Syntactic categorization in early language acquisition: formalizing the role of distributional analysis

Timothy A. Cartwright, Michael R. Brent*

Department of Cognitive Science, Johns Hopkins University, 3400 North Charles Street, Baltimore, MD 21218, USA

Received 15 July 1996, final version 5 December 1996

Abstract

We propose an explicit, incremental strategy by which children could group words with similar syntactic privileges into discrete, unlabeled categories. This strategy, which can discover lexical ambiguity, is based in part on a generalization of the idea of sentential minimal pairs. As a result, it makes minimal assumptions about the availability of syntactic knowledge at the onset of categorization. Although the proposed strategy is distributional, it can make use of categorization cues from other domains, including semantics and phonology. Computer simulations show that this strategy is effective at categorizing words in both artificial-language samples and transcripts of naturally-occurring, child-directed speech. Further, the simulations show that the proposed strategy performs even better when supplied with semantic information about concrete nouns. Implications for theories of categorization are discussed.

1. The role of distributional analysis in grammatical category acquisition

As a part of acquiring a language, children must learn the grammatical categories of individual words. This is difficult, because the same phonological word can be assigned to different categories across languages and even within one language. For instance, the word /si/ is a verb (see) or noun (sea) in English, and a conjunction (si, ‘if’) or adverb (si, ‘so’) in French. In this paper, we propose a novel theory that consists of a strategy by which young children could exploit distributional information to categorize words. Further, we present a series of computer simulations demonstrating that this strategy is effective. The theory is motivated by formal principles of statistical inference and stated formally, but its qualitative properties are clear and easy to understand.

* Corresponding author. email: brent@jhu.edu.
Previous research has focused on discovering sources of information that children could exploit to categorize words, on weighing the relative importance of each source, and on developing learning processes that exploit these information sources. One such information source is distributional regularity, which refers to the fact that words occurring in similar contexts tend to belong to the same grammatical category. A second source is the correlation between the meanings of words and their grammatical categories; for example, words referring to concrete objects are almost always nouns. The phonological forms of words are also correlated with syntactic category; for instance, bisyllabic words with final stress are more likely to be verbs than those with initial stress (as in record, noun, and record, verb). Finally, the analysis of distributional regularities might be more effective if the environments for those analyses were syntactic phrases smaller than complete sentences. For this reason, children might benefit from using whatever syntactic knowledge they have to try to find phrase boundaries; furthermore, using sentential prosody, such as pitch contours, may provide some information about the locations of major syntactic boundaries (e.g., Fisher and Tokura, 1996; Gleitman and Wanner, 1982; Jusczyk et al., 1992).

In this paper, we propose a theory about the learning mechanism by which children exploit distributional regularity. This theory improves upon previous distribution-based proposals, because (a) it makes few assumptions about the availability of syntactic knowledge, yet is compatible with modern theories of syntax; (b) it assumes sentences are processed one at a time and are forgotten after processing; (c) it results in a discrete categorization of input tokens; (d) it allows word types to be put in more than one category; (e) it can exploit other sources of information pertaining to categorization, such as semantics; and (f) it combines all these properties in a detailed, explicit learning strategy.

In the remainder of this introduction, we review evidence for the importance of distributional, semantic, and phonological information in categorization, then describe how the proposed strategy exploits distributional information in a novel way. Experiments 1 and 2 demonstrate that computer simulations of the strategy are very successful at learning the categories implicit in samples generated from artificial languages that are defined exclusively in distributional terms. Experiments 3 and 4 show that the same simulation program can learn grammatical categories from transcriptions of naturally-occurring, child-directed speech. Finally, in Experiment 5 we present one way in which semantic information could be exploited within our theoretical framework, and demonstrate that the simulation program of Experiments 1–4 benefits from the addition of this semantic information. In the General Discussion, we relate the quantitative evidence of the experiments to qualitative properties of the simulations, discuss the theoretical implications of the results, and suggest directions for future work.

2. Theories of category acquisition

In this section, we review the development of category acquisition theories
based on the use of distributional regularities, semantics, and phonology, as well
as computer simulations of category acquisition.

2.1. Cognitive theories

2.1.1. Distributional cues

The idea that children might form grammatical categories by grouping together
those words that occur in the same environments goes back at least to Bloomfield
(Bloomfield, 1933, p. 276). Harris’s procedures for linguistic analysis, which he
termed distributional analysis, included a similar notion of category discovery
(Harris, 1951, 1954). Harris defined the distribution of a word as ‘the sum of all
its environments’ (Harris, 1954, p. 146) and defined a word’s environment as its
position relative to other words in all utterances in which it occurred. To simplify
the description of environments, classes of words could stand in place of
individual words. Thus, utterances were thought to be formed ‘by choosing
members of those classes that regularly occur together and in the order in which
these classes occur’ (Harris, 1954, p. 146). This last idea, including its probabilis-
tic nature, is directly reflected in the learning strategy proposed in this paper; we
call the sequence of classes describing an utterance its template.

Harris did not publish a full analysis of English grammatical categories based on
his procedures. Following similar procedures, though, Fries (1952) constructed
simple utterance templates to identify word categories. For instance, any single
word that could grammatically complete the template ‘The ____ is good’ was part
of his Class A (i.e., was a noun). This template is not sufficient to find all nouns
(e.g., cats), so the template was generalized to ‘(The) ____ is/was/are/were
good.’ (meaning The was optional and several choices of verb were permitted). By
creating and generalizing a set of similar templates, Fries identified a total of 19
word classes.

Continuing this line of thinking, Maratsos (Maratsos, 1988; Maratsos and
Chalkley, 1980) outlined a distributional category acquisition procedure, which
focused on identifying correlations among the morphological environments in
which words occur. For instance, a large group of words (containing some of the
words that we would call verbs) occurs before the past tense morpheme -ed, before
the present tense -s, and after various forms of do; on the basis of this evidence,
Maratsos suggested, children could group together these words. However, this
theory is problematic in that it ignores the question of determining which
morphemes are relevant to the analysis (Gleitman and Wanner, 1982; Pinker,
1979, 1984). English-learning children could easily learn that walk, look, and
believe belong in the same category, but they would have to know to look for
words following do and preceding -s, -ed, and -ing. Furthermore, words like put,
sleep, and think enter into more complex morphological patterns that must be
identified (see Brent, 1993, 1994, for one approach). However, using neighboring
morphemes in this way requires knowing that do, -s, -ed, and -ing are special, that

---

1 We thank Robin Clark for pointing out Fries’s work.
they form a category of syntactically important morphemes. In other words, to learn categories, children would first have to categorize the input. If they don’t, then they would have to conduct their tallies and correlations over every possible environment.

Bloomfield, Harris, Fries, and Maratsos sharpened intuitions about the importance of distribution for categorization, but none of them developed a fully explicit, automatic, and computationally practical process by which distribution could be exploited. Nevertheless, these researchers contributed three important ideas. First, lexical categories must be defined in terms of their distribution. Second, distribution is defined with respect to some structured environment. Although linguists have refined the structural descriptions of sentences, it is still true that a categorization strategy based on distributional analysis depends on the analysis of structured environments (e.g., Elman, 1990; Pinker, 1984). In this paper, we show that analyzing even minimally structured environments is useful for early category acquisition. Finally, Harris and Fries observed that generalizing environments from sequences of words to sequences of classes (templates) yields more compact and understandable descriptions of sentences. We use templates in our learning strategy for an additional reason: Generalization is necessary because no language learner ever hears enough minimal pairs of sentences to learn a complete categorization of words.

2.1.2. Semantic cues
In grade school we were taught that a noun is person, place, thing, or idea, that a verb is an action, and so on; that is, grammatical categories were defined semantically. If this were true, category acquisition would be equivalent to learning word meanings. However, grammatical categories cannot be defined in purely semantic terms; they must be defined structurally, based on their use in a grammar (e.g., Chomsky, 1965; Fries, 1952; Maratsos, 1988; Pinker, 1984). Nevertheless, the correlation between grammatical categories and semantic classes is both strong and universal (Bates and MacWhinney, 1982; Schlesinger, 1988), and it is even stronger in children’s vocabularies than adults’ vocabularies (Brown, 1957). Perhaps children use this correlation to begin learning syntactic categories by grouping together words that belong to the same semantic category. For example, children could infer that since cat, dog, and toy refer to concrete objects, they all belong together in the same (as yet un-named) category. According to this story, purely structural definitions could later be generalized from these initially semantic proto-categories. For instance, the category containing cat would ultimately be generalized to include words that refer to abstract ideas, like belief. We explore the usefulness of exploiting semantic information in this way in Experiment 5.

Another possible use of the correlation between semantic and grammatical categories was developed by Grimshaw (1981) and Macnamara (1982); Pinker (Pinker, 1984, 1987) refined this idea as part of his semantic bootstrapping hypothesis. Under the assumption that Universal Grammar provides a set of grammatical categories, such as noun and verb, it remains problematic to link
groups of distributionally grouped words to their proper categories and hence to
their roles in the grammar. Grimshaw, Macnamara, and Pinker suggested that
semantics should be used to solve this linking problem. For example, the fact that
a word refers to a concrete object can be used to label it as a noun. After that,
words can be categorized either by distributional means or by further application
of semantic correlations, although the distributional analysis always has the last
say. Hence, truly grammatical categories are learned, but the process is begun with
help from semantics. The research reported here does not directly explore this
possibility, but is compatible with it; see the General Discussion.

2.1.3. Phonological cues

Although there is no universal mapping from phonological word forms to their
syntactic categories, there are language-specific correlations between certain
phonological properties of words and syntactic categories; children may be able to
exploit those correlations at some stage of category acquisition.

Kelly (Kelly, 1992, 1996; Cassidy and Kelly, 1991) has shown that numerous
phonological properties of English words (especially nouns and verbs) correlate
with their syntactic categories. For example, bisyllabic nouns tend to have
first-syllable stress, whereas bisyllabic verbs tend to have second-syllable stress;
also, verbs tend to have fewer syllables than nouns (at least in child directed
speech). There are some languages that even fully distinguish among syntactic
categories with phonological properties—Kelly (Kelly, 1992, p. 355) cites Hebrew
and Russian as examples. Furthermore, Kelly and others have shown that adults
and, for some phonological properties, even children are sensitive to these
correlations (e.g., Cassidy and Kelly, 1991; Kelly and Bock, 1988; Sorenson et al.,
1978).

In a different vein, Morgan et al. (1996) claimed that function words—for
example, determiners and prepositions—and content words—nouns, verbs, adjectives,
and adverbs—can be distinguished in any language based on phonological
properties. In general, they say that function words tend to be minimal relative to
content words; however, the definition of minimal changes from one language to
another. For instance, function words may have fewer syllables, may be composed
of lighter syllables, or may have higher incidence of underspecified phonemes in
their underlying forms than content words. Typically, no single property reliably
predicts syntactic class, but rather, the combination of many properties considered
together does so. For example, in a sample of English child-directed speech, vowel
duration separated function and content words with at most 65% accuracy;
however, a multivariate analysis including several phonological properties sepa-
rated words with 83–91% accuracy, depending on the speaker (p. 247).

Both kinds of phonological correlates to syntactic category are language-
particular; thus, children would have to learn the particular patterns of correlation
for their language before being able to exploit them. No one has directly shown
that children exploit phonological cues for categorization, but Kelly (Kelly, 1992,
1996) and Morgan et al. (1996) argue that children’s documented sensitivities to
other phonological properties (for reviews, see Jusczyk, 1993; Jusczyk and Kemler Nelson, 1996; Mehler et al., 1988) warrant further exploration of this hypothesis.

Sensitivity to phonology is only the beginning of a full story. To demonstrate that the correlations between phonology and syntactic categories could be useful in categorization, Morgan et al. (1996) trained a Kohonen network—a type of neural network that clusters input data without being given the correct categorizations for the data—to cluster words based on their phonological properties. The network was very good at clustering novel function and content words with other members of their classes. This is a useful result, but these clusters are much broader than typical grammatical categories, hence some other process is required to further subdivide these clusters into more traditional grammatical categories.

Thus, language-independent correlations between phonological forms of words and their syntactic categories seem to provide limited information to category acquisition. Nevertheless, phonological cues may ultimately contribute to the acquisition of syntactic categories after other learning processes—such as the one proposed in this paper—provide training material for acquiring language-particular correlations. Combining the suggestions of Kelly (Kelly, 1992, 1996) and Morgan et al. (1996), phonology may at least help distinguish among nouns, verbs, and function words.

2.1.4. Summary of cognitive theories

Despite the potential importance of semantic and phonological cues, everyone seems to agree that grammatical categories must ultimately be defined in grammatical (i.e., structural) terms. However, agreement on that point does not solve the following problem: Categories are defined in terms of the syntactic environments in which they can occur, but those syntactic environments are defined in terms of allowable sequences of categories. Thus, structural definitions are circular in nature (Maratsos, 1988; Pinker, 1984), and children must have some way of breaking into the system. Many researchers agree that, once started on the correct path, some not-yet-understood distributional analysis will be able to finish the job—perhaps working with semantic and phonological cues. A central goal of this paper is to make less mysterious the distributional component of such a theory.

2.2. Computational models

Later in this paper, we investigate our theory of category acquisition using a series of computer simulations. These simulations differ substantially from others reported in the literature, so this section briefly reviews those other simulations and argues for the necessity of a new approach.

Before the early 1970s, researchers did not distinguish between the problem of learning words’ categories and the problem of learning syntax, so computational models of category acquisition were merely parts of computational models of grammar acquisition. Some models (e.g., Siklóssy, 1971) were ad hoc implementations of the idea of distributional analysis and could not reasonably serve
as foundations for cognitive theories; others (e.g., Feldman, 1972; Horning, 1969) were based on techniques in probability and statistics that were used to address the issues of formal learnability raised by Gold (1967). See Pinker (1979) for a thorough review of these and other, similar approaches.

2.2.1. Hierarchical cluster analysis

In the early 1970s, Kiss (1973) started a new direction in the computational modeling of category acquisition by using a form of hierarchical cluster analysis to group together words whose distributional patterns were similar. In this subsection, we describe hierarchical cluster analysis, discuss the differences between it and our approach, then summarize how it has been applied to category acquisition (Elman, 1990, 1991, 1993; Mintz et al., 1995; Redington et al., 1995).

First, we review the processing involved in a hierarchical cluster analysis. Before applying cluster analysis, the contexts in which words have occurred in some language sample are represented by vectors of numbers. A distance measure, computed between individual vectors or groups of vectors, is established to reflect the similarity of words’ vectors. The cluster analysis uses the vectors and the distance measure to form a large number of nested word categories. Starting with each word in its own category, a hierarchical cluster analysis proceeds iteratively by finding the two categories separated by the least distance and joining them together to form a new category. This process continues until only one cluster remains, containing all the original input words. The result is a hierarchy of categories with the original input words at the bottom and increasingly larger categories as higher levels; if there are \( n \) words, then \( 2^n - 1 \) categories are formed.

To use hierarchical cluster analysis for learning categories, one must define three parameters: A list of words to be analyzed (the target words), the process by which vectors are derived from the contexts of target words, and a distance measure. Generally, the vector for a target word is derived from its co-occurrence patterns with certain other words known as context words. The context words are typically those meeting some minimum frequency criterion. There are numerous distance measures in common use (see, e.g., Sokal and Sneath, 1963), one of which is selected for the particular application.

There are two major differences between hierarchical cluster analyses and our proposed learning strategy. First, our strategy results in a set of discrete categories of words, whereas hierarchical cluster analysis results in a large number of nested categories. It is not the nestedness of the categories per se that is problematic, but rather the large number of categories. The ultimate function of distributional categorization is to reduce the search space for later syntactic generalizations. For example, a good categorization should have the property that if one word in a category can serve as the subject of a sentence—that is, it is a noun phrase—then the other words in that category can serve as subjects, too. But in a standard hierarchical cluster analysis, each word belongs to a number of categories, including one that contains no other words and one that contains all other words. Such a categorization leaves open the question of which level to use for generalizing a given observation. It is possible to obtain a set of discrete categories
from the nested categories by halting the cluster analysis at some arbitrary similarity threshold, but the cluster analysis process itself cannot be used to inform a learner of which threshold to use. To date, no one has proposed a method—relying only on input data for using hierarchical cluster analysis to obtain discrete categories.2

Another difference is that our learning strategy is incremental: It operates on one sentence at a time, forgetting previous sentences. On the other hand, the standard implementations of hierarchical cluster analysis—including those described below—process all of their language input in one batch. To make an incremental version of cluster analysis, the set of vectors would have to be updated after each input sentence, then the entire clustering of words would have to be re-computed; that is, there is no obvious way to incrementally update the clustering given an incremental change to the underlying set of vectors. And cluster analysis is a time-consuming process: The processing time for each input sentence would be proportional to the square of the number of word types seen so far. It is unclear how a cognitively-plausible acquisition theory could be built around this method. Nevertheless, studies using cluster analysis can provide researchers important information about the usefulness of analyzing local word contexts for forming categories.

2.2.2. Computational studies using hierarchical cluster analysis

Many of the researchers using hierarchical cluster analysis defined distributional contexts by a set of word positions immediately adjacent to target words; Kiss (1973) used one position after targets,3 Mintz et al. (1995) used one position before and after targets, and Redington et al. (1995) used two positions before and after targets. For each target word, lists were made of the words occurring in each context position and their frequencies; the resulting lists of frequencies defined the word’s vector. To reduce the number of word/position elements in each vector, Kiss eliminated context words whose relative frequencies were less than .01, and Mintz et al. and Redington et al. restricted the number of context words tracked in each position to the most common words in their corpora (200 and 150 words, respectively). Before submitting them to cluster analyses, Kiss and Redington et al. normalized the frequencies in each target word’s vector to relative frequencies.

Kiss analyzed the 31 most frequent words from his 15,000 word sample of child-directed speech; Mintz et al. analyzed the 200 most frequent words from each of their 3 child-directed speech samples (of roughly 6,000 sentences each); and Redington et al. analyzed the 1,511 most frequent words from a 2.5 million word sample of mostly child-directed speech (all words occurring at least 100 times were used). In each case, researchers reported parts of the resulting hierarchy or some of the discrete categorizations that could have been obtained

2 Although Mintz, Newport, and Bever are working on a solution to this problem (T. Mintz, personal communication, May, 1996).
3 Kiss used other, non-hierarchical clustering techniques as well. Generally, results were consistent with the hierarchical clustering and are therefore not discussed here.
from the resulting hierarchy; Redington et al. reported quantitative evaluations of all possible discrete categorizations obtained from one hierarchy. Generally, the analyses appeared to recover reasonable clusters of syntactically related words; that is, syntactic relatedness seemed to be correlated with distance in the vector space, and was therefore reflected in the clustering. Mintz et al. further demonstrated that their results were robust even when function words were given limited or no role in determining target words’ contexts.4

These programs used very local definitions of context—no more than two words to either side of target words—yet were able to cluster words. However, positionally-defined contexts can violate syntactic phrase boundaries. For example, suppose the context for a target word is defined to be its immediate successor. Then, the context for cat in the sentence the cat meowed is ‘___ meowed’. However, cat and meowed do not form a phrasal unit. More precisely, in the structural description of the cat meowed, the target word cat and its context word meowed are not dominated by a phrasal node that dominates no other words. The appropriate domain for syntactic analysis is a complete phrase, and thus approaches that ignore phrasal boundaries may add substantial noise to their analyses.5 We avoid this problem by using contextual domains—complete sentences—that do not violate phrase boundaries.

Elman (1990) used a different approach in computing vectors for words. His input consisted of 10,000 two- and three-word sentences generated from a simple, finite artificial language; there were 13 word classes and 29 word types, so each word type was expected to occur about 950 times in the input sample. A simple recurrent neural network was trained to predict the next word of the input. After 6 complete training passes through the input, learning was turned off and the inputs were once again presented to the network. On this final pass, the activation on the network’s 150 hidden units caused by each word was recorded as the vector characterizing the context in which the word occurred. Two types of hierarchical cluster analyses and other analyses of the neural network (Elman, 1991, 1993) show that the network learned something about the underlying word classes.

To summarize, distribution-based computer models of word categorization have used hierarchical cluster analysis. Contexts were small—anywhere from one successor to two immediately adjacent positions6—and were not defined structurally. Despite these restricted analysis contexts, all the studies described above achieved reasonable, intuitive clusterings of words.

Except for Kiss (1973), no one has proposed a theory of category acquisition based on or simulated by a computer model; Mintz et al. (1995), Redington et al.

4 In related work, Brill (Brill, 1991; Brill et al., 1990) obtained discrete categories by clustering together those words whose contexts reached an arbitrarily-set threshold of similarity.
5 This is not to say that computer programs using distributional contexts that violate phrasal boundaries are completely ineffective, only that respecting phrasal boundaries might render them more effective. Mintz, Newport, and Bever are working on a procedure that uses hierarchical cluster analysis but does not violate phrase boundaries (T. Mintz, personal communication, May, 1996).
6 It’s not clear how large contexts were in Elman’s (Elman, 1990, 1991, 1993) work. He (Elman, 1991) presents evidence that simple recurrent networks can track fairly long-distance relationships.
(1995), and Elman (1990) explicitly state that they were not proposing acquisition theories, but were exploring the usefulness of the information sources. Given the limitations of hierarchical cluster analysis discussed earlier, it seems unlikely that this technique will be the foundation of a theory of grammatical category acquisition in children. Nevertheless, these studies represented a step forward. Until these computer models were developed and run, proponents of distributional analysis assumed with little or no empirical evidence that distributional information could inform a procedure for learning grammatical categories. In the remainder of this paper, we build on these results by presenting a completely new theory of early grammatical categorization and a computer simulation of it.

3. Reconstructing distributional categorization

When hearing a sentence, children must determine how to put the words of the sentence into structurally-defined groups. Assuming that they do not know how many categories their language uses, they might have to create a new group for a word, or merge together two existing groups based on the new evidence the sentence provides. Further complicating the task are words that are categorically ambiguous; *brush*, for example, can be used as a noun or verb. Thus, children cannot assume that a new instance of a word should be put in the same group as earlier instances of the same word.

We motivate our proposed solution to this problem by appealing to an intuitive form of distributional analysis, then describe in detail the solution itself and review its formal underpinnings (Appendix A provides a complete description of the underlying formalism). We conclude the introduction with several examples that show how the theory’s components interact in response to different input conditions.

3.1. Intuitive distributional analysis

The following pair of sentences constitutes a minimal pair, in that they differ in word choice at exactly one position.

I saw the cat.
I saw the dog.

On the basis of these two sentences, it seems natural to conclude that *cat* and *dog* belong to the same grammatical category. To be confident in this conclusion, one would like to see other pairs of sentences that differ only by the choice of *cat* or *dog* in some position, but even one example is strongly suggestive.

It is tempting to construct a theory of category acquisition around the exploitation of sentential minimal pairs. However, very few minimal pairs of nearby sentences occur in natural speech, and hence there is too little data to completely acquire a set of grammatical categories in this way. Nevertheless, the
minimal-pairs idea can be generalized in a way that makes it useful for category acquisition. Consider the following sentences, which have no words in common, and therefore do not form a minimal pair.

My cat meowed.
Your dog slept.

Suppose that my and your are thought to be in the same category (labeled Det, for *determiner*), and that cat and dog are thought to be in a different category (N, for *noun*). Further suppose that meowed and slept, which are novel words, are in the temporary categories X and Y. These sentences can now be represented in terms of templates—sequences of categories—instead of sequences of individual words. The templates for the two sentences are

Det N X
Det N Y

It is easy to see that these templates form a minimal pair, and thus there is evidence for putting meowed (category X) and slept (category Y) together into one category. In this sense, minimal pairs of templates can be viewed as *generalized minimal pairs*, because sentential contexts are generalized by replacing specific words with the categories to which they belong.

A theory of category acquisition based on generalized minimal pairs seems more reasonable than one based on minimal pairs per se. A few true minimal pairs are needed to start the categorization process. This requirement seems acceptable: In a casual inspection of spontaneous child-directed speech, we found several instances of nearby minimal pairs (see also Snow, 1977, and papers cited therein). After a few words have been categorized, the task of finding more generalized minimal pairs becomes easier.

Basing a theory on generalized minimal pairs still leaves some issues unresolved. The importance of sentence length needs to be addressed; it is clear that a generalized minimal pair of, say, seven-word sentences is stronger evidence for categorizing together the distinguishing words than a generalized minimal pair of two-word sentences. Also, there must be some mechanism by which words can be put into more than one category. This, entails balancing the preference to put all instances of a word type in the same category against evidence suggesting that some instances of that type belong in a different category.

In this paper, we propose a categorization strategy that resolves these issues while retaining the promising aspects of generalized minimal pairs. This proposal describes how generalized minimal pairs (via templates) can be used to group words having similar syntactic privileges based on distributional information. For now, this proposal does not address the issue of how groups are labeled with category names (such as *noun* and *verb*). To avoid confusion, we refer to the intended output of the learning strategy as *groups*; we reserve the term *categories* for groups that have been labeled with some grammatically meaningful category, such as *noun*, *count noun*, or *ditransitive verb*. 
3.2. From intuitions to theory

We propose that children analyze the sentences they hear in two steps: First, each word from an input sentence is assigned to a new group that has no other members, then pairs of groups may be merged together based on the evidence provided by the current sentence. Categorization is thus recast as deciding which, if any, groups should be merged given an input sentence. Children need not consider merging every possible pair of groups, because there is no motivation to merge two groups when neither one is part of the current input sentence. Furthermore, if no single merge seems useful in the current situation, the child should not randomly merge groups in hopes of finding a combination of merges that is useful. In fact, because we assume that it is difficult—perhaps impossible—for children to undo a merge, it is always better to be conservative in the merging process; more precisely, it is better to postpone a merge until there is more evidence for it, than it is to prematurely (and possibly erroneously) merge a pair of groups. But how is the child to decide when a particular merge is useful?

We propose a theory in which children follow a particular set of preferences in deciding which, if any, groups to merge. These preferences cannot be defined solely in terms of the arrangement of words into groups; in order to perform a distributional analysis, children must track contexts in which words occur, and should prefer to group together words that occur in similar contexts. Before listing the preferences themselves, we discuss our method for tracking contexts.

3.2.1. Contexts

In our theory, contexts are tracked by a list of sentence templates. Recall that a template is a sequence of group tags; it stands for the set of sentences whose words have the same sequence of groups. For instance, if the sentence ‘Arnold saw the dog’ has the template ABCA (where each letter is a group tag), then the sentence ‘Helen fed the cat’ should have the same template. The sentences ‘Sue wept’ and ‘My old car died’ each have different templates from the one given above; The first has a different length and the second has different group sequence.

The templates used to track contexts could be interpreted as a simplistic grammar. However, the focus of our theory is on learning word groups and not on learning a grammar. We view the grammar-like component of our theory as just a tally of the contexts in which words and groups occur; we do not intend it as a substitute for a modern generative grammar, such as Government and Binding Theory (e.g., Haegeman, 1991), Lexical Functional Grammar (e.g., Kaplan and Bresnan, 1982), or Optimality Syntax (Barbosa et al., in press; Prince and Smolensky, 1993).

Nevertheless, templates are related to full syntactic trees in the following way. The bottom level of a syntactic tree is a sequence of words; each word is dominated by its category, and each category symbol is dominated by a hierarchy of phrase markers culminating in the phrase marker for the entire sentence. If the words and all the phrase markers above the category symbols are removed, what remains resembles a template. The only difference is that the category symbols in
a full syntactic tree have a certain meaning in virtue of the role they play in the rules or constraints of the rest of the grammar. In our system of templates, there is no grammar besides the templates themselves, and hence group symbols have no meaning independent of the set of templates in which they occur. Thus, templates are a minimal structure for tracking words’ contexts, but they are compatible reductions of the information contained in full syntactic trees.

According to our theory, the templates needed to describe all sentences heard so far are kept in memory. One can imagine more linguistically sophisticated ways of tracking contexts, but templates offer a good balance between grouping performance and computational complexity (see General Discussion).

3.2.2. Preferences

To summarize, our proposal about the preferences children use in deciding which groups to merge defines these preferences in terms of the arrangement of words into groups and the arrangement of groups in templates. We present the preferences related to the arrangement of groups in templates first, followed by the preferences related to the arrangement of words into groups. After listing the preferences, we describe the formal analysis that inspired them.

One final note: Different preferences may cause the child to prefer conflicting merges. For instance, one preference is for the total number of groups to be as small as possible, and another is for each group to contain as few members as possible. Given any pair of groups, the preference for a small number of groups suggests that the two groups should be merged, while the preference for small groups suggests that the two groups should not be merged. A method to balance conflicting preferences is described after the preferences.

1. Minimize the number of templates. This is the primary contextual preference and most directly implements the idea of reducing generalized minimal pairs. Consider the case of two sentences, one whose template is ABCD and one whose template is ABCE (where the capital letters denote arbitrary group tags). Suppose groups D and E are merged, and their union is called group Z; replacing all Ds and Es in the template list results in both of the above templates being re-written as ABCZ. Whereas there were two templates before the merge, now there is only one; the number of templates is reduced, in accordance with this preference. Because only one pair of groups can be merged at a time, minimizing the number of templates is equivalent to finding pairs of templates that merge as the result of a single group merge. Therefore, this preference drives the search for and reduction of generalized minimal pairs (i.e., minimal pairs of templates).

2. Minimize the sum of the lengths of the templates. This preference also works to reduce the number of stored templates, because group merges cannot change individual template lengths. Unlike preference #1 however, this preference for merging templates is stronger for long templates than for short ones. For example, consider the templates AB, AC, PQRSTUV, and PQRSTUW. Merging groups B and C leads to merging the first two templates, resulting in one less template of length 2; merging groups V and W leads to merging the last two templates, resulting in one less template of length 7. As discussed earlier, there is stronger
evidence for merging groups V and W than for merging groups B and C, because the longer minimal pair is less likely to have occurred by chance. Thus, all else being equal, we predict that children would prefer the V-W merge. This preference addresses one of the shortcomings of the intuitive notion of generalized minimal pairs—its inability to make distinctions among generalized minimal pairs of different lengths.

3. Create templates with the highest possible frequency. Commonly heard contexts are better sources of distributional evidence than infrequently heard ones, because they are less likely to reflect ungrammatical noise or other chance occurrences. Templates are used to track contexts, and so it is best to merge groups so as to create templates that describe as many sentences as possible, thus pooling evidence from diverse sources.

Next are the preferences related to the arrangement of words into groups.

4. Minimize the total number of groups. Every group merge helps to satisfy this preference, because every merge—regardless of the composition of the groups involved—reduces the total number of groups by one. There must be preferences that lead to a reduction of the number of groups, otherwise there would be no grouping.

5. Put all instances of a word type together. This is another strong preference for merging, although it is more selective than the previous one. Clearly, instances of a word type belong together, unless there is convincing evidence to the contrary.

6. Minimize the number of types whose instances are divided among different groups. All other things being equal, it is better to have one type that belongs to several groups than it is to have several types that each belong to two groups. This is consistent with the intuition that languages have some unambiguous words; thus, there must be some evidence for putting a word into more than one group.

7. Minimize the number of words in each group. This preference and the next work against merges. Preferences of this sort are needed, or else nothing prevents the learner from putting all words into one large and useless group. We hypothesize that children merge groups conservatively; hence, merges must be justified.

8. Minimize the number of groups consisting of more than one type. Some merges are more clear-cut than others. Generally, it is obvious that instances of a word type belong together, especially when the word type is not yet grouped with any others. However, much more positive evidence is needed when grouping together different word types.

9. Maximize the frequency of word types within their groups. When a word type occurs in more than one group, this preference says it is best to use the group in which the type has the greatest frequency. That is, all else being equal, it is best to go with the more strongly established grouping of the type. This is important: If only a small number of tokens occur in another group, they may be there because of noise or error, so the group should not be used unless there is strong evidence for it.

Finally, there is one preference that bridges between groups and templates.

10. Use large groups (in terms of numbers of types) in the templates. This is
another preference for using common things: It is better to use groups that are firmly established than groups that are small and that may have resulted from noise or error. In some respects, this adds to the drive to merge groups, because merging two groups creates a new, larger group.

While we believe that this list of preferences would help a child make good decisions about which, if any, groups to merge, we have not yet said how these ten conflicting preferences are combined and balanced to yield decisions. Furthermore, we have justified the list of preferences on intuitive, non-rigorous grounds. In the next section, we show how our proposal was motivated by formal principles of learning.

3.3. Categorization as optimization

Our theory was motivated in part by casting the problem of early category acquisition as an optimization problem. A simple example of optimization is linear regression, which is the problem of finding the best straight line to describe a set of points in space. One way to approach this problem is by search and evaluation: Search through the space of possible lines by guessing at slope and intercept values, then evaluate each line by computing the sum of the squared vertical distances between the line and the data points. The best line, the one with the smallest sum-of-squared-errors, is retained as the best fit.

In general, optimization is the attempt to find the element of some domain for which the value of an objective function is minimal (or maximal). In the example above, the objective function was the sum of the squared vertical deviations and the domain was the set of possible slope/intercept pairs. The domain can be non-numeric, as is the case for the categorization problem. Furthermore, the objective function may take into account multiple, conflicting preferences, each weighted differently. In the regression example, each point acts as a separate preference whose strength is proportional to the square of its vertical distance from the proposed line.

The Minimum Description Length (MDL) paradigm (e.g., Brent and Cartwright, 1996; Ellison, in press; Li and Vitányi, 1993; Quinlan and Rivest, 1989; Rissanen, 1989) is an analytic framework in which inductive learning problems are analyzed as optimization problems. This paradigm has already been applied to various language learning problems (e.g., Ellison, in press; Stolcke and Omohundro, 1994), and to modeling other aspects of language acquisition (Brent and Cartwright, 1996). Stated simply, the MDL paradigm says that the optimal description of some data is the shortest one. In this approach, a description of some data is composed of (a) a hypothesis—a generalization of the data in terms of a (pre-determined) class of possible hypotheses, and (b) a derivation—the observed data described in terms of the hypothesis. Thus, the value of an MDL...
objective function is the combined lengths of the hypothesis description and the derivation description. These two parts must be specified in such a way that the original observed input can be completely reconstructed. Thus, they constitute a factoring of the input into a reliable generalization (the hypothesis) and some unexplained or chance residue (the derivation).

Continuing the regression example, a set of points in two dimensions can be described by a polynomial curve (the hypothesis) and the vertical difference between each data point and the curve at that point (the derivation). The simplest such hypothesis is a straight line, described by two parameters (e.g., the slope and intercept). Unless the data points happen to fall on a straight line, though, such a simple hypothesis may yield large errors between hypothesis and data. Assuming that large numbers require longer descriptions than small numbers, then large errors result in a long derivation. Fitting a higher-degree polynomial reduces errors and therefore reduces the size of the derivation, but also increases the size of the description of the hypothesis. Thus, as is usually the case, there is a trade-off between making the hypothesis simpler and making it describe the data more accurately.

We can now recast the categorization problem as an MDL optimization problem (see below for an example). Hypotheses describe the assignment of words to groups and the composition of templates in terms of those groups; derivations describe the sequence of templates seen in the input sentences and which words, within each group, occur at each template position. For example, suppose four sentences have been observed:

- this is a kitty
- this is a doggie
- what a nice kitty
- what a cute doggie

A fairly good description of these sentences would have these groups:

A: {this}  B: {is}  C: {a}
D: {kitty (d1), doggie (d2)}  E: {what}  F: {nice (f1), cute (f2)}

For later reference, each group has been assigned an arbitrary tag, A through F, and in groups containing more than one word type, each word is also assigned a separate tag.

Given this grouping of words, each sentence can be described by its corresponding template:

Chomsky (Chomsky, 1965, pp. 30–47) laid out a program for developing explanatory linguistic theories that is similar to MDL except that hypotheses are evaluated in terms of the length difference between the raw data and the hypothesis, neglecting the length of the derivation. Neglecting the length of the derivation can be interpreted as neglecting what is sometimes called indirect negative evidence—the evidence provided by the absence of certain predicted forms from the input data.
this is a kitty → ABCD
this is a doggie → ABCD
what a nice kitty → ECFD
what a cute doggie → ECFD

Note that only two distinct templates are used, ABCD and ECFD. Thus, the complete template list for the four sentences is

1: ECFD; 2: ABCD

For later reference, each template has been assigned an arbitrary integer label.

These two pieces of information—the list of groups and the list of templates—form the hypothesis. We can see that this is a generalization of the actual input because it could be used to describe sentences not occurring in the input. For instance, the template ECFD could be used to describe the sentence *what a cute kitty*, which did not occur in the input. Thus, the hypothesis alone does not completely describe the input, and so, as required by the MDL paradigm, a derivation describing the input in terms of the hypothesis must also be provided. The first part of the derivation lists the templates corresponding to the input sentences:

2, 2, 1, 1

In other words, template 2 describes the first and second sentences, and template 1 describes the third and fourth sentences. Looking up template 2 in the hypothesis, the sequence of groups in the first and second sentences can be reconstructed: ABCD. But this does not indicate which actual words occurred in the input; given the group definitions in the hypothesis, the first sentence could be *this is a kitty* or *this is a doggie*. So, for each group containing more than one word type, the label of the word within the group must be given, as in

Sentence 1:  \( - - - d1 \)
Sentence 2:  \( - - - d2 \)
Sentence 3:  \( - - f1 d1 \)
Sentence 4:  \( - - f2 d2 \)

(where the dashes are place-holders indicating a template position requiring no further specification). Now each sentence can be completely reconstructed. For example, sentence 1 has the group sequence ABCD; group A has one member, *this*; group B has one member, *is*; group C has one member, *a*; and from group D, which has two members, word \( d1 = \text{kitty} \) actually occurred in the input sentence.

Returning to the definition of MDL, the optimal description of some data is the shortest one. It turns out that we can derive a formula for the length of a description once we have determined the format for a complete description. The
description length formula and its derivation from the description scheme outlined above is given in Appendix A. The formula is stated in terms of the number of groups, the number of groups containing only one word type, the number of word types, the number of groups each word type is in, the frequency of each word/group combination, the number of templates, the length of the longest sentence, the frequencies of group tags in the template list, and the frequencies of the templates in the derivation. The formula gives rise to the preferences described earlier; Appendix A points out the connections between preferences and terms in the formula.

One of the benefits of basing our list of preferences on the MDL paradigm is that the description scheme and thus the description length formula determine the balance between preferences. For instance, the description length formula we used has a single term for the number of groups (which contributes to the preference for fewer groups) and a term for two times the number of templates (which contributes to the preference for fewer templates). We did not select these coefficients; rather, they arose from the process of deriving the description length formula. The formula we used is not the only possible one (we chose the one in Appendix A in part for its computational efficiency), but other variants also give rise to the preferences listed above.

3.4. Learning algorithm

At this point, we can describe the complete learning algorithm. We hypothesize that, upon hearing a sentence, children create a new group for each word, including familiar words. At this point, the sentence is described by a new template consisting of the sequence of new groups. According to our hypothesis, children add this new template to their template lists. They then evaluate, according to preferences described above, the possibility of merging each group in the new template with each of the other groups in their mental lexicon. The merge that results in the best arrangement of groups and templates is done, if the resulting arrangement is better than the current one. Merging continues in this way until no merges can improve the arrangement of groups and templates. Note that an input sentence may result in no merges, if no merges improve upon the initial categorization of the sentence.

3.5. Examples of the algorithm in action

In this section, we present four input samples and discuss the action of the learning algorithm on each of them. For each example, we assume no other sentences have been processed.

3.5.1. Example 1: No action

Consider the processing that would take place after hearing the sentences *my cat meowed* and *your dog slept*; assume no merges were done after the first sentence.
After the second sentence is read in and its words assigned to new groups, the hypothesis is as follows.

Groups: A: {my} B: {cat} C: {meowed} D: {your} E: {dog} F: {slept}

Templates: 1: ABC, 2: DEF

The boxed template corresponds to the current sentence. In this and the remaining examples, the derivation is not shown because it plays little role in these simple examples.

The search begins for pairs of groups to merge. The pairs open for consideration are those containing at least one group from the current sentence’s template: D-A, D-B, D-C, D-E, D-F, E-A, and so on. In this simple example, each possible group merge results in a similar new hypothesis, so we discuss just one possibility, merging group D into group A (i.e., merging groups A and D and calling their union group A). The hypothesis resulting from this merge is as follows.

Groups: A: {my, your} B: {cat} C: {meowed} D: {dog} E: {slept}

Templates: 1: ABC, 2: AEF

On one hand, this merge is good because the number of groups will decrease from 6 to 5 (in accord with preference #4), and group A becomes more frequent in the template list (#10). On the other hand, this merge is bad because the number of words in group A increases from 1 to 2 (in violation of preference #7), and the number of groups with only one type will decrease from 6 to 4 (#8). No other preferences are affected by this merge. According to the description length function in Appendix A, the positive vote from preference #4, and the negative vote from preferences #7 and #8 are all strong votes, but the positive vote from preference #10 is relatively weak. The net effect is that the negative votes outweigh the positives, and thus this merge is not warranted. The same analysis applies to all other possible merges in this example, and so no groups are merged as a result of hearing these sentences; given that the two sentences share no context, this is the intuitively correct result.

3.5.2. Example 2: Minimal Pair Merge

Suppose the first two input sentences were my cat meowed and my cat slept. After initial processing of the second sentence, the hypothesis is as follows.

Groups: A: {my} B: {cat} C: {meowed} D: {my} E: {cat} F: {slept}

Templates: 1: ABC, 2: DEF

Most of the possible merges result in the same set of positive and negative votes as

*Merging D into A and merging A into D are identical operations, since group tags are arbitrary.*
in the first example, and thus will not be warranted. However, consider the merge of group D into group A. This would result in the hypothesis shown below (the 2 in parentheses after my indicates that 2 instances of my are in group A).

Groups: A: {my (2)} B: {cat} C: {meowed} E: {cat} F: {slept}

Templates: 1: ABC; 2: AEF

This merge is good because it reduces the number of groups from 6 to 5 (in accord with preference #4), and it puts both instances of the type my in one group (#5). No other preferences are affected by this merge. There are no negative votes, and so there is no reason not to perform the merge.

Identical reasoning applies to the merge of group E into group B, and hence there is a tie. Ties are broken arbitrarily, and here we have chosen to merge group D into group A as shown above.

In the next iteration of the process, group E is merged into group B for the reasons cited above. Now the hypothesis is:

Groups: A: {my (2)} B: {cat (2)} C: {meowed} F: {slept}

Templates: 1: ABC; 2: AEF

The algorithm iterates again, considering the pairs A-B, A-F, B-C, B-F, C-A, and F-C. All pairs result in worse hypotheses, except for merging group F into group C, which results in merging the two templates:

Groups: A: {my (2)} B: {cat (2)} C: {meowed, slept}

Templates: 1: ABC (2)

This merge is highly favored because it reduces the number of groups from 4 to 3 (in accord with preference #4), and it reduces the number and total lengths of the templates (#1 and #2). The merge is disfavored because it increases the number of types in group C (in violation of preference #7), and decreases the number of groups with one type from 2 to 0 (#8). However, reducing the number and total lengths of the templates is very important in the description length function, so the positive votes far outweigh the negative ones, and the merge is performed. After that, no further merges will be warranted.

3.5.3. Example 3: Chain reaction

Continuing the previous example, suppose that the next sentence heard is a kitten meowed. Without going through the details, it should be clear that the new instance of meowed joins the other instances of the same word by merging into group C, but nothing else happens:
Next, the sentence *a cat slept* is heard. Each new word joins the other instances of its type: *a* goes into group D, *cat* goes into group B, *slept* goes into group C; the resulting new template is DBC:

**Groups:** A: [my (2)] B: [cat (2)] C: [meowed (2), slept (2)] D: [a] E: [kitten]

**Templates:** 1: ABC (2); 2: DEC

Now there are two possible template merges, because templates 1 and 3 merge if groups A and D merge, and templates 2 and 3 merge if groups B and E merge. The former is chosen because it creates a template with higher frequency than the latter (in accord with preference #3):

**Groups:** A: [my (2), a (2)] B: [cat (3)] C: [meowed (2), slept (2)] D: [a (2)] E: [kitten]

**Templates:** 1: ABC (3); 2: DEC

And finally, merging group E into group B results in templates 1 and 2 merging:

**Groups:** A: [my (2), a (2)] B: [cat (3), kitten] C: [meowed (7), slept (7)]

**Templates:** 1: ABC (4)

This example demonstrates the power of generalized minimal pairs. The two sentences *a kitten meowed* and *a cat slept*, share only one word (*a* in the first position) and hence do not form a minimal pair. However, preceding input established that *meowed* and *slept* were in the same group, and thus the templates for the sentences formed a minimal pair. Although this example is contrived, the same basic mechanism applies to more realistic examples, as the experiments show.

3.5.4. **Example 4: Ambiguity**

Traditional distributional analysis has been rightly criticized on the basis of examples like this:

Fred saw the cat
Fred saw the dog
Where is the cat
Where is the dog
Where is the saw

In such examples, traditional methods would put both instances of *saw* into the
same category as dog and cat, thus implying that dog and cat can occur wherever saw can.\(^{10}\)

What can our proposed strategy do with these sentences? After initially processing the last sentence, and joining its first three tokens with their types, the hypothesis is:

Groups: A: \{red\} (2) B: \{saw\} (2) C: \{the\} (5) D: \{cat, dog\} (2)

E: \{Where\} (3) P: \{is\} (3) G: \{saw\}

Templates: 1: ABCD (7); 2: EFCD (7); 3: EFCDG

But what happens to the remaining word token, saw in group G? It could join the other instances of saw in group B, it could join group D because of the minimal pair in templates 2 and 3, or both things could happen—the words saw, cat, and dog could become one large group. The preferred choice depends on the frequencies of the words saw, cat, and dog, the sizes of the groups to which they belong, and the frequency of the template EFCD. The larger group B (containing saw) is, the more likely it is that the new instance of saw goes there. The more frequent the template EFCD is, the more likely it is that the new token goes into group D; that is, when the distributional context is high-frequency and hence presumably reliable, context outweighs the preference for putting instances of the same word type together. Finally, unless there are sufficient preferences against one of these choices (as is the case in this simple example), the large group containing saw, cat, and dog is created. The effects of ambiguity are explored further in Experiment 2.

To summarize, we analyzed the grouping of words with the same syntactic privileges as an optimization problem and derived a description length function using a Minimum Description Length (MDL) analysis. This description length function, combined with an incremental search mechanism, yielded an incremental grouping strategy. From this strategy, we abstracted a set of ten qualitative preferences that appear to be important considerations in categorization, regardless of the details of the MDL analysis. Further, we have argued that these preferences are intuitively plausible. Specifically, they include the grouping rules that we have referred to as generalized minimal pairs. The notion of generalized minimal pairs is intuitively appealing and similar systems have been described in the past, especially by Fries (1952). However, our MDL analysis yielded the first formal learning strategy to incorporate the full power of merging groups based on generalized minimal pairs. The generalized minimal pair abstraction does not address the issue of lexical ambiguity, nor does it handle reliability or strength of evidence, but these issues can be handled, at least in principle, by our MDL.

\(^{10}\) For instance, Pinker (Pinker, 1979, p. 240) offers the following four sentences as an example: Hottentots must survive; Hottentots must fish; Hottentots eat fish; Hottentots eat rabbits. The crucial ambiguous word is fish, which can be a noun or a verb, exactly like saw in our example.
analysis. Thus, what has been presented so far appears to be a major step forward in providing a theoretical foundation for the widely held notion that distributional analysis plays a major part in syntactic category acquisition. However, it remains to provide empirical support for the suggestions that (a) the proposed strategy could help children to group syntactically similar words from the input available to the child; and (b) children actually do display the proposed preferences, or something quite like them. In the following simulation experiments, we address the first issue, showing that the proposed strategy is effective at grouping words in both artificial languages and spontaneous, child-directed English.

4. Experiment 1

In Experiment 1, we used computer simulations to test whether the strategy described above would be useful in learning the categories of an artificial language defined in purely distributional terms. The artificial language used in the experiment was based on a simple template grammar: Sentences were generated by selecting a template, then selecting a word from each category in the template. Because the distribution of each word was defined by the templates in which its category occurred, and because the simulation program used templates to identify words’ categories, the hope was that the program would be effective in recovering the underlying categories. If not, then any subsequent success on natural language inputs would be mysterious. If the program was effective given artificial but not natural language inputs, that would suggest that our proposed strategy is good at categorizing words by analyzing their distributional contexts, but that template lists are the wrong means of tracking contexts for natural languages. However, if the program was effective given artificial language inputs and even moderately effective given natural language inputs, then this would suggest that our categorization strategy is useful, and that template lists can represent something about the distributional properties of natural languages.

We tested the program on input generated from a simple template grammar in which each word belonged to only one category. In pilot work, we discovered that the simulation program was always correct when it merged two groups, and so we focused on determining the number of sentences required to achieve a complete categorization of all words.

4.1. Method

4.1.1. Simulation program

Each simulation in Experiment 1 was run by a program that implemented the distribution-based algorithm described in the introduction and used the description length formula in Appendix A. The input to each simulation was a sequence of sentences generated from the template grammar described below, where each sentence was represented as a series of distinct word tokens. After a simulation
finished, the program returned a categorized version of the input; that is, each word token was paired with a group tag assigned during the simulation.

4.1.2. Inputs

The template grammar contained six categories, A through F, and used a set of five templates, ABC, DEF, ABCD, EAFB, and DCEF. Each category contained four word types:

A: \{a1, a2, a3, a4\} \hspace{1cm} D: \{d1, d2, d3, d4\}

B: \{b1, b2, b3, b4\} \hspace{1cm} E: \{e1, e2, e3, e4\}

C: \{c1, c2, c3, c4\} \hspace{1cm} F: \{f1, f2, f3, f4\}

Word labels were chosen for our convenience, and the simulations could not use them in the categorization process. Sentences were generated by randomly selecting one of the five templates, then, for each category in the chosen template, randomly selecting and outputting a word from that category. Templates were chosen with equal probability; since each category occurred 3 times among the 18 symbols in the template list, each was expected to comprise 1/6 of the words in a sample. Each word type was chosen from its category with equal probability (1/4). A total of 896 unique sentences could be generated from this grammar.

To examine the rate at which category information could be recovered from samples of this language, simulations were run on inputs of eight different lengths: 10, 15, 20, 25, 30, 35, 40, and 500 sentences.

4.1.3. Scoring

The output of each simulation was the input sequence of tokens, where each token was tagged with the group assigned to it by the simulation. This information, along with the correct category for each token, was given as input to a scoring program.

The scoring program determined for each pair of tokens from a simulation (a) whether the tokens belonged together (i.e., were from the same grammatical category), and (b) whether the simulation grouped them together (i.e., they had identical group tags). A pair was classified as a hit if the tokens were from the same category, and the simulation grouped them together; as a miss if the tokens were from the same category, but the simulation failed to group them together; as a false alarm if the tokens were from different categories, but the simulation put them together incorrectly; or as a correct rejection if the tokens were from different categories, and the simulation kept them apart.

We report two summary measures, accuracy and completeness. Accuracy is the percentage of pairs grouped together in a simulation that belong together. It is computed as the number of hits divided by the total number of pairs grouped by the simulation—the sum of hits and false alarms. Completeness is the percentage of pairs belonging together that the simulation program correctly grouped together.
4.2. Results and discussion

For each input-length condition, we generated 100 different input samples, ran the simulation program separately on each, and averaged the resulting scores. Every simulation yielded perfect (100%) accuracy. Thus, when the simulation merged two groups, it was always correct. Completeness scores for input lengths 10–40 are graphed in Fig. 1. Completeness starts low, near 30%, when there are only 10 sentences, and reaches 100% by 40 sentences. In fact, when the input samples contained 40 sentences (at most, 4.5% of the sentences in the language), all input sequences resulted in perfect categorization; by 30 sentences, over half of the input sequences resulted in perfect categorization.

The learning procedure we have proposed is sensitive to the number of word tokens and sentences heard, among other things (preferences 1, 2, 3, 5, 9, and 10 are sensitive to frequencies of words or templates). To ensure that more input did not worsen performance, we ran a set of 100 simulations on inputs containing 500 sentences. All achieved perfect performance.

These results suggest that the learning strategy can be effective at categorization and can generalize quite rapidly from a small sample.

5. Experiment 2

An important feature of our theory is that it explains how words can be grouped
into more than one category; we used the simulation program to test this aspect of
the theory by adding ambiguous words to the lexicon used in Experiment 1. This
made the learning task harder, and the program did not always converge on the
correct answer, but pilot work suggested that performance was better when the first
clear evidence for ambiguous words occurred later in the input. As a result, we
examined the effects on performance of the relative frequency of categorically
ambiguous lexical items, and of the number of unambiguous sentences preceding
the ambiguous ones.

5.1. Method

5.1.1. Simulation program and scoring

The same program and scoring method used in Experiment 1 were used in this
experiment.

5.1.2. Inputs

The grammar for this experiment was identical to that used in Experiment 1,
with the addition of three ambiguous words: \textit{ae} was added to both categories A
and E, \textit{bd} was added to both categories B and D, and \textit{cf} was added to both
categories C and F. This addition increased the size of the language generated to
2,125 unique sentences. Note that all 896 sentences generated by the templates
with the unambiguous lexicon from Experiment 1 are also generated by the
templates with the ambiguous lexicon.

As before, the 24 unambiguous words were chosen from their categories with
equal probability. The 3 ambiguous words were chosen from their categories with
varying probabilities. In the High Frequency condition, ambiguous types were
chosen with the same probability as any other category member. Each category
contained five words, so each word was chosen with a probability of 1/5; thus,
each ambiguous word, being a member of two categories, was expected to
comprise twice as much of a sample as each unambiguous word. In the Medium
Frequency condition, ambiguous word types were chosen with half the probability
of other category members; thus, each unambiguous and ambiguous word was
expected to comprise equal amounts of a sample. In the Low Frequency condition,
ambiguous types were chosen with a quarter the probability of other category
members; thus, each ambiguous type was expected to comprise half as much of a
sample as each unambiguous word.

The number of sentences in the sample generated using the ambiguous lexicon
was fixed at 85, because (a) fewer sentences would result in very low expected
frequencies for each ambiguous type in the Low Frequency condition, and (b) pilot
work suggested that adding more sentences did not alter performance. Inputs
containing sentences generated using the ambiguous lexicon were preceded by
either 0 (No Prefix) or 40 (Prefix) sentences generated using the unambiguous
lexicon. Thus, in the Prefix condition, the simulation program could learn a
substantial subset of the target categories before encountering ambiguous words.
The prefix size of 40 was chosen because Experiment 1 showed that 40 sentences were sufficient to establish the unambiguous component of the lexicon.

5.2. Results and discussion

Accuracy and completeness scores are shown in Table 1. Completeness was consistently between 90% and 100%. Four of the six conditions resulted in very high accuracy as well. Accuracy was greater with the unambiguous prefix than without. Also, accuracy increased as the ambiguous types became less frequent. This frequency effect was less pronounced when there was an unambiguous prefix, because accuracy was already quite high in those conditions.

The pattern of results obtained in the No Prefix condition—completeness near 100%, but accuracy below 90%—is indicative of too much merging. The high completeness implies that pairs of words that belonged together were nearly always grouped together. Given that, a low accuracy (especially as in the High Frequency, No Prefix condition) implies that many pairs were grouped together that should have been left apart. Examination of individual outputs revealed that the presence of ambiguous words could be the catalyst for merging pairs of groups sharing the same ambiguous word; that is, groups A and E, groups B and D, or groups C and F were sometimes incorrectly merged. If all three extra merges occurred, accuracy was about 30%; if only two occurred, accuracy was about 50%; and if only one occurred, accuracy was about 70%. As the frequency of the ambiguous words decreased, the frequency of these extra merges decreased. This

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Prefix Size</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 Sentences</td>
<td>40 Sentences</td>
<td></td>
</tr>
<tr>
<td>High</td>
<td>M</td>
<td>SD</td>
<td>M</td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>51.0</td>
<td>7.16</td>
<td>91.3</td>
</tr>
<tr>
<td>Completeness (%)</td>
<td>97.7</td>
<td>2.99</td>
<td>92.8</td>
</tr>
<tr>
<td>Medium</td>
<td>M</td>
<td>SD</td>
<td>M</td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>63.7</td>
<td>10.62</td>
<td>98.0</td>
</tr>
<tr>
<td>Completeness (%)</td>
<td>97.4</td>
<td>2.61</td>
<td>98.0</td>
</tr>
<tr>
<td>Low</td>
<td>M</td>
<td>SD</td>
<td>M</td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>81.7</td>
<td>14.89</td>
<td>99.4</td>
</tr>
<tr>
<td>Completeness (%)</td>
<td>98.4</td>
<td>1.69</td>
<td>99.4</td>
</tr>
</tbody>
</table>

Note: n = 100 for each condition.

11 We did not analyze these results statistically, because variances were not equal across conditions, and because of the difficulty in interpreting interactions with percentage data.

12 This observation helps explain the high standard deviations in accuracy in the No Prefix conditions. That is, there was a mix of results from up to four distributions, as simulations yielded accuracies of about 30%, about 50%, about 70%, or about 100%, depending on the input.
is because, as the frequencies of ambiguous words decreased, evidence for word ambiguities tended to occur later; hence, groups had more time to become established and the likelihood of merging whole groups decreased.

Even when a pair of groups sharing an ambiguous word was not mistakenly merged, a few instances of the ambiguous word were often incorrectly categorized in the other group of the pair. This type of error accounted for most of the decreases in accuracy in the Prefix conditions.

These results show that it takes fairly extreme conditions to make the algorithm perform poorly. On average, 55% of the sentences generated using the ambiguous lexicon in the High Frequency condition contained at least one ambiguous word. When the simulation program was given the opportunity to learn the unambiguous core of the grammar, however, it learned to categorize the ambiguous words well. The overall pattern of results is consistent with our prediction that the longer the time until evidence for an ambiguous word occurred, the better the results.

In general, the distribution-based learning procedure performed very well on input generated from simple, distributionally-defined artificial languages, even when presented with a reasonable amount of ambiguity. In order to investigate the effectiveness of this learning strategy on natural language, we turned to actual child-directed speech for our inputs.

6. Experiment 3

Since Experiments 1 and 2 suggested that the proposed categorization strategy is effective for artificial languages generated by template grammars, we set out to determine whether it is useful for categorizing words in spontaneous, child-directed English. We did not expect perfect or even near-perfect performance, for two reasons. First, our theory treats whole sentences as single contextual units, rather than using the smaller domains typical of linguistic analysis. This will tend to lead to undergeneralization of contexts, and hence to under-grouping. Second, the available transcripts of speech directed to children younger than two years old that also met our other criteria were very short (see Inputs below). As a result, we focused on whether the categorization strategy we have proposed was useful, not whether it solved the entire categorization problem.

To evaluate the usefulness of our distributional strategy, we compared the categorization performance of our distribution-based simulation program to the performance of two baseline programs that do not exploit distributional information. The first baseline program randomly assigned word tokens to groups (token baseline). However, this baseline ignored the fact that, in natural languages, two instances of the same word are more likely to be from the same category than two instances of distinct words. Thus, the second baseline program randomly assigned word types to groups, so that all instances of a word type were assigned to the same group (type baseline). Resulting categorizations for all three programs were scored as in Experiments 1 and 2, and also in a way that downplayed the importance of grouping together tokens of the same type.
6.1. Method

6.1.1. Simulation programs

The same distribution-based program from Experiments 1 and 2 was used in this experiment, with one variation: Declarative, imperative, and interrogative sentences were maintained separately in the list of templates (see Inputs below).

The token-baseline program randomly assigned word tokens to categories. It started with each word token being tagged with its correct category (see Scoring Standard below), then randomly permuted the assignments of tags to word tokens. The resulting tagging of the input preserved the original number of tokens in each category. The type-baseline program randomly assigned word types to categories. It started with each word type being tagged with the category of the plurality of its tokens; ties, which were rare, were broken by assigning the category appearing first in the sequence noun, verb, copula, auxiliary/modal, adjective, preposition, determiner, adverb, conjunction, interjection, inflection. Then, the program randomly permuted the assignments of tags to word types. The resulting tagging of the input preserved the original number of types in each category.

6.1.2. Inputs

All three simulation programs were run on edited transcripts of spontaneous, child-directed English. The inputs were the first-session orthographic transcripts made by Bernstein Ratner (1984) and were obtained from the CHILDES database (MacWhinney and Snow, 1990). The speakers were nine mothers speaking freely to their 13- to 21-month-old children (mean = 18 months); the children ranged in linguistic sophistication from being prelinguistic to producing multi-word utterances.

The original transcripts were edited to make them as internally consistent as possible. In order for the program to group together tokens of the same type (see preference #5), it must know when two tokens belong to the same type. Thus, transcripts were primarily edited to eliminate spelling variations within types. For example, the original orthographic forms da + da, dada, and daddy were all considered tokens of the type daddy. Furthermore, there were many contraction-like forms in the original, such as w(oul)dja, wouldja, and wouldya. Due to the difficulty of knowing which contractions are perceived as single words versus multiple ones, we decided to spell out all contractions. Thus, all three forms above became would you.

Sentence breaks were taken to occur wherever a period (.), comma (,), exclamation mark (!), or question mark (?) occurred (see MacWhinney, 1995, for details of the CHAT file encoding standard). Pauses, marked in the orthographic transcripts by a hash-mark (#), were treated as sentence terminators, except when they clearly interrupted a complete sentence; other obviously incorrect sentence divisions were fixed.13 All quoted material was replaced by the symbol QUOTE in

13 In child-directed speech, sentence boundaries tend to be clearly marked by pauses and other prosodic cues; see, for example, Fernald et al. (1989) and Fisher and Tokura (1996).
the main clause, then the quoted material occurred next as though it were a separate sentence.

Once delimited, each sentence was classified as being declarative, imperative, or interrogative. We assume these distinctions could be made based on prosodic information, but since prosodic information was not encoded in the original transcripts, sentences were classified on the basis of punctuation in the transcripts and by our best judgments. The point of this classification is that different sentence classes have different distributions of their constituents (at least in English); since our system works by finding distributional similarities, it is important to distinguish the classes.

Non-words were removed wherever they occurred. Sentences consisting of the repetition of just one word type, including all single-word sentences, were also removed because they contain little or no distributional information.

The resulting input files contained 172–384 sentences (mean = 244) and 662–1,553 word tokens (mean = 994).

6.1.3. Scoring

A scoring standard was created by labeling each word token in the inputs with grammatical category labels derived from those used in the Brown corpus (the correspondence between the Brown corpus labels and our labels is given in Appendix B). The following 11 categories were used: noun, verb, copula, auxiliary/modal, adjective, preposition, determiner, adverb, conjunction, interjection, and the infinitive marker to. If a word type was used in more than one category, each token was categorized according to its own environment. The tagged Brown corpus served as an authority in resolving questions about the proper category of a token.

Simulation results were scored by the algorithm used in Experiment 1. In that scoring algorithm, the total number of hits, misses, false alarms, and correct rejections contributed by each pair of word types is the number of pairs of word tokens of those two types—that is, the product of the number of tokens in each type. We call this token-scoring.

We were concerned that token-scoring might place too much emphasis on high-frequency words, so we also used a second scoring algorithm, which we call type-scoring. In type-scoring, hits, misses, false alarms, and correct rejections are tallied as described in Experiment 1, but these counts are normalized so that each pair of types affects the total score equally.\(^{14}\) For instance, if a pair of types yielded 4 hits, 1 miss, 1 false alarm, and 4 correct rejections, then there are a total of 10 token pairs, and the normalized tallies are 4/10 hits, 1/10 misses, 1/10 false alarms, and 4/10 correct rejections. Thus, the total contribution of each pair of word types to the tallies is 1.0.

\(^{14}\) We thank Paul Smolensky for this idea.
Table 2  
Average accuracy and completeness by simulation program and scoring method (Experiment 3)

<table>
<thead>
<tr>
<th>Simulation Program</th>
<th>Token-Scoring</th>
<th>Type-Scoring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M</td>
<td>SD</td>
</tr>
<tr>
<td>Token Baseline</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>22.8</td>
<td>2.01</td>
</tr>
<tr>
<td>Completeness (%)</td>
<td>22.8</td>
<td>2.01</td>
</tr>
<tr>
<td>Type Baseline</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>27.5</td>
<td>2.28</td>
</tr>
<tr>
<td>Completeness (%)</td>
<td>31.8</td>
<td>2.68</td>
</tr>
<tr>
<td>Distributional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>85.3</td>
<td>9.43</td>
</tr>
<tr>
<td>Completeness (%)</td>
<td>17.8</td>
<td>6.05</td>
</tr>
</tbody>
</table>

Note: n = 9 for each condition.

6.2. Results and discussion

The scores obtained by applying both scoring methods to the results of all three simulations are given in Table 2. All scores are averages over the nine inputs. Accuracy was significantly greater for the distributional algorithm than the token-baseline algorithm (using token-scoring, t(8) = 21.81, p < .0001; using type-scoring, t(8) = 27.57, p < .0001) and the type-baseline algorithm (using token-scoring, t(8) = 20.29, p < .0001; using type-scoring, t(8) = 27.40, p < .0001). Completeness was significantly lower for the distributional algorithm than the token-baseline algorithm (using token-scoring, t(8) = 2.88, p = .02; using type-scoring, t(8) = 25.09, p < .0001) and than the type-baseline algorithm (using token-scoring, t(8) = 7.54, p < .0001; using type-scoring, t(8) = 35.30, p < .0001). Because the pattern of results was the same for both scoring methods, we only report the results of token-scoring in the remaining experiments.

The distribution-based simulation program did a better job categorizing the child-directed inputs than either baseline program. The baseline programs were so inaccurate that they hardly qualify as having learned anything useful; they certainly are not a foundation upon which future learning can be built. On the other hand, the distributional program was generally very accurate, no matter which scoring system was applied. The relatively high completeness of the baseline programs is not surprising, as they were constrained to use the correct number of groups. With all words being placed into only 11 groups, chances were high that each type and token would be correctly grouped with many other types and tokens from its category. The distributional algorithm did not know the correct number of groups, and its conservative merging strategy led to low completeness, especially at the level of types.

As before, we were concerned about the lengths of our input files. Although the distributional simulation program seemed to perform conservatively enough to support further learning, this was not adequately tested by our short inputs.
Therefore, in Experiment 4 we ran the same three algorithms on a single, large input formed by concatenating all nine of the inputs used in this experiment.

7. Experiment 4

7.1. Method

7.1.1. Simulation programs

The three simulation programs from Experiment 3 were used.

7.1.2. Inputs

The nine input files from Experiment 3 were concatenated (in alphabetical order) to create one input file of 2,197 sentences and 8,947 word tokens.

7.1.3. Scoring

Accuracy and completeness scores were computed using token-scoring as described in Experiment 3. In addition, we examined performance of the distributional simulation program over time by computing scores after processing each block of 100 sentences. Scoring in this incremental case was limited to pairs of words containing at least one member from the current block, so that each pair of tokens in the entire input was counted exactly once.

7.2. Results and discussion

The results of all three simulations are given in Table 3. Fig. 2 is a graph of the incremental scores from the distributional program.

Using the substantially longer input file resulted in the same overall pattern obtained in Experiment 3: The distributional program categorized words much more accurately than the baseline programs, but was slightly more conservative in doing so.

For the distributional program, the results show that accuracy declined in early processing, leveling off to about 70% after approximately 1,000 sentences. It is difficult to interpret the early variability of the completeness scores, but completeness generally stayed constant at about 20%.

<table>
<thead>
<tr>
<th>Simulation Program</th>
<th>Accuracy (%)</th>
<th>Completeness (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token Baseline</td>
<td>22.6</td>
<td>22.6</td>
</tr>
<tr>
<td>Type Baseline</td>
<td>26.4</td>
<td>34.4</td>
</tr>
<tr>
<td>Distribution-Based</td>
<td>68.1</td>
<td>22.0</td>
</tr>
</tbody>
</table>

Note: The scores for each simulation program come from exactly one simulation run.
As discussed earlier, exploiting a correlation between semantic and syntactic categories might be useful in learning grammatical categories. For instance, a child could learn that certain words refer to concrete objects, perhaps frequently hearing ball in the context of seeing balls and doggie when Fido is around. Once confident of the words’ referents, the child could use canonical structural realizations (Grimshaw, 1981) to infer that ball and doggie belong to the same grammatical category. In the next two experiments, we tested whether using semantic information in this way would improve the performance of the distributional program.

Most researchers agree that children can learn the meanings of nouns referring to concrete objects by observing the real-world contexts in which they are used. In this experiment, we examined the potential influence of semantic information on our distributional simulation program.

The simulations were given a small set of high-frequency nouns that were marked as referring to concrete objects; these words were automatically grouped

---

15 For words that are heard only in multiword utterances, children must figure out which word in the utterance refers to the observed object. In the current paper, we have followed most other theorists in abstracting away from the problems of segmentation and word discovery. This idealization implies that children know when they are hearing a single-word utterance. To the extent that this is true, it would be sufficient for present purposes if children applied canonical structural realizations only to single-word utterances. For other, more exciting approaches to the problem of resolving reference in multiword utterances, see Fisher et al. (1994); Gleitman and Gillette (1995); Pinker (1984); Siskind (1996).

16 The extent to which they can learn the meanings of verbs from real-world context alone is a topic of debate (e.g., Gleitman, 1990; Gleitman and Gleitman, 1992; Pinker, 1984); see Brent (1996) for further discussion.
together. It is possible that distributional information alone was sufficient to group these nouns together without the semantic information, in which case we would not expect to see an improvement in scores. In other words, the semantic cues may be redundant. If not, then perhaps the semantic information would not only help group together the high-frequency nouns, but would also affect the distributional analysis in some way. Thus, we also tested whether the addition of semantic information resulted in changes to the categorization of words that were not semantically marked.

8.1. Method

The distributional program from Experiments 1–4 was used. When given the set of words known to refer to concrete objects (listed below), the program grouped together instances of those words without regard to their distributional properties. The group containing the concrete-object words was not treated specially in any other respect.

The nine input files from Experiment 2 were used. In the Semantics condition, the 24 most frequent nouns referring to concrete objects were marked as such. Word frequency was based on frequency counts from the complete set of nine files; each word in the list occurred at least 14 times. The words were block, blocks, book, box, boy, brush, bunny, chair, daddy, dog, doggie, door, dragon, flowers, girl, hair, hand, house, kitty, mommy, paul, phone, ring, and telephone.

To test whether the addition of semantic information improved performance, accuracy and completeness scores were computed for simulations with and without semantic information using token-scoring. To test whether the addition of semantic information affected words other than the semantically-labelled nouns, accuracy and completeness were also computed using a variant of token-scoring: If both members of a pair of tokens were semantically labelled, that pair was excluded from scoring; otherwise the pair was scored as usual. The goal of this modification was to exclude from scoring all pairs that were forced into the same group by semantics independently of distribution. The pairs that were scored were grouped for distributional, not semantic, reasons, although the distributional analysis could have been affected by the forced grouping of the 24 nouns. The simulations from Experiment 3, which did not use semantic information, were re-scored by the same method so that simulations with and without semantics could be compared.

8.2. Results and discussion

The average scores for each condition are given in Table 4. The upper half of the table shows results from scoring all token pairs: Average accuracy declined with semantic labelling, but this difference was not significant (t(8) = 1.54, p = .16); average completeness improved, and this improvement was significant (t(8) = 3.16, p = .01). The lower half of the table shows results from scoring the restricted set of token pairs: Average accuracy declined with semantic labelling,
Table 4
Average accuracy and completeness by semantics condition and scoring system (Experiment 5)

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Accuracy (%)</th>
<th>Completeness (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M</td>
<td>SD</td>
</tr>
<tr>
<td>Scoring All Token Pairs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Without</td>
<td>85.3</td>
<td>9.43</td>
</tr>
<tr>
<td>With</td>
<td>81.7</td>
<td>12.40</td>
</tr>
<tr>
<td>Scoring Token Pairs not Consisting of Two Semantically-Labelled Tokens</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Without</td>
<td>84.6</td>
<td>10.69</td>
</tr>
<tr>
<td>With</td>
<td>80.2</td>
<td>13.99</td>
</tr>
</tbody>
</table>

Note: \( n = 9 \) for each condition.

but this difference was not significant (\( t(8) = 1.75, \ p = .12 \)); average completeness improved, and this improvement was nearly significantly (\( t(8) = 2.29, \ p = .051 \)).

The results from scoring all token pairs show that the addition of even a small amount of semantic information improved performance by leading to more correct group merges. Therefore, the semantic information was not redundant: It could have been that all of the 24 most frequent nouns were grouped correctly without semantic information, but this was not the case. Nor was it the case that none of them was grouped correctly without semantic information. By looking at the output, we determined that the simulations using only distributional information correctly grouped together some of the frequent nouns (usually about half of the possible occurrences). Thus, using distributional information alone was effective at grouping some of the nouns, and using both distributional and semantic information was even more effective.

The results obtained by excluding semantically-grouped pairs of tokens suggest that the addition of semantic information also improved the distributional analysis; however, the evidence for this improvement was not as strong as we might hope for. This may reflect a limitation of our method of distributional analysis, or the method by which semantic information was used.

9. General discussion

The purpose of these simulations was to investigate whether the proposed learning strategy would be effective at categorizing words. The quantitative results suggest that the simulation program was able to begin grouping words based on their distributional patterns, and was accurate enough that subsequent language acquisition processes would not be crippled by early mistakes. Qualitative results give a better understanding of the behaviors behind the numbers, so we begin the discussion with a qualitative characterization of both the simulation results and processing behavior. Then, we discuss the theoretical implications of the simulation results, first for our theory, then for semantics-based theories of category
acquisition. Next, we consider ways in which the apparently diverse theories of category acquisition might be unified. Finally, we discuss the assumptions behind our theory and some possible extensions of it.

9.1. Qualitative evaluation of outputs

First, we examine output from the distribution-based simulation program (see Cartwright, 1996, for the complete set of simulation outputs). For example, on one input file used in Experiment 3, the simulation returned 17 groups containing two or more types, listed below with each word’s frequency shown in parentheses.

1. smile (2), ball (1), balloon (1), bunny (2), kitty (1), boy (4), house (1), phone (5), doggie (10), telephone (2), block (9)
2. this (19), that (36), it (39), alice (5), him (4)
3. feed (4), get (13), feel (3), do (12)
4. open (2), take (4), pull (2)
5. right (2), away (6), okay (1)
6. on (7), in (6), one (1)
7. want (14), say (14)
8. the (34), a (33)
9. what (43), who (2)
10. out (5), up (4)
11. touch (3), tickle (2)
12. your (7), alice’s (1)
13. sit (1), fall (1)
14. go (5), are (9)
15. daddy (8), good (5)
16. to (20), QUOTE (13)
17. sorry (6), lonesome (1)

In addition, the simulation returned 121 single-type groups, of which 64 (53%) consisted of words that occurred only once in the input sample. For this simulation, token-scoring yielded an accuracy of 95.1% and a completeness of 17.3%; type-scoring yielded an accuracy of 94.2% and a completeness of 5.4%.

17 The single-type groups containing relatively high frequency words were is (66), you (50), can (15), i (15), there (10), me (9), am (8), brush (7), hair (7), look (7), and at (6). The remaining single-type groups are given in blocks by decreasing frequency: 5: blocks, book, for, here, and let; 4: down, going, nice, see, them, think, too, and will; 3: how, kiss, not, put, something, and these; 2: about, and, answer, baby, come, could, didn’t, does, dolly’s, don’t, give, have, hello, many, milk, monkey, now, over, says, smells, soft, toys, wait, was, where, with, and would; 1: again, all, another, build, built, butterfly, button, can’t, comb, daddy’s, did, doggie’s, door, drink, ear, eat, face, fingers, girl, got, has, hat, he, hi, his, hug, hungry, just, kisses, knocked, know, laugh, licking, like, loren, mommy, or, own, picture, press, really, ring, scare, scared, second, show, so, speak, squeak, sweet, talk, tell, thank, things, tongue, touching, try, us, we, why, wicked, wild, and yourself.
Token-scoring yielded higher completeness because it gave more weight to the fact that tokens of each type were usually grouped together.

The high accuracy scores reflect two facts: Most words were used unambiguously in the input, so the accuracy was not hurt by the program’s unambiguous grouping; and the word types that were grouped together in the output belonged together, with only a few exceptions. Groups 1 and 2 contain nouns—singular count nouns in group 1, and lexical noun phrases in group 2 (noun groups are not always so cleanly divided into sub-types, as was the case with this file). Groups 3, 4, 7, 11, and 13 are all perfect verb groups: Group 13 contains intransitive verbs, and the other verb groups contain transitive verbs. In fact, the only grouping errors are in groups 5, 6 (two prepositions and the numeral one), 15 (daddy and good), and 16 (to and QUOTE—the quotation placeholder). As an example of the sort of input that causes errors, daddy and good were grouped together because of the sentences

was that good?
was it daddy?

The words that and it were already grouped together (group 2), so these sentences formed a generalized minimal pair.

Overall, the scoring methods used in the experiments seem to accord well with a qualitative evaluation of the simulation results.

9.2. Qualitative evaluation of the processing behavior

In the Introduction, we described our theory in terms of preferences and a simple search algorithm. While these two components fully and accurately describe the processing of a sentence, they do not make clear the typical order of the events that occur during processing. Two processes emerged as central to the analysis of most sentences: (a) Each token in the current sentence, initially assigned to a new, distinct group, was merged into a group containing other instances of the same word type, and (b) groups were merged when doing so led to template merges (i.e., when two templates formed a minimal pair). When a new token could be merged into more than one group containing other instances its type, that token was usually put into the group containing the greater number of instances of its type.

When there was a choice between joining a new token with others of its type or merging templates, the template merge was preferred. However, template merges could not occur until after all but one of the token-into-type merges. For example, consider the sentences

The error is that away was categorized as an adverb, but right and okay were adjectives. It turns out that away may be an adjective in this case; the mother said, several times, 'It’s away,' to convince her daughter that a scary toy had been put away. Nevertheless, away was categorized as an adverb in the scoring standard.
The cat meowed
The cat slept

In processing the second sentence, the tokens of the, cat, and slept are assigned to new groups—say X, Y, and Z—and the corresponding new template is created for the sentence—XYZ. There is no overlap between this and any other template, because the groups in the template (X, Y, and Z) are new. Thus, no single group merge can cause the new template to merge with an existing template. But suppose that group A already contains tokens of the, and group B already contains tokens of cat. Then, token-into-type merges occur: Group X (the) is merged into group A, and group Y (cat) is merged into group B. Now, the template for the first sentence is ABC (where group C contains tokens of meowed), and the template for the second sentence is ABZ, and so merging groups C and Z will result in a template merge.

The interplay of these processes determines when a type’s instances are put into more than one group. When just one token of a sentence remains to be merged into an existing group, two things can happen: It can merge into a group containing other instances of the same type, or it can merge to cause a template merge. In most cases, one of these choices does not exist, or both choices involve the same merge; but in rare cases, both possibilities exist and imply different group merges. When such a choice exists, the template merge usually wins, and the token in question is merged into a group that did not previously contain other instances of that type; exceptions occur when the group containing other instances of the type is large, and the templates under consideration are short or rare.

9.3. Implications for other theories

The theory proposed in this paper improves upon earlier theories based on the use of distributional information: It is detailed and explicit; it assumes sentences are processed one at a time without requiring old sentences to be memorized; it results in discrete categories; it allows words to be in multiple categories; it can exploit other information sources (such as semantics); and it is based on established formal principles of induction (MDL). These are useful properties, but what does the current proposal say about the ultimate role of distributional analysis in a complete theory of category acquisition? In this section, we first suggest some possible answers to this question, then consider another proposal of grammatical category acquisition Pinker’s (Pinker, 1984, 1987) semantic bootstrapping hypothesis.

9.3.1. Distributional analysis and syntactic structure

The distributional learning strategy proposed here exploits minimal knowledge about the syntactic structure of language. The structural knowledge it does use can be summarized in three points. First, it presupposes that words belong to discrete categories such that substituting words within the same category preserves grammaticality. Second, it presupposes that word types may belong to more than
one category. Third, it presupposes that the sentence is a valid domain over which to conduct distributional analysis; in other words, sentence boundaries are syntactic boundaries. Despite the fact that the proposed learning strategy exploits so little syntactic knowledge, it was able to effectively group syntactically related words in the computer simulations reported here. This suggests that a substantial amount of categorial inference is possible with minimal structural inference. Further, we have shown for the first time that such categorial inference can be done incrementally; that is, one sentence at a time.

It is not yet clear how much children rely on structural knowledge for inferring syntactic categories. We have shown that they could go surprisingly far with very little structural knowledge, but they may well use more structural knowledge than our strategy implies, when such knowledge is available. Indeed, fairly sophisticated structural knowledge is probably required to achieve accurate and robust categorization (e.g., Pinker, 1984).

On the other hand, there are two reasons to believe that children may possess a categorization mechanism as minimal as the one we have proposed. First, it is hard to imagine how children could learn much about their native languages' syntactic structures without already knowing something about the grouping of words into categories, even with the benefit of a highly-restrictive innate grammar. Indeed, almost every theory of syntactic acquisition presupposes at least some category acquisition. Thus, there may be a critical bootstrapping stage at which a minimal categorization mechanism of the type we have proposed is dominant.

The second reason to believe in a stage at which categorization relies on minimal structure is computational. Clearly, it is not possible for all of syntactic learning to proceed independently of categorization. Thus, there are only two choices: Either there is some categorization independent of syntactic structure, or else categorization and learning syntax are interdependent throughout language acquisition. The latter assumption would require children to learn more things and depend on more assumptions at once. This means that they must consider more possible combinations of hypotheses about their languages. This is not necessarily impossible, but it is certainly more computationally challenging than keeping categorization independent of syntactic structure until the broad outlines of each have become clear.

Next, we consider another cognitive approach to category acquisition—one that can use semantic information to link words to syntactic categories.

9.3.2. Semantic bootstrapping

The main point of Pinker's (Pinker, 1984, 1987) semantic bootstrapping hypothesis is that the process of linking words to their appropriate syntactic categories is initiated using semantic information. In addition to this well-known component of his work, Pinker also argues for the need for a distributional analysis component:

[The semantic bootstrapping hypothesis] does not claim that children fail to perform distributional analyses or that they allow semantically based
analyses to override distributionally based analyses. Rather, it claims that children always give priority to distributionally based analyses, and is intended to explain how the child knows which distributional contexts are the relevant ones to examine. (Pinker, 1984, p. 42).

Pinker elaborated how distributional analyses could override semantic analyses of words’ categories (Pinker, 1984, p. 114), and even suggested that, “…simplistic distributional properties such as word order and co-occurrence... could have a role to play [in bootstrapping syntactic categories].” (Pinker, 1987, p. 438). Presumably this sort of distributional analysis method would include the one presented in this paper.

Thus, semantic and distributional analyses seem complementary, and so we believe our theory could be dovetailed with Pinker’s. To accomplish this, the work presented in Experiment 5 would have to be extended. In that experiment, semantic information was used to categorize words independently of distributional information, and distributional evidence could not override the categorizations forced by the semantic information. It may be worthwhile to examine a theory in which the distributional analysis strategy we have proposed can override semantic information. In sum, the results of Experiment 5 suggest that semantic information is useful, but the details of how it is best exploited remain to be worked out.

9.4. Assumptions and extensions

The theory proposed in this paper is based on simplifying assumptions about the structure of the input, the role of long-term memory, the role of syntax (mentioned above), and the structure of the output. In future research, we hope to examine and perhaps relax some of these assumptions.

9.4.1. Input

Our theory assumes that the input to early categorization is a sequence of sentences, each represented as a sequence of word tokens. This assumption that speech can be parsed into discrete words is standard (e.g., Pinker, 1984, p. 28). Furthermore, we have developed a theory (based on formal principles similar to those underlying the theory presented here) of how children could use distributional and other cues to learn to segment speech and acquire a lexicon (Brent and Cartwright, 1995, 1996). Also, there is a growing body of research that suggests that infants can segment out at least familiar words from continuous speech by approximately 7.5 months (Jusczyk and Aslin, 1995).

But what if the input is represented differently? Perhaps input symbols are not exactly orthographically-defined words, but rather units at some other level of representation. Because our learning strategy does not exploit any properties of input tokens other than their distributions, there is no obvious reason why it would work less effectively given inputs represented as morphemes or as a mixture of morphemes, morphemically-complex words, and common phrases or idioms (e.g., thank + you as one ‘word’). In fact, it seems that distributional information about
inflectional morphemes, such as the past tense marker -ed, could be useful to the categorization process (e.g., Maratsos and Chalkley, 1980). However, these ideas remain to be empirically tested.

Independent of the level of representation of words is the way in which speech is divided into sequences of words. For instance, several researchers (e.g., Fisher and Tokura, 1996; Gleitman and Wanner, 1982; Jusczyk et al., 1992) have suggested that young children may use various prosodic features to identify major phrase boundaries in speech; if so, then perhaps the input to early categorization is a sequence of phrases instead of a sequence of sentences. We know of no reason why our learning strategy would be less effective given phrase-sized inputs. In analyzing the simulation results, we found that generalized minimal pairs of short sentences tended to cause the most group merges (see Cartwright, 1996, for more details); thus, if input sequences were shorter on average—because they were phrases, not sentences—we expect that more groups would be merged. Assuming that some of these extra merges would be correct and some incorrect, it seems most likely that completeness would improve somewhat and accuracy would decline somewhat.

9.4.2. Memory

Our theory does not address the role of long-term memory, but in the simulations, long-term memory was perfect: Once a token was put in some group, it was always perfectly remembered as a member of that group. Although children’s ability to learn linguistic knowledge seems good, it probably is not that good, and so the way in which long-term memory is modeled by the simulation program could be refined.

A model of forgetting may improve our explanation of early category acquisition. Recall that group merges are permanent in the current theory, so that once two groups are merged, they cannot be pulled apart again, nor can individual types or tokens be removed from the group. Thus, early incorrect groupings can only be overcome by putting new instances of the incorrectly grouped words into a new, more appropriate group; assuming all new instances of a given word are grouped correctly, the number of erroneously grouped tokens will eventually be small relative to the number of correctly grouped ones. This solution is not completely satisfying, however, because the old tokens remain incorrectly grouped. Ideally, in such a situation the older tokens would be forgotten as members of their group, thereby purifying the categorization. That is, the presence of a word in a group would depend on there having been relatively recent instances of that word in that group. This suggestion is related to others in which forgetting may result in improved learning (Braine, 1971; Goldowsky and Newport, 1993).

In general, forgetting would not be part of the theory of categorization itself, as the strategies therein are used solely to determine which group merges are best.

9.4.3. Syntax

As mentioned in the discussion of the semantic bootstrapping hypothesis, the method of tracking contexts could be refined to take into account the syntactic
structure of sentences. One problem to be resolved is: Which comes first, the syntax or the categories? That is, syntax is defined in terms of categories, but categories are defined in terms of their use in the syntax. Pinker’s (Pinker, 1984, 1987) semantic bootstrapping hypothesis offers one way to break the circularity, as do prosodic bootstrapping proposals (Gleitman and Wanner, 1982; Morgan et al., 1987; Morgan and Newport, 1981). As discussed above, it seems to us that a relatively limited form of syntax is required to assist category acquisition, but this is another area for future work.

9.4.4. Output

The theory proposed in this paper presupposes that the number and composition of categories is unknown at the onset of learning, and that discrete grammatical categories are the correct result of learning. Since the internal representations of grammar and the lexicon are not well understood yet, we cannot be certain that discrete categories are the correct form of output. Perhaps learning hierarchies of groups is most appropriate. However, we note that nearly every proposed theory of syntax makes use of a relatively small number of discrete categories. Thus, for the moment this assumption seems safest.

Universal Grammar may specify some or all of the grammatical categories used by languages. Our theory would have to be non-trivially revised to incorporate this assumption because it is currently motivated in part by the need to discover the correct number of categories. Nevertheless, it seems worthwhile to investigate the consequences of assuming that the number of categories is innate.

9.5. Conclusion

These experiments suggest that the theory proposed in this paper is capable of explaining aspects of early grammatical category acquisition. We have shown that it is possible to define a form of distributional analysis that can be applied to one sentence at a time. Further, we have shown one way in which distributional and semantic information could be combined in category acquisition. The theory is founded on established principles of induction, yet its qualitative behaviors are understandable and intuitively appealing independent of those principles; it is detailed and explicit, and can be directly implemented as a computer simulation. Finally, it seems possible to connect this theory to others in the literature, thereby increasing our understanding of the range of processes involved in language acquisition.

Acknowledgments

We wish to thank Lila Gleitman, Alec Marantz, Mike McCloskey, and Paul Smolensky whose comments and criticisms were valuable in the preparation of this paper. This work was supported in part by NIH grant 1 R29 DC 030382-01 to
Appendix A

Details of the description length formula

In this appendix, we present the description length formula used in the experiments, motivate this formula, and relate it to the preferences described in the body of the paper.

The description length formula given below results from a Minimum Description Length (MDL) analysis of the categorization problem. We used the MDL framework because it has been successfully applied to other language learning problems (e.g., Ellison, in press; Stolcke and Omohundro, 1994) and to modeling other aspects of language acquisition (Brent and Cartwright, 1996). For a complete exposition of the MDL framework, see Li and Vitányi (1993), especially chapters 1 and 5; for an overview of Huffman coding, see Cormen et al. (1990); and for a discussion of the relationship of MDL to Maximum Likelihood and Bayesian inference, see Rissanen (1989), especially chapters 1 and 2.

We now present the variables used in the description length formula and the formula itself, followed by a description of the motivation for the different terms in the formula. Table A1 defines the variables in the formula. Any given categorization of the input determines a value for each of these variables. The complete description length function is

\[
g \left( \frac{wH(GROUPS[*]) + GH(GFREQ[*]) + TH(TFREQ[*]) + \sum_{j=1}^{g} m_j H(FREQ[*],j)}{n + t + \sum_{j=1}^{g} n_j} \right) + g - o + 2 \left( n + t + \sum_{j=1}^{g} n_j \right) + 3
\]

Table A1

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Number of word types in the input</td>
</tr>
<tr>
<td>g</td>
<td>Number of distinct groups in the tagged input</td>
</tr>
<tr>
<td>o</td>
<td>Number of groups containing only one word type</td>
</tr>
<tr>
<td>t</td>
<td>Number of templates implied by the tagged input</td>
</tr>
<tr>
<td>GROUPS [i]</td>
<td>Number of groups in which word type i occurs</td>
</tr>
<tr>
<td>GFREQ [j]</td>
<td>Number of occurrences of group j in the template list</td>
</tr>
<tr>
<td>FREQ [i,j]</td>
<td>Number of instances of word type i in group j</td>
</tr>
<tr>
<td>TFREQ [k]</td>
<td>Number of occurrences of template k</td>
</tr>
<tr>
<td>TLEN [k]</td>
<td>Length, in group tags, of template k</td>
</tr>
</tbody>
</table>
In the MDL framework, the optimal description of some data is the one in which the data are represented most compactly in terms of a hypothesis and a derivation. In our MDL analysis of the categorization problem, hypotheses are representations of the assignment of words to groups and the composition of templates in terms of those groups; derivations are representations of the sequence of templates observed in the input and the words, within their groups, occurring at each template position. Specifically, hypotheses assign distinct sets of indices to the word types and groups; the word-type indices are used to represent which words occur in each group, and the group indices are used to represent templates. Derivations assign distinct sets of indices to templates and word types within their groups; template indices are used to represent the sequence of templates observed in the input, and word-type-within-group indices are used to represent, for each position of each sentence, which word occurred in the input.

The representational system uses four distinct sets of indices to stand in place of various objects: Word types, groups, templates, and word types within groups. In order to achieve a compact representation of the input, we should try to use indices that result in optimal coding lengths (Shannon and Weaver, 1949); in practice, Huffman codes (Hamming, 1986; Huffman, 1952) yield near-optimal encodings and can be easily computed. The fundamental idea of optimal coding is to assign indices in such a way that all objects receive the shortest possible indices; when trade-offs must be made, the most frequent objects receive the shortest indices, and the least frequent objects receive the longest indices.

The description length formula shown above is dominated by four entropy-times-length terms that results from optimally coding various sequences in the representational system. Each of these terms is described below.

Each word type is assigned an index, and these indices are used to represent which words occur in each group. A word type $i$ occurs in $\text{GROUPS}[i]$ groups, so the total number of word/group pairs is

$$w = \sum_{i=1}^{n} \text{GROUPS}[i]$$

The entropy of the relative group counts over all word types is

$$H(\text{GROUPS}[\ast]) = -\sum_{i=1}^{n} \frac{\text{GROUPS}[i]}{w} \log \left( \frac{\text{GROUPS}[i]}{w} \right)$$

Each group is assigned an index, and these indices are used to represent templates. Group $j$ occurs $\text{GFREQ}[j]$ times in the list of templates, so the total number of group indices used in the template list is

$$G = \sum_{j=1}^{g} \text{GFREQ}[j]$$

The entropy of the relative frequencies of groups in the template list is
Each template is assigned an index, and these indices are used to represent the sequence of templates observed in the input. Each template $k$ is used $\text{TFREQ}[k]$ times in representing the input, so the total number of template instances, which is equal to the total number of sentences in the input, is

$$T = \sum_{k=1}^{t} \text{TFREQ}[k]$$

The entropy of the relative frequencies of the templates is

$$H(\text{TFREQ}[*]) = -\sum_{k=1}^{t} \frac{\text{TFREQ}[k]}{T} \log \left( \frac{\text{TFREQ}[k]}{T} \right)$$

Within each group, each word type is assigned an index, and these indices, from all groups, are used to represent the words within their groups observed in the input. A word $i$ occurs $\text{FREQ}[i,j]$ in group $j$, so the total number of word tokens in group $j$ is

$$m_j = \sum_{i=1}^{n} \text{FREQ}[i,j]$$

The entropy of the relative frequencies of word types within group $j$ is

$$H(\text{FREQ}[*], j) = -\sum_{i=1}^{n} \frac{\text{FREQ}[i,j]}{m_j} \log \left( \frac{\text{FREQ}[i,j]}{m_j} \right)$$

In addition to using the indices to represent various sequences, the indices themselves and the way in which they correspond to objects must be represented. As shown in Fig. 3, indices constructed by Huffman coding can be represented by a tree, where the leaves represent the items to be encoded, each left branch is labeled 0, each right branch is labeled 1, and the index for each item is the sequence of 1s and 0s on the path from the root of the tree to that item. For example, the code for $A$ in Fig. 3 is 00, the code for $B$ is 01, the code for $C$ is 100, and so on. When the code is constructed according to the Huffman procedure,
every non-leaf node will be binary branching. In our representation, leaves do not need to be labeled with objects, because objects are ordered by other means and assigned to leaves in left-to-right order. However, we do need to represent the shape of the tree—which nodes are leaves and which are branching nodes. This can be done using exactly one bit per node, as described by Quinlan and Rivest (1989), resulting in a representation that takes \(2x-1\) bits to represent \(x\) indices.

Thus, representing the indices of the \(n\) word types takes \(2n-1\) bits; representing the indices of the \(g\) groups takes \(2g-1\) bits; and representing the indices of the \(t\) templates takes \(2t-1\) bits. Representing the indices of the \(n_j\) word types within group \(j\) takes \(2n_j-1\) bits. However, if there is only one type in group \(j\), then no index is needed for that type; we stipulate that such groups require zero bits, whereas the formula given above says that \(2(1)-1=1\) bit is needed. To get the total length of representing the within-group indices, we sum \(2n_j-1\) over all groups \(j\) and correct for the groups containing only one type:

\[
\sum_{j=1}^{g} (2n_j - 1) - o
\]

Finally, each template must be associated with its length, measured in groups. Each template length is represented as a fixed-length binary number whose length is that required by the longest template. There are \(t\) such numbers, so the total length of the template lengths is

\[
t \log \max_k \text{TLEN}[k]
\]

Summing the lengths of the representational system’s components and simplifying algebraically yields the formula given at the beginning of the appendix.

In Table A2, we show the correspondence between terms of the description length formula and the preferences; note that some terms result in more than one preference, and some preferences result from more than one term.

<table>
<thead>
<tr>
<th>Table A2</th>
<th>Relationship between description length formula and preferences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formula term</td>
<td>Preferences(s)</td>
</tr>
<tr>
<td>(wH) (GROUPS[*])</td>
<td>5, 6, 7</td>
</tr>
<tr>
<td>(GH) (GFREQ[*])</td>
<td>1, 2, 10</td>
</tr>
<tr>
<td>(TH) (TFREQ[*])</td>
<td>3</td>
</tr>
<tr>
<td>(\Sigma m.H(\text{FREQ}[*], j)))</td>
<td>4, 6, 7, 9</td>
</tr>
<tr>
<td>(t \log \max_k \text{TLEN}[k])</td>
<td>1</td>
</tr>
<tr>
<td>(g)</td>
<td>4</td>
</tr>
<tr>
<td>(-o)</td>
<td>8</td>
</tr>
<tr>
<td>(2n)</td>
<td>—</td>
</tr>
<tr>
<td>(2t)</td>
<td>1</td>
</tr>
<tr>
<td>(2 \Sigma n_j)</td>
<td>5, 6, 7</td>
</tr>
</tbody>
</table>
Appendix B

Relationship between Brown corpus tags and our category tags

The table below indicates which Brown corpus tags (Francis and Kucera, 1982) correspond to each of our category tags. The various forms of *do* and *have* (Brown corpus tags DO, DOD, DOZ, HV, HVD, HVG, HVN, HVZ) could be true verbs or auxiliaries; the Brown corpus tagging system does not distinguish between those uses, but we did.

<table>
<thead>
<tr>
<th>Category</th>
<th>Brown Corpus Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noun</td>
<td>CD, EX, NN, NNS, NP, NPS, NR, NR, PN, PPL, PPLS, PPO, PPS, PPSS, WPO, WPS</td>
</tr>
<tr>
<td>Verb</td>
<td>DO, DOD, DOZ, HV, HVD, HVG, HVN, HVZ, VB, VBD, VBG, VBN, VBZ</td>
</tr>
<tr>
<td>Copula</td>
<td>BE, BED, BEDZ, BEG, BEM, BEN, BER, BEZ</td>
</tr>
<tr>
<td>Aux./Modal</td>
<td>DO, DOD, DOZ, HV, HVD, HVG, HVN, HVZ, MD</td>
</tr>
<tr>
<td>Adjective</td>
<td>JJ, JJR, JJJS, JJT, OD</td>
</tr>
<tr>
<td>Preposition</td>
<td>IN</td>
</tr>
<tr>
<td>Determiner</td>
<td>ABN, ABX, AP, AT, DT, DTI, DTS, DTX, NNS, NNS$, NPS$, NPS$, NPS$, NPS$, WDT, WPS</td>
</tr>
<tr>
<td>Adverb</td>
<td>ABL, QL, QLP, RB, RBR, RBT, RN, RP, WQL, WRB</td>
</tr>
<tr>
<td>Conjunction</td>
<td>CC, CS</td>
</tr>
<tr>
<td>Interjection</td>
<td>UH</td>
</tr>
<tr>
<td>Inflection</td>
<td>TO</td>
</tr>
</tbody>
</table>

References


