<td>21311</td>
<td>11321</td>
<td>13221</td>
</tr>
<tr>
<td></td>
<td>22214</td>
<td>22411</td>
<td>21411</td>
<td>11412</td>
<td>14221</td>
</tr>
<tr>
<td></td>
<td>12231</td>
<td>12311</td>
<td>13211</td>
<td>13212</td>
<td>31122</td>
</tr>
<tr>
<td></td>
<td>12241</td>
<td>12411</td>
<td>14112</td>
<td>14122</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12213</td>
<td>13211</td>
<td>13112</td>
<td>31122</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12214</td>
<td>11241</td>
<td>14112</td>
<td>41122</td>
<td></td>
</tr>
</tbody>
</table>


features fixed at Value 2. Although these decisions did not exhaust the possible combinations of constrained feature values, they served to define member types with overlapping feature patterns.

The member type patterns were permuted to generate the actual category members while retaining the distinctive arrangements of features constrained to have Value 2. This was done by randomly selecting a Value 1 feature in the pattern and changing it to be Value 3 or 4. The lower portion of Table 1 gives the category members defined in this way. The frequency of members within each member type was manipulated deliberately, so that certain distinctive patterns occurred more often than others. Hence, there were a large number of Type A members, fewer Type B and B' members, and still fewer Type C members.

An alternative category was created as the mirror image of the one shown in Table 1. This was done by interchanging Values 1 and 4, and Values 2 and 3. Thus, for Category 1 member 11231, there was a corresponding Category 2 member 44324.1

1 Subjects did not study this numeric notation, but rather items constructed by replacing randomly the abstract features and values with semantic descriptors of people.
Manipulating Exposure to Variance During Learning
The category members were divided into four equal samples; each sample had 10 Category 1 members and the corresponding 10 Category 2 members. These four samples were presented independently over the course of four training-testing blocks. During a block's training phase, a category member description was shown on a CRT screen; the subject guessed the category to which it belonged and received feedback on the decision. The description and its correct category remained on the screen for an additional 10 s for the subject to study. After three passes through a training sample, a subject then entered a testing phase, during which he or she made speeded accuracy and typicality judgments on all the category members. Following that, the next block began, with its associated training sample. By the end of the fourth block, all subjects had studied all category members paired with their associated category in a training phase.

Subjects' exposure to the category variance was manipulated by the manner in which members were selected from the category distribution to form each sample. The Elio and Anderson (1984) study investigated several kinds of sampling conditions, but the results of interest here hinge on two types: the gradual variance condition and the representative variance condition. Table 2 gives the selection-without-replacement rules used to create the four training samples under these two conditions. In the representative variance condition, the frequency of each member type mirrors the frequency with which these member types appear in the category distribution. In the gradual variance condition, the first sample is a low-variance set comprised mostly of Type A members. The second and third samples gradually incorporate category members of the other types. The fourth sample is a representative sample.

Manipulating Learning Mode
Two kinds of instructions were used. The default instructions discouraged any active hypothesis testing or attempt to identify rules. Subjects were told...
that the categories were extremely complex, that there was no single, simple rule for determining category membership, and hence the easiest and most successful way to "get through" the training phases was to concentrate on memorizing each exemplar and its associated category label. We will call this the "passive learning" condition. The other instructions indicated that rules governed category membership and subjects were asked to generate their current hypotheses about membership rules or to indicate how predictive each feature was of each category. We will call this the "active learning" condition.

Summary of Empirical Results
Experiment 1 of the Elio and Anderson (1984) study used only passive learning instructions. Subjects receiving gradual variance training samples had significantly higher typicality ratings ($p < .05$) on their last (Block 4) transfer test than did subjects given representative variance training. Representative variance subjects were 80% accurate in their category judgments on this last test, whereas gradual variance subjects were 91% accurate, a difference that approached significance. For both typicality and accuracy, gradual variance subjects held a consistent advantage over representative subjects across all transfer tests. In Experiment 3, the active versus passive instructional manipulation was crossed with gradual and representative variance training. There was a significant interaction between variance order and learning mode on the final test's typicality ratings ($p = .035$) and accuracy scores ($p = .01$). Accuracy and typicality means were higher in the active learning condition given representative variance training and higher in the passive learning condition given gradual variance training ($ps < .04$ and .02 for accuracy and typicality respectively). Across all four transfer tests, representative variance subjects under the active learning instructions maintained a constant advantage over representative variance subjects following a passive learning strategy. This advantage was significant at the .04 level and was not qualified by any higher order interactions. Table 3 gives the block-by-block accuracy and typicality scores for Experiment 3 as a function of training condition and instructional condition.

Simulation Goals and Methodology
For this simulation study, we concentrated on Experiment 3 results. Although there were some interesting effects concerning member types, our simulation goals were aimed only at accounting for the general interaction between variance order and learning mode: an advantage on transfer tests for representative variance training under an active learning mode, and an advantage for gradual variance training under a passive learning mode. Specifically, we wanted to identify two types of process models. The first type should produce higher accuracy and typicality scores across the four transfer tests under gradual variance training than under representative
TABLE 3
Observed Accuracy and Typicality Scores
(from Elia & Anderson, 1984, Experiment 3)

<table>
<thead>
<tr>
<th>Variance Order</th>
<th>Passive Learning</th>
<th>Active Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Representative</td>
<td>Gradual</td>
</tr>
<tr>
<td>Accuracy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer Test</td>
<td>.75</td>
<td>.74</td>
</tr>
<tr>
<td></td>
<td>.78</td>
<td>.81</td>
</tr>
<tr>
<td></td>
<td>.78</td>
<td>.82</td>
</tr>
<tr>
<td></td>
<td>.80</td>
<td>.83</td>
</tr>
<tr>
<td>Typicality</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transfer Test</td>
<td>1.72</td>
<td>1.82</td>
</tr>
<tr>
<td></td>
<td>1.99</td>
<td>2.26</td>
</tr>
<tr>
<td></td>
<td>2.03</td>
<td>2.49</td>
</tr>
<tr>
<td></td>
<td>2.14</td>
<td>2.68</td>
</tr>
</tbody>
</table>

variance training. The models that accomplished this would be candidate accounts of passive learning. The second model type should produce higher accuracy and typicality scores under representative variance training than under gradual variance training. This model set would constitute candidate active learning models. The differences in the parameters that define these two classes of models would elucidate some general principles that contribute to a computational account of learning strategies.

The space of possible process models was defined by a general learning framework. We did not know what kinds of models within this framework would simulate either the passive or active learning performance curves or even the variance order effects under either of the learning modes. Rather, this simulation effort was aimed at identifying the critical parameters within the framework that, with one set of values, produced passive learning results and, with another set of values produced active learning results. Our exploration of the model space was guided by our conjectures about how active learning, viewed as a hypothesis-testing approach, might map into particular constraints on particular mechanisms. The next section presents the general framework that defined the space of models we investigated.

THE LANA SIMULATION FRAMEWORK

Cognitive simulation can be viewed as a search through a space of possible models. The boundaries of that space, and hence the character of the models that populate it, are defined by some set of theoretical assumptions. Briefly stated, the theoretical assumptions defining the model space that we considered were the following. During the learning, there is limited memory for
1. Accept current instance for classification and put it in working memory.
2. Retrieve feature-pattern—category patterns and add these to working memory.
3. For each pattern in working memory, compute a score based on its strength and similarity to current instance.
4. Choose the highest scoring pattern. Assign current instance to that pattern’s category.\(^a\)
5. If the classification decision was correct then
   5.1 If a generalized pattern was used, strengthen it.
   Else if instance pattern was used, mark it “used for a correct decision.”
   5.2 Form a maximally specific generalization between the current instance and the pattern used to classify it.
Else if the decision was incorrect and a generalized pattern was responsible, then
   5.3 Reduce the generalized pattern’s strength.
   5.4 Form a new, more discriminating pattern.
6. Update long-term memory.
   6.1 If a pattern created in Steps 5.2 or 5.4 already exists, then strengthen that existing pattern.
   Else add the new pattern to long-term memory.
   6.2 Add recently used instance patterns.\(^b\)
7. If working-memory size has been reached, then drop the oldest instance pattern and add the current trial’s instance pattern and its correct category tag.

\(^a\) If no pattern meets the minimum strength or similarity requirements, a random category assignment is made.
\(^b\) This step is a model-dependent parameter (see text).

recently studied instances and their associated category. An abstraction process detects similarities among instance patterns involved in a decision-making process and forms generalized feature patterns that reflect these similarities. When an instance is presented for classification, potentially relevant patterns are retrieved from long-term memory on the basis of their similarity to the presented item and their strength. Strength is a function of the frequency with which a generalized pattern reoccurs across instances in the same category and has been used to make correct judgments. Depending on whether a learning-trial decision is correct or incorrect, the responsible pattern may be strengthened or modified, and new patterns may be created. Learning occurs as this set of mechanisms, in some particular computational instantiation, produces and refines an internal representation of category knowledge in the form of feature pattern associated with category membership.

We implemented these assumptions in a system that we call LANA. The remainder of this section provides the processing details of the theoretical assumptions outlined before. Table 4 presents the control algorithm for how learning proceeds in this framework. We divide our explanation of the framework into two main sections: structures and their organization, and the processes that operate on these structures. Parameters that we manipulated and tested for passive and active learning models are presented later.
Structures and Their Organization
There are three important features about the structures that LANA models manipulate and how they are organized: the distinction between instance patterns and generalized patterns, the distinction between working memory and long-term memory, and the notion of pattern strength.

Representing Instances and Generalizations
Instances are represented as ordered lists of fixed feature values, corresponding to the abstract notation used for category items in Table 1, with an associated category tag, for example, $12134\rightarrow$ Category 1.

A generalized pattern is represented as an ordered list of constrained and variable feature values with an associated category tag. The pattern $1I--2\rightarrow$ Category 1 represents the association of Value 1 on the first two dimensions and Value 2 on the last dimension with a Category 1 assignment; the dashes signify that values on Dimensions 3 and 4 are unconstrained in this particular association.

Instance patterns and generalized patterns differ only in their form, that is, the generalized pattern allows for variable matching on certain dimensions, whereas instance patterns are a completely constrained set of feature values with an associated category tag.

Working Memory and Long-Term Memory
LANA models have a working memory and a long-term memory, distinguished by the kinds of structures that each contains and the kinds of processes that operate on those structures. Generalized patterns are held in long-term memory. Instance patterns may also populate long-term memory; whether or not they do is a model-dependent parameter that we will discuss later.

Working memory consists of (a) the last three instances studied during learning trials and their correct category assignment, and (b) any additional patterns, either instance patterns or generalized patterns, that are retrieved from long-term memory as a part of the classification process (described in the following section). In addition to the last three instances, working memory also holds the category decision for the current learning-trial instance and whether that decision is correct or incorrect.

Pattern Strength
Both instance patterns and generalized patterns are assigned a strength when they are created. The strength of a generalized pattern is changed under two circumstances. The first case is when it is incremented on correct learning trials or decremented on incorrect learning trials. The second circumstance is when a generalized pattern is recreated, that is, the same generalized form reoccurs. The impact of pattern frequency on transfer per-
Performance has been demonstrated in these kinds of category-learning tasks (e.g., B. Hayes-Roth & F. Hayes-Roth, 1977) and strengthening mechanisms governed by stimulus frequency or past success exist in some form or other across a wide range of learning models and paradigms. The strengthening and weakening functions change a pattern’s strength by adding or subtracting one unit of strength.

If something within this framework could be viewed as an automatic mechanism, then strengthening is it. Strengthening constitutes a powerful learning mechanism in induction because pattern strength implicitly represents the knowledge that a particular association is either frequent or useful. When retrieval processes are defined in part on pattern strength, then strength directly determines which patterns will be refined and hence indirectly determines the nature of new patterns that will be formed. In machine-learning parlance, this influence of strengthening can be viewed as a bias (Utgoff, 1986) that affects the direction of search in a space of possible concept definitions.

Processes
There are three main processes that use, modify, and create the structures described previously retrieving patterns from long-term memory, making a decision, and creating new generalizations. The next sections follow the control algorithm in Table 4 to explain these mechanisms.

Retrieving Patterns from Long-Term Memory
When an item is presented during a learning or classification trial, the system computes a retrieval score for all patterns in long-term memory patterns. The retrieval score is an equally weighted addition of a pattern’s strength and its similarity to the presented item. The similarity between two patterns is a function of the number of matched fixed features minus the number of mismatched fixed features between two patterns (i.e., variables do not count as contributing to either a match or mismatch). Patterns that exceed a similarity threshold value (set at 2 for models presented here) and exceed a strength threshold value (a model-specific value) are added to working memory.

Making a Decision
At this point, working memory contains three instance patterns from previous trials plus any patterns retrieved from long-term memory. All these patterns are ranked according to their retrieval scores. A classification judgment is based on the pattern with the highest score. In the case of a

2 The retrieval score for an instance pattern is its similarity plus a strength value. The strength value assigned to instance patterns is varied as a model-dependent parameter that affects the sorting of instance and generalization patterns and is discussed in a later section.
tie, one of the tying patterns is selected at random. If no patterns meet the strength and similarity retrieval thresholds, the classification decision is made randomly. On learning trials, the system receives feedback on the accuracy of the decision; on testing trials, no feedback is given and a typicality rating is computed instead (this is explained later).

Creating Generalized Patterns

After feedback occurs on learning trials, generalized patterns are formed through two possible mechanisms. The first mechanism operates when the system correctly classifies an item presented on a learning trial, using either an instance pattern or a generalized pattern in working memory. In this case, the maximally specific generalization of the two patterns is created. For example, using 12231→Category 1 to classify correctly a learning-trial item like 11214 yields 1-2→→Category 1. The pattern used on learning trials may either be an instance pattern, as in this example, or a generalized pattern.

The second learning mechanism that creates generalized patterns occurs on incorrect trials. When a generalized pattern leads to an incorrect category judgment, a discrimination process creates a new pattern that would correctly classify the item, using differences between the given item and the applied pattern. Suppose the pattern --2 4→Category 2 were used to classify incorrectly the Category 1 item 11214. The discrimination process isolates differences between the pattern associated with Category 2 and the given Category 1 pattern. One such discriminating difference is the presence of the Value 1 on the first dimension. Thus, a new pattern, 1-2-4→Category 1, is formed to represent a feature combination that distinguishes this Category 1 item from the similar Category 2 pattern. This controlled discrimination approach is similar to those proposed in other learning frameworks (e.g., Anderson, 1987).

There can be more than one discriminating difference between the two patterns, a kind of credit–blame assignment problem, which is why learning on error trials can be difficult. In our previous example, there are five different discriminating patterns that can be formed by unique combinations of the distinguishing features. The discrimination process for all the models we present in the following created only one new discriminated pattern, randomly selected from the set of possibilities.

Remarks on the Framework

As part of our investigation of simulation for this task, we explored other variations of the processes and structures described previously, but do not discuss them in this article. This included alternative definitions of the generalization and discrimination mechanisms and alternative definitions of working memory and processes. We discovered in some cases that minor changes to one function or one parameter produced small performance dif-
Simulating learning mode and variance effects

Table 5: Simulation Models and Parameter Values

<table>
<thead>
<tr>
<th>Model</th>
<th>Match to Retrieve</th>
<th>Availability</th>
<th>Selective Remembering</th>
<th>Instance Memory</th>
<th>Generalization Bias</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>partial</td>
<td>delayed</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>partial</td>
<td>delayed</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>partial</td>
<td>delayed</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>partial</td>
<td>immediate</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>full</td>
<td>delayed</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>partial</td>
<td>delayed</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>full</td>
<td>immediate</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>8</td>
<td>full</td>
<td>immediate</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>full</td>
<td>immediate</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>10</td>
<td>full</td>
<td>immediate</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>full</td>
<td>immediate</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>full</td>
<td>immediate</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>full</td>
<td>immediate</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

References, and our investigations were aimed at identifying those mechanisms that caused big differences. For example, whether working memory holds three or four previously seen instances has little impact on performance; whether it is allowed to be infinitely large certainly does. We believe that small changes to parameter values (e.g., working-memory size set at 3 vs. 4) should not lead to big swings in performance; if they do, there may be something more fundamentally wrong with the model or the representation.

The most significant assumptions of the framework we described before are: (a) the presence of generalization as a similarity-detection mechanism and the circumstances under which it creates and modifies patterns; (b) the retrieval scheme based on pattern similarity and strength; and (c) the recreation assumption under which the strength of reoccurring feature patterns is increased. Within this framework, we explored alternative models for passive and active learning, defined as unique configurations of parameter values that controlled this fixed set of mechanisms. The next section presents the parameters we investigated and our rationale for doing so.

Defining alternative simulation models

We viewed an active learner as someone engaged in an explicit hypothesis-testing strategy and considered how that descriptive account might be realized as some configuration of particular parameter settings that controlled the processes defined by our framework. Table 5 presents the set of models, defined by five parameter settings, that we discuss in the following. These
parameters determine (a) whether a long-term memory pattern must fully match or need only partially match some presented pattern to be retrieved, (b) whether availability of a new abstracted pattern is delayed or immediate, (c) whether there is selective memory for only the best (strongest) patterns from one learning block to the next, (d) whether there is a bias to base learning-trial decisions on retrieved patterns over recently seen instances, and (e) whether there is long-term memory for instance patterns.

The general framework (described in the preceding section) is a set of learning and retrieval processes operating within a general architecture. The parameters we detail in this section can be viewed as governing the specifics about how particular processes within this general framework operate. It is the combination of processes and data structures that define a representation. Thus, one set of values for parameters that control how abstracted patterns are retrieved or used defines one representation, another set of values for those same parameters constitutes a different representation. In our discussion of these parameters, it will become apparent that we had clearer descriptive notions about active learning than we did about passive learning: the passive learning notions were often defined as complements to these. We now give details on how each of these parameters and their possible values relates to our view of descriptive accounts of active or passive learning.

Full versus Partial Matching for Pattern Retrieval

In the framework that we defined, the generalization mechanism operates as the similarity-detection process. We assumed that this occurs under either an active or passive learning strategy. What these feature patterns represent can be defined differentially by subjecting them to different processes. One way to distinguish these feature patterns as explicit rules for an active learning model is to insist that they fully match an input pattern in order to classify it. If abstracted patterns are category-membership rules that an active learner wishes to verify, then the conditions of any particular rule either match a presented case or they don't. Hence, the retrieval and use of any abstracted pattern would be contingent on whether its feature-value specifications are completely present in some input pattern that needs to be classified.

We implemented a matching parameter that specified whether or not abstracted patterns had to fully match a presented item in order to be retrieved. Under a full-match constraint, the generalized pattern \textit{Category 1} would not be retrieved when item 11422 was presented because the item does not match the specified Value 3 on Dimension 4 in this pattern. When this parameter is set to allow partial matches, the generalized pattern in the preceding example would be retrieved. We conjectured that the partial match setting on this parameter is akin to a more "analogical" use of category-level information, relative to the exact match interpretation of rules.
Forcing abstracted patterns to fully match a presented item also distinguishes them from instance patterns. Instance patterns in working memory can only partially match any presented item (except for instances matching themselves), independent of this matching parameter. When the partial matching scheme is in effect, the similarity of abstracted patterns and of instance patterns to a presented item is computed in the same way.

**Delayed versus Immediate Availability**

The idea that different processing constraints on the same data structure constitute different representations can be extended to what we might call "availability." If we want to treat abstracted feature patterns as deliberate and consciously formed hypothesis, then such feature patterns should be available for use as soon as they have been detected. That is, if the system detects a similarity between two instances of Category 1, the resultant generalization—to represent a deliberately formed hypothesis—should be available for verification as soon as it is applicable. Alternatively, these same abstracted patterns formed under a passive learning strategy can represent regularities detected perhaps less deliberately and hence more slowly across a larger set of input patterns. We implemented this distinction, which we call immediate versus delayed availability, through a parameter that specified how soon after its formation an abstracted pattern was eligible for retrieval from long-term memory. Under immediate availability, once a pattern was formed, it was available for retrieval. Under delayed availability, a pattern had to receive two more strength units beyond this initial strength in order to be retrieved. The only way this can happen is for the pattern to be re-created through two more generalization operations on subsequent learning trials.

**Selective Remembering for Strong Patterns**

We have been characterizing active learning as a hypothesis-testing approach. A plausible by-product of a deliberate hypothesis-testing strategy might be to discard hypotheses that have proven less useful and selectively retain only the most successful membership rules. We instantiated this assumption through a parameter that, after each of the four consecutive learning-testing blocks, removed from long-term memory any pattern that did not have the minimum strength to be retrieved. This step might seem irrelevant: If a pattern doesn't meet the minimum strength to be retrieved, why bother to remove it? However, a pattern in this framework can accrue strength not only from being retrieved and successfully used, but by being re-created on separate occasions. To remove patterns that are below a retrieval threshold serves to contract the search space of what the system might identify as membership rules, if given sufficient opportunity. Put another way, it forces the system to move in just those directions that have
proven promising early on in the learning process, and to abandon apparently unpromising directions.

From a hypothesis-testing viewpoint, selective remembering can be seen as focusing on the early (apparent) winners and eliminating the early (apparent) losers. The compliment of this process for a passive learning strategy is simply not to engage in any deliberate pruning. Although the strongest patterns are always favored in the retrieval process, even patterns below threshold still have the chance to accrue strength via re-creation. More simply, the space of patterns is not controlled in any deliberate sense.

**Bias to Use Generalized Patterns for Decision Making**

In the next two sections we describe two parameters related to the use of previously seen instances to make learning-trial decisions. There are two places instance information can come into play in our general framework: whether recently seen instance patterns or retrieved generalizations are preferred to make learning-trial decisions, and whether there is any long-term memory for specific instance patterns.

Under a strong hypothesis-testing model, we might conjecture that a learner would prefer to use a retrieved rule to classify a learning-trial item even if a more similar instance pattern was recently seen and remembered. This is consistent with the notion that abstracted generalization patterns, coupled with certain types of processes, represent rules or hypotheses to be verified. This bias was implemented by assigning a strength value to instances that was either negligible or equal to the strength value given to newly formed generalizations. We call this parameter "generalization-use bias." It served to sort instance patterns and retrieved generalizations on Step 3 of the control algorithm (see Table 4). When this bias is on, instance patterns receive a negligible strength value, forcing all instance patterns to the bottom of the sort. This ensures that any retrieved generalized pattern would be selected over any instance pattern for a learning trial decision, regardless of its similarity to the presented instance. When the generalization bias is off, working-memory instance patterns receive a strength value that is equal to any newly formed generalization. This allows working-memory patterns to compete in the sorting process on a more equal footing with retrieved generalizations, and an instance pattern may win this competition over a retrieved generalization if it is more similar to the instance that the system must classify. This technique of assigning different strength values to instances to affect the sort was nothing more than a programming convenience to implement this sort of bias.

What are the implications of this bias from a learning perspective? When instances are used in this kind of analogical fashion to classify some input pattern, the generalization mechanism will kick in. So the more instance-based classifications, the more diverse the pool of generalizations becomes.
When generalizations are preferred for a learning-trial decision, then an existing pattern is likely to be strengthened (if the decision is correct) rather than new interinstance similarities being detected. An indirect effect of this biasing parameter may be to control how quickly the system's learning mechanisms are focused on strengthening existing patterns rather than forming new patterns for the pool.

**Memory for Instance Patterns**

The second opportunity to increase the reliance on instance information within the framework is to move instance patterns out of working memory and put them into long-term memory. This would allow a studied instance to influence classification and learning beyond the three-trial working memory. The issues associated with this process were, first, which instances are remembered, and second, in what form are they remembered. We decided that (a) instances from working memory that had been used twice to make classification decisions would be remembered in long-term memory; (b) these instances would be imperfectly remembered—one or two features were randomly dropped from their feature pattern; and (c) once migrated over to long-term memory in this form, these patterns would not be distinguished from, or processed differently than, generalized feature patterns. This last point is important to appreciate: Once an instance pattern makes it to long-term memory in this form, it is eligible to be retrieved, strengthened, or discriminated just like any feature pattern that arises from a generalization process. We call this parameter "instance memory" and we conjectured that it would play a larger role in modeling passive learning than in modeling active learning.

**Remarks on Model Parameters**

The parameters outlined earlier are a small subset of the parameters and values that we systematically explored to create and test different simulation models for passive and active learning on this task. In principle, every mechanism is a model parameter and may be related to a computational account of learning strategies, for example, what is remembered in working memory, details governing how or when generalization or discrimination processes occur, what kind of similarity metric is used, what kind of strengthening function is used, and so forth. The set of possible parameters within this one framework, and possible values on those parameters, make for an extremely large set of distinct models. What we present here is not the chronological or complete account of what parameters worked or did not, but a distilled subset of what mechanisms had the consistent and significant effects on performance in this framework.

The models in Table 5 are the ones for which we present and evaluate simulation results. We also implemented an instance-based model that
captured the general themes originally proposed and tested in Medin and Schaffer's (1978) context model for category learning and Hintzman and Ludlum's (1980) MINERVA computer simulation. In MINERVA, category instances are represented as a set of both properties and property relationships, each with separate associated strengths. The initial encoding of the memory trace is simply a copy of the item description. Individual properties are lost from the example trace over time in an all-or-none fashion; there is no other forgetting mechanism. A new item is matched for similarity against all traces in memory and the degree of match between the item and the trace is computed using a nearest neighbor formula. A retrieval threshold controls the minimum degree of match and the size of the retrieval set. Classification of a new item is based on the exemplar most similar to the test item.

We captured the essence of this approach within the LANA framework in the following ways to create an instance-based model. First, we turned off the generalization and strengthening mechanisms. Second, instances were perfectly remembered for one trial, after which a randomly selected set of properties was removed from the instance pattern. The result (equivalent in form to a generalized pattern) was placed in a permanent long-term memory. When an item was represented, all patterns that had at least one feature in common with the presented item were retrieved and ranked according to our similarity metric. The highest ranking pattern was used to make classification decisions; in the case of ties, a random choice of the highest ranked patterns was made.

**TASK SIMULATION AND RESULTS**

**Method**

For each stimulation model, we followed the procedures used in Elio and Anderson (1984) in the following ways. Each model run corresponded to one human learner. For each run, the system received four samples of training items that were constructed according to either gradual variance or representative variance training conditions (see Table 2). The results we present in the following are the averages of 15 gradual variance and 15 representative variance runs per model; the training samples for each run contained different, randomly selected items in accordance with the sampling rules in Table 2.

Like the task for human learners, each run was organized into four blocks, each block consisting of learning phase and a test phase. The general course of events for each block's learning trial follows the control algorithm in Table 4. In the learning phase of each block, the system received three passes through one sample of 20 items, each pass using a different random order for the items. An item from the sample was presented to the system
for classification into either Category 1 or Category 2. All patterns that met the strength and similarity thresholds were retrieved, scored, and ranked according to the methods described earlier. The best-ranked pattern was used to make the classification decision. Pattern abstraction and strengthening processes than occurred. Working memory was updated with the new item and the oldest instance pattern in working memory was dropped. If the instance-memory mechanism was in effect, any instance that was used twice for making a learning-trial decision was transferred to generalized-pattern memory as described earlier.

Each learning phase was followed by a test phase in which the system classified all category items. Only patterns retrieved from long-term memory were used on these testing trials, there was no memory for recently seen instances. It seemed implausible that the last three studied instances would be retained throughout the testing phase because (a) there was a distinct break between each learning and testing phase for human subjects, and (b) there were over 100 testing trials, which presumably created some degree of interference for recently studied items.

We also attempted to model typicality judgments because this measure was particularly sensitive to the variance conditions and task manipulations for human learners (see Table 3). We conjectured that, in our framework, such judgments could be a function of the similarity, strength, or both, of the pattern used to classify an item. Therefore, we computed typicality ratings using each metric. As in the original experiments, the mean typicality rating score for a subject was computed as the mean of the sum of typicality scores on correct classifications minus the sum of the typicality scores on incorrect classifications.

The LANA system was implemented in Lisp on a Xerox Workstation under the Lyric operating system.

Results
In this section, we present the results for each of Table 5's models trained under both the representative variance and gradual variance conditions. As we noted earlier, our simulation goal was to identify (a) those models that generated better transfer performance given representative training, and (b) those models that generated better transfer performance given gradual variance training. The former set of models constitutes candidate active learning models and the latter set corresponds to candidate passive learning models.

1 We used the same three measures associated with retrieved patterns that we computed on earlier steps of the control algorithm. Specifically, the typicality rating based on strength was the pattern's scaled strength value; the typicality rating based on similarity was the same similarity metric described earlier (the degree of match minus the degree of mismatch); the typicality rating based on both these features was what we referred to earlier as a pattern's retrieval score (the equally weighted addition of the strength and similarity value).
We evaluated each model's fit to both the observed active learning and to the observed passive learning data by computing a Pearson's correlation coefficient. Each correlation coefficient is based on eight observed-simulated data pairs: the means for each of the four transfer tests under both representative variance and gradual variance training. Thus, for a given model, its four transfer-test accuracy means under representative variance training and its four transfer-test accuracy means under gradual variance training were paired with the eight corresponding observed means for subjects given passive learning instructions (yielding one correlation coefficient) and then again with the eight corresponding observed means for subjects given active learning instructions (yielding a second correlation coefficient). This approach served to test model's ability to simulate the relative ordering of representative variance and gradual variance performance means under either the passive learning or the active learning instructional set.

A candidate passive learning model is one that has a significantly good fit to the observed passive learning data and a poor fit to the observed active learning data, with the reverse being true for a candidate active learning model. Given this criterion, we should ask how similar the observed passive learning and active learning data points are to each other because we are insisting that a good model of either type of learning would fit one data set well and the other data set poorly. The correlation coefficient between the observed passive learning and active learning accuracy means is -.05; for typicality, it is -.23.

Table 6 gives the correlation coefficients for the models presented in Table 5. The Appendix gives the means (based on 15 runs per training-variance condition) generated by these models. As we noted earlier, this model set is a subset of all the models we tested. We focus on these models because they clarify the relative contribution of the different parameters we presented earlier. We have organized Table 6 into two groups of models. Group 1 models are the models that have a good fit to the passive learning data and a poor fit to the active learning data. Group 2 models are the models that have a good fit to the active learning data and a poor fit to the passive learning data. The correlations for the typicality data are based on the strength plus similarity metric for computing typicality judgments. The exception is the instance model: Typicality ratings in that model are based on similarity alone. At the end of this section, we discuss the differences among the three alternative metrics that we used for generating typicality ratings.

Before discussing individual models in detail, we want to underscore the two signature differences between the Group 1 models that fit the passive learning data and the Group 2 models that fit the active learning data: (a) Group 1 models retrieve patterns under a partial match scheme whereas Group 2 models use the full-match scheme; and (b) Group 1 models
TABLE 6
Pearson’s Correlation Coefficients for Observed and Simulated Data

<table>
<thead>
<tr>
<th>Observed Learning-Mode Data Set</th>
<th>Accuracy</th>
<th>Typicality</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Passive</td>
<td>Active</td>
</tr>
<tr>
<td>Group 1 Models</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>.68</td>
<td>-.06</td>
</tr>
<tr>
<td>2</td>
<td>.37</td>
<td>-.86</td>
</tr>
<tr>
<td>3</td>
<td>.87*</td>
<td>-.25</td>
</tr>
<tr>
<td>4</td>
<td>.35</td>
<td>-.21</td>
</tr>
<tr>
<td>5</td>
<td>.19</td>
<td>-.90</td>
</tr>
<tr>
<td>6</td>
<td>.75*</td>
<td>-.11</td>
</tr>
<tr>
<td>Instance</td>
<td>.79*</td>
<td>.53</td>
</tr>
<tr>
<td>Group 2 Models</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>-.15</td>
<td>.73</td>
</tr>
<tr>
<td>8</td>
<td>.07</td>
<td>.85*</td>
</tr>
<tr>
<td>9</td>
<td>-.07</td>
<td>.75*</td>
</tr>
<tr>
<td>10</td>
<td>.18</td>
<td>.76*</td>
</tr>
<tr>
<td>11</td>
<td>.03</td>
<td>.84*</td>
</tr>
<tr>
<td>12</td>
<td>.05</td>
<td>.84*</td>
</tr>
<tr>
<td>13</td>
<td>-.27</td>
<td>.73*</td>
</tr>
</tbody>
</table>

* p < .01, one-tailed (6 df). A one-tailed test is appropriate because we are interested only in positive correlations between observed and predicted data points. Each cell represents the correlation between a simulation model’s representative and gradual training data and the corresponding observed data set for the learning-mode conditions defined by the column heading.

are delayed-availability models but Group 2 learning models are immediate availability models. Put another way, it was not possible to get a good fit in our framework to passive or active learning without this particular configuration of these two parameters. Now we consider the models and their fits to the data in more detail.

### Passive Learning Models

Several models in Group 1 meet our criteria for candidate passive learning models: significant correlations to the passive learning data and correspondingly poor fits to the active learning data. The correlations for Model 3, Model 6, and the instance model are significant for both the accuracy and typicality data. Other Group 1 models have significant r values for the typicality data but not for the accuracy data. We computed a t test for differences between two dependent correlation coefficients to assess whether two different models with significant correlations were significantly different from each other. Applying this analysis to the accuracy correlations, Model 3, Model 6, and the instance model were not significantly different from each. For the typicality correlations, Model 3 was different than
Model 1, \( t(5) = 13.89 \), the instance model, \( t(5) = 3.92 \), and Model 6, \( t(5) = 7.95 \). The correlations of these last two models were not different from each other. Finally, Model 3's typicality correlation was not different from Model 2's correlation.

We included Model 5 in Group 1 to illustrate the effect of changing the matching parameter from partial (as in Model 2) to full: Its fits to the accuracy and typicality data are considerably worse. Model 4 illustrates that an immediate availability model with a partial match parameter yields acceptable fits to the passive typicality data, but fails to simulate the accuracy data.

Whereas these results might point to Model 3 as the "best" account of the data, we do not want to single out it or any other model in this regard. There are limitations associated with a statistic like Pearson's \( r \) for identifying any particular model as the preferred one given the data sets we had. The means for all the models that had high accuracy and typicality correlations are qualitatively different from each other and, from a more subjective viewpoint, in their similarity to the observed data (see Appendix). For example, the instance model's representative variance accuracy and typicality means are often higher than the corresponding gradual variance means, unlike the observed data. Model 6's accuracy means for the gradual variance case shows a slower improvement than what was observed for this condition. Thus, it is more important to focus what these models as a set suggest, namely, that two parameters—partial matching and delayed availability—were critical in modeling passive learning in our framework.

Model 3 did not use selective remembering, that is, it did not prune low-strength patterns from long-term memory between training blocks. This is a feature of the instance model as well, which continues to accumulate fragments of studied instances. Both these models have better accuracy fits than the models that computed the relatively strongest patterns between blocks and explicitly deleted the rest. This outcome seems consistent with a view of passive learning as the absence of any deliberate manipulations of the evolving pattern base. The good fit of a purely instance-based model—one with no learning mechanisms like generalization or strengthening—is important for a functional understanding of a model like Model 3. We return to this topic in the General Discussion.

Before leaving the Group 1 models, we want to remark on the roles of the instance-memory and the generalization-bias parameters. The Group 1 models in Table 6 do not exhaust all possible combinations of alternative settings for these two parameters, with the other three parameters held constant. However, from the larger set of models that we considered, we know the following. First, having some degree of long-term memory for instance

\* All \( p \) values are significant at the .05 level for a two-tailed test.
patterns makes a contribution to transfer accuracy and this alone improves model fits. This makes sense because the instance-based model uses this mechanism alone as its sole learning mechanism. Therefore, memory for instance patterns can be viewed as an additional learning mechanism for some models: It added more patterns to long-term memory that were subsequently treated just like any other generalized pattern. Second, any effects of the generalization-bias parameter were apparently overshadowed by the effects of the other parameters in this task. Relative to the matching and learning-rate parameters, the instance-memory and generalization-bias manipulations played a minor role in simulating passive and active strategy effects on this task. Put another way, we consistently obtained pretty good active learning and passive learning fits with different combinations of values on these two parameters, but not with different combinations of the matching and availability-rate parameters.

**Active Learning Models**

The Group 2 models are those that had good fits to the active learning data and poor fits to the passive learning data. All have significant correlations to both accuracy and typicality data sets; none of these correlations are significantly different from each other for either data sets. However, from a more subjective evaluation of the observed means in Table 3 and the simulated means in the Appendix, it is clear that none of the models had the nearly flat learning curve associated with gradual variance training and the steeper curve associated with variance training.

Some pairwise comparisons across Groups 1 and 2 illustrate the roles that the immediate availability and full-matching parameters play in modeling passive and active learning data in this framework. Group 2's Model 11 is just like Group 1's Model 4, except that it forces a full match to retrieve abstracted patterns, underscoring that this difference is critical in simulating the active learning data. Group 1's Model 5 is also like Group 2's Model 11, except that Model 5 is a delayed-availability model: This combination of parameters does not produce performance close to either the passive or the active learning data.

There are variations among the Group 2 models on whether or not they (a) have selective memory for good patterns, (b) have long-term memory for instance information, and (c) are biased to use generalizations on learning-trial decisions. As we found with the passive learning models, these three features played a negligible role relative to the full-match and immediate availability parameters in simulating the active learning data. For the Group 1 models, memory for instance information contributed to accuracy results, but that benefit seems limited only to the partial match, delayed-availability context. For full-match models, fragments of instance patterns in long-term memory from the first gradual variance sample have the same poor chance
of fully matching new variations in the training instances as would the abstracted generalizations. This is because we treated all patterns in long-term memory the same, regardless of whether they originated as an imperfectly remembered instance or as a generalized feature pattern. As for the bias to test abstracted patterns over better matching short-term memory instance patterns, that too seemed to contribute relatively little variation in model performance (e.g., Model 8 vs. Model 11) in the context defined by the immediate availability and full-match parameters.

Typicality Metrics
Each model computed typicality scores using three different metrics: the strength of the pattern used to classify an item, its similarity to the item, and a combination of strength plus similarity. The results presented in Table 6 are based on the last metric, but we can use the $t$ test statistic for differences between dependent correlation coefficients to evaluate whether or not some of the best fits to observed typicality would be any different using either of the first two metrics.

From the Group 1 models, we selected Model 3. The correlation of Model 3's strength-based typicality ratings to the observed passive typicality data ($r = .94$) was significantly different than the correlation based on the strength plus similarity typicality ratings given in Table 6, $t(5) = 3.61$; it was not different than the correlation based on a similarity metric ($r = .97$), $t(5) = 0.12$. For the Group 2 models, we looked at Model 8 and Model 12. There were no differences among the correlations achieved with the three alternative typicality metrics for either model. However, the rank ordering of the simulated means generated by these two models is slightly different, depending on which metric is used. For Model 8, both the strength-based typicality means and the strength plus similarity means have significant Spearman's rank order correlations with the observed means ($r = .79$ and .81, respectively; $p < .05$), but the similarity-based typicality means do not ($r = .50$). For Model 12, the simulated typicality means for each of the three metrics yielded significant Spearman's rank order correlations with the observed typicality means.

There is no clear story here on modeling typicality within this framework, except that some measure of pattern strength is important. A pattern's strength reflects its past success in making classification decisions, which is related to the frequency with which the pattern occurs across exemplars. Pattern strength contributes to determining which patterns are retrieved and used for both classification and typicality judgments. So a typicality rating computed on pattern similarity alone has indirectly been influenced by pattern strength. Typicality is a complex matter, to understate matters, and we have no claims that these results speak to larger issues of typicality. Certainly, we could have computed both classification and typicality decisions in other ways, for example, based on the votes of all retrieved patterns associated
with both categories. We note only that a relatively simple approach sufficed for the simple feature-value vectors that described category members in this task.

GENERAL DISCUSSION

We used simulation as a method for identifying and evaluating alternative process models of performance in a category-learning task that manipulated two features: the order in which the learner encountered category variance and an instructional set that induced a more passive or a more active approach to the task. This use of simulation tested both the descriptive notions of active and passive learning, and their translation into particular computational mechanisms within a given framework. In assessing the results of a simulation effort, this distinction is an important one. Replicating the behavior of any large system is difficult, because small design and implementation decisions may have subtle effects on the system's overall behavior. For this kind of work, the important result is the functional analysis of why a set of computational mechanisms succeeded in simulating the observed data. In the remainder of this article, we want to come full circle, lifting the simulation results out of their computational realization to a more functional analysis. We also discuss several issues related to our use of simulation to identify process models for performance on this task.

A Functional Analysis of the Passive and Active Models

We investigated alternative models for passive and active learning within a single general framework. We defined a model as a particular set of parameter settings governing the processes that defined the framework. Within this framework, we found that delayed availability and partial match models yielded results in line with the observed passive learning data, and that immediate availability, full-match models gave results similar to the observed active learning data. To understand the functional effects of these mechanisms, it is helpful to characterize the learning task as a search process in a pattern space for those patterns that are highly predictive of category membership.

Let's first consider the effect of what we call delayed availability in our framework and how this yields an advantage for gradual variance training. First, by requiring that abstracted patterns recur several times before they are retrievable, the system continues to use recent instances to make learning-trial decisions. If those instances are not very similar to the pattern that must be classified, then random guesses are made and no learning occurs. If an instance-based decision is correct, a new generalization is created. Put another way, the longer the system is making instance-based decisions during learning, the more the space of patterns is seeded with generalizations, and the more diverse those generalizations are. With a low-variance sample,
the patterns populating that space are relatively constrained and the likelihood of any of them being re-created (and garnering their additional evidence) is higher than it is for a representative sample with more interitem variance. This gives the edge to gradual variance learning under a passive mode: the low interitem similarities within the sample yield regularities that the system can use sooner. Any given regularity in a representative variance sample does not recur as often.

Although the regularities learned under gradual variance training are by definition nonrepresentative, the partial matching mechanism allows these patterns to accommodate the variation of instances in subsequent training samples. As new variation is encountered on subsequent samples, these nonrepresentative patterns will either partially match a good portion of them and be strengthened further, or will be incorrect and become fine-tuned via a discrimination mechanism. Because we based retrieval on similarity as well as strength, the selection of a pattern for a classification decision will be biased towards larger and hence more predictive patterns. The data in Table 3 show that, when human learners followed a passive strategy, the gradual variance training advantage was consistent across all four training blocks, unperturbed by the introduction of new types of category members.

Why would the complement of these parameters yield results like those we found for active learners, namely, an advantage for representative training over gradual variance training? The effect of active learning instructions on human learners was something of a hindrance in the gradual variance condition. Under the immediate availability approach, as soon as a generalization is created, it can be retrieved. This removes the disadvantage associated with a representative training sample: Any regularity that is detected will be used immediately. This is consistent with the notion that an active learner is on the lookout for any regularity to verify and use as a rule or hypothesis. It is also consistent with the empirical finding that representative variance subjects were better off when learning under an active strategy than under a passive strategy. When the system is trained with gradual variance samples, its initial set of rules will not fully match the variation in later training samples. In this case, the system can do nothing except revert back to instance-driven learning-trial decisions. In effect, it must start learning new rules again.

Under this account, a gradual introduction of category variance represents a moving target to this immediate availability full-match model of an active learning strategy. First the learner converges on one localized area of the search space, then finds that new input is not fully accommodated by those patterns applied as rules, and then starts over again in other areas of the space as dictated by the current set of instances. All that can happen is for the system to be instance driven again because its current rule
set is not appropriate. This view of active learning as being driven by instance information may seem counter to its characterization as a rule-driven learning mode. However, it is entirely consistent with the process of being an active learner, that is, attending to currently emerging regularities. For the active learner given gradual variance, different regularities keep emerging; for the active learner consistently given representative samples, the initial regularities remain useful.

**Learning Under an Instance-Based Model**

Two different models, the LANA Model 3 and an instance model, offered good accounts of passive learning under representative and gradual variance training. The instance model might not be thought to have learning mechanisms, but in fact it does have one: It caches instance fragments. From a methodological viewpoint, it is important and useful to find that models developed under different frameworks simulate the same data set. It helps to identify the functional effects of mechanisms in either model. This in turn elucidates general principles about the phenomenon of interest that extend beyond the particular simulation program.

Before we discuss the functional similarities between these two models, it is important to appreciate how an instance model with no explicit learning mechanisms produces improvements in classification over a series of training samples. The sort of instance model we implemented creates a population of imperfectly remembered instances. Repeated exposure to a stimulus set increases the probability that fragments of frequently occurring (hence, predictive) patterns will dominate the representation, increasing the likelihood that they will be retrieved. This in turn produces improvements over blocks as more fragments are added.

Both the instance model we tested and Model 3 are more similar than they might otherwise appear. Both used a partial match retrieval scheme, both favored larger matching patterns for a classification decision, and neither model pruned feature patterns from their pattern base. The functional effect of instance caching, especially for large stimulus sets, may be similar to an abstraction model that uses a strengthening mechanism to reflect a sensitivity to feature-pattern frequencies. Indeed, for large and variable categories like those used here, the predictions of instance-based models and rule-based models may be hard to disentangle (Elio & Anderson, 1981).

We did not explore variations of the instance model. The challenge is using this type of model to accommodate strategy effects of the sort observed under the active learning condition. Because the model just continually adds instance fragments to long-term memory, some additional mechanism would seem necessary to disrupt learning under gradual variance training. One possibility is a process that continually assesses characteristics of the emerging pattern space and uses this assessment to influence dynamically how
subsequent instances are processed. For example, highly salient (frequent) features could influence a selective attention mechanism that controls encoding and retrieval. Unexpected variance in a new sample of category members may disrupt performance if selective attention, set by a biased sample, biases what features of the newly variable instances are encoded and directs the retrieval process to an inappropriate set of instance patterns. Parameters changes to a selective attention mechanism can produce different strategy effects within a single, instance-based framework (Medin & Smith, 1981) and this notion has parallels in other learning theories, such as Billman and Heit's (1988) model for observational learning and Grossberg's (1976, 1987) adaptive resonance theory. When the learning task is characterized as a search process, a dynamic selective attention mechanism functions in much the same manner as the selective retrieval and memory mechanisms in abstraction models that we discussed before: It serves to localize the learning mechanisms in some areas of the space to the exclusion of others, which is detrimental if the initial training experience has some kind of bias.

Relation to Other Work on Strategy Effects

Much work has been done to account for learning-strategy effects within specific models (e.g., Medin & Smith, 1981; Reitman & Bower, 1973) and more general theoretical distinctions have been proposed between implicit and explicit learning, as defined by Reber and his colleagues (Reber, 1969, 1976; Reber & Allen, 1978) and between analytic and nonanalytic learning as defined by Brooks (1978). The instructional manipulations and task demands investigated have some correspondence to these distinctions, but there are important differences as well. For the task we simulated, learners knew that two categories existed and that their primary task was to learn how to classify members correctly into one category or the other. This task is unlike those that are often used to investigate implicit learning, in which learners are unaware of some structure underlying the material being studied and receive some unexpected test concerning this structure. On the other hand, the emphasis on rule testing and hypothesis generation in the task we simulated served, we believe, to push learners closer to one end or the other of some hypothesis-testing continuum. The net effect may be similar to instructional manipulations used to induce analytic strategies or explicit search for some particular structure. Indeed, the task requirement for active learning subjects to report upon their current hypotheses about critical rules or features was a powerful inducement to take a strong hypothesis generation and testing approach to the material.

The work reported here is similar to other efforts at investigating how a common computational framework may account for different strategy effects instead of positing a different model to account for each different strategy. Medin and Smith (1981) argued for modeling strategy effects as
changes to parameters settings within a single framework rather than as alternative learning mechanisms. They demonstrated that Medin and Schaffer's (1978) context theory can account for different strategy effects with changes to saliency weights on stimulus features. Nosofsky et al. (1989), however, reported evidence suggesting that rule-based models offer better accounts of performance under rule-based strategy instructions than do exemplar-based models. The task we modeled and the simulation work reported here do not speak directly to the issue of exemplar-based versus rule-based processing. However, the categories in the task we investigated were relatively large, and as we noted earlier, an instance-based model for such large and variable categories may be indistinguishable from a model based on abstraction and sensitivity to subpattern frequencies. It remains for future work to assess how the processes implicated by this simulation work can be accommodated in other computational paradigms and can account for similar strategy manipulations on other kinds of tasks.

On Simulation as an Empirical Method
Simulation is used in cognitive science as a means to a variety of different ends: to understand a complex cognitive task, to operationalize a particular model of some phenomenon, to assist in generating empirically testable predictions of a theory. We used simulation in a slightly different way, namely, as a generator of alternative process models that we tested against the observed data. Although we began with a set of predispositions concerning the learning mechanisms in a general framework, we did not begin models of passive or active strategies within that framework whose validity we wished to demonstrate. We did have descriptive notions associated with these two strategies, but approached the effort as an exploration of a model space in which these notions mapped onto different computational mechanisms.

There are several challenges to using simulation in this manner and their common theme is the conceptual leap between descriptive or functional accounts of some behavior and the instantiation of those accounts as particular computational processes. For example, the descriptive accounts conjectured to underlie the phenomena can be "wrong," so their realization as computational processes will not yield the desired effects. The descriptive accounts can be "right" but their translation into some particular computational implementation may not yield the desired behavior because subtle implementation choices mitigate the impact. A descriptive account may have more than one possible computational instantiation and unless these alternative renditions are identified and yield testable predictions, one cannot chose among the alternatives.

An additional challenge in simulating behavior on a complex task is the need to make a large number of decisions about "supporting" mechanisms, for example, short-term memory, similarity judgments. There are some guid-
ing principles (an obvious example is that memory is not infinite and perfect) but it is unlikely that there is any direct data about how these mechanisms operate within the context of the task of interest, or how they are affected by the general features of the task. In this work, we explored alternative implementations of supporting mechanisms (e.g., working-memory processes, alternative generalization and discrimination designs) and it is a time-consuming enterprise. Throughout an exercise like this, one constantly questions the degree to which the computational realization or implementation detail of any mechanism unintentionally influences the behavior of interest. A simulation offers an idea about some general principles that underlie the phenomenon being addressed. If those principles or the functional account are correct, then the same effects should be realized when the principles are translated into some other computational paradigm.

How can we better identify what the general principles are? The more complex the underlying computational model and theory, the more difficult it becomes to assign credit to any single aspect in simulating some data set or to anticipate how a set of independent mechanisms will interact to produce some behavior. Any given simulation model is just one model in a very large space of models. This may be why any particular simulation model of a task or phenomena can be greeted with unease or disinterest by various segments of the cognitive science community: It is like a single experiment in which a large set of factors yielded some result. For any successful simulation model of some phenomenon, there are likely to be other "nearby" models in the space that also account for the data. It is important to see what that set of models looks like. Similarly, there will be nearby variations of the successful model that do not account for the data. What does that set of models look like? Exploring the space around one successful simulation model is important because it elucidates what the critical factors are in a complex world. In other words, it helps with the credit assignment problem and a functional analysis of where the action is in simulating some behavior.

The parameters we discussed in this article are only a subset of the ones we explored. For example, we considered that different strategies might translate into different uses of error-trial information. This led us to explore alternative discrimination mechanisms that differed in the kind of category information that was extracted from incorrect classifications. In the end, this level of mechanism did not matter for the behavior and task demands that we were interested in simulating. Instead, the action behind simulating an interaction between strategy and task demands occurred at a more coarse level: how quickly category-level regularities were detected and available for use, and how those regularities were applied to make decisions. This is not necessarily the kind of outcome one can anticipate in general, and it becomes clear only through an exploration of a model space. If this account is correct, then the corresponding coarse adjustments across qualitatively different
SIMULATING LEARNING MODE AND VARIANCE EFFECTS

Simulation frameworks should also yield the same effects. And, perhaps, the finding that strategy effects can be simulated with coarse adjustments within any framework is the interesting insight about the cognitive machinery and our current understanding of strategies.

The points we have raised in this section are not altogether novel and have occurred in many forms over the years as the merit and methodology associated with simulation have been considered. We can wonder why this kind of experimental approach to simulation is not followed more as a matter of course. The problem may be twofold. First, it has been our experience that the space of possible programs that fail to model a complex cognitive task is extremely large, even with no regard for psychological plausibility, so achieving a successful simulation model is harder than it might otherwise appear. Nonetheless, the failures are informative when considered along with the success and should be part of the overall analysis. Second, unless one has the intention of exploring a space of models, explicit and implicit assumptions that may be critical to the behavior are often hardwired into the simulation software. It becomes less obvious and certainly less inviting to manipulate them as model parameters. If one goes into a simulation effort with the mind-set of exploring a space of models, then the software design to facilitate this exploration will naturally follow. Indeed, a critical eye of any process model can come up with a distressingly large number of design and implementation decisions, any and all of which could be parameters for defining alternative models that might behave in fundamentally different or equivalent ways.

Limitations of the Models and Framework

There are several occasions in the general framework in which some explicit pattern selection occurs: Classification decisions are based on a single best pattern of those meeting retrieval thresholds; only that selected pattern has its strength modified or participates in the creation of new patterns; only one discrimination pattern is randomly selected from the set of possibilities. Although this is similar to other models of this sort of task (e.g., Hintzman & Ludlum, 1980), it is counter to the flavor of connectionist paradigms and other induction frameworks (e.g., Hintzman, 1986) in which there is a collective influence of patterns activated by a stimulus on decisions made about the stimulus. A logical concern about the net effect of these selections is that the observed simulation effects may be an artifact of some feature of these patterns. In addition, the learning effects on each trial are localized on a single, possibly idiosyncratic, pattern representing a single point in the search space. These possible selection effects would be more of a concern if the simulation results were based on only a single run per condition, rather than the average of 15 runs, each using different randomly generated training samples.
The last point that we wish to make concerns the locus of strategy effects, representation, and how these relate to the approach we have taken in this work. One view of strategy effects, put forth by other researchers as well (e.g., Medin, Dewey, & Murphy, 1983), argues that strategies induced by instructions alter the representation but not the fundamental nature of underlying inductive processes. A representation is defined by structures and the processes operating on those structures. We did not tag patterns in long-term memory as "rules" or "nonrules" in one manner for passive learning models and in another way for active learning models. For the most part, we subjected the same structures (e.g., abstracted feature patterns, instance information) to different processes (e.g., retrieval criteria, matching criteria) and in that manner defined different representations. Subjecting the feature-pattern structures to immediate availability full-match processes defined a representation that yielded results associated with a deliberate rule-based, hypothesis-testing mode. A delayed-availability partial match process coupled with the same structures defined a representation that gave results more akin to a passive, perhaps "analogical" use of category-level information. However, it is not clear how this approach would account for other recognition or recall results associated with strategy differences. For example, rule-testing learners have good memory for, and can report on, hypotheses they have tested, but they have poor memory for the exemplars actually studied (Reitman & Bower, 1973). Furthermore, this framework does not address the kinds of features about the task or learning experience that would compel a learner to switch strategies spontaneously.

Final Remarks
We simulated the effects of different instructional sets and their interaction with variance information with a common set of processes that differed only in the constraints operating on them. The impact of different processing models need not be modeled as a different set of processes that produce qualitatively different kinds of information. At least for this kind of learning task, we found that qualitatively different representations can emerge within a common framework through the subtle interactions of a small set of parameter changes. The results presented here also provide models of exemplar order effects, independent of the learning-strategy distinction. This is important in its own right. Although exemplar order effects are often viewed as an undesirable consequence for most learning systems, we prefer to believe that being affected by information order ultimately works as a feature, rather than a bug, of the cognitive architecture. We hope this work prompts continued research interest in developing learning frameworks that model this information order effect and in understanding the computational accounts of learning strategies.
REFERENCES


