Simplicity and learning to distinguish arguments from modifiers

Leon Bergen[1]*, Edward Gibson*[2]*, and Timothy J. O'Donnell*[3]
[1] University of California San Diego
[2] Massachusetts Institute of Technology
[3] McGill University, Canada CIFAR AI Chair, Mila

ABSTRACT

We present a learnability analysis of the argument-modifier distinction, asking whether there is information in the distribution of English constituents that could allow learners to identify which constituents are arguments and which are modifiers. We first develop a general description of some of the ways in which arguments and modifiers differ in distribution. We then identify two models from the literature that can capture these differences, which we call the argument-only model and the argument-modifier model. We employ these models using a common learning framework based on two simplicity biases which tradeoff against one another. The first bias favors a small lexicon with highly reusable lexical items, and the second, opposing, bias favors simple derivations of individual forms – those using small numbers of lexical items.

Our first empirical study shows that the argument-modifier model is able to recover the argument-modifier status of many individual constituents when evaluated against a gold standard. This provides evidence in favor of our general account of the distributional differences between arguments and modifiers. It also suggests a kind of lower bound on the amount of information that a suitably equipped learner could use to identify which phrases are arguments or modifiers.

Keywords: linguistics, machine learning, computational linguistics, syntax, statistics

We then present a series of analyses investigating how and why the argument-modifier model is able to recover the argument-modifier status of some constituents. In particular, we show that the argument-modifier model is able to provide a simpler description of the input corpus than the argument-only model, both in terms of lexicon size, and in terms of the complexity of individual derivations. Intuitively, the argument-modifier model is able to do this because it is able to ignore spurious modifier structure when learning the lexicon. These analyses further support our general account of the differences between arguments and modifiers, as well as our simplicity-based approach to learning.

1 INTRODUCTION

The expressivity of natural language is made possible by a division of labor between an inventory of stored items (e.g., morphemes, words, idioms, etc.), known as the *lexicon*, and a set of structure-building operations which combine lexical items to create new expressions, known as the *grammar*.[1] The operation of the grammatical system is highly constrained by requirements imposed by specific lexical items. Consider the verb *put*. In its most basic usage, this verb can only appear in sentences which contain constituents expressing: (i) who is doing the putting, (ii) what is being put, and (iii) the destination of the putting event. The sentence **John put the loaf of bread* is incomplete, while the sentence *John put the loaf of bread in his kitchen cupboard* is not. Furthermore, *put* imposes other requirements on sentence structure, such as the requirement that object being put be expressed as a noun phrase. We will refer to such lexically-specified requirements as the *argument structure* of *put*.

[1] Note that throughout this paper, we use the term *lexical item* to refer to the elementary units combined by a grammar formalism – whether or not they contain surface words. In the tree-adjoining grammar tradition, which we make use of here, these would more formally be called *elementary trees*. Hence, whenever we use the term lexical item, we are referring to what are typically referred to as elementary trees in that literature.

Over the last decades, many linguistic theories have adopted a *lexically-driven* view of grammar. Under such an architecture, grammatical computation is performed by small number of structure-building operations (e.g., UNIFY, MERGE, etc.) whose behavior is controlled by the argument-structure specifications of lexical items (Bresnan 2001; Chomsky 1995a,b; Culicover and Jackendoff 2005; Gamut 1991; Gazdar *et al.* 1985; Heim and Kratzer 1998; Huddleston and Pullum 2002; Jackendoff 2002; Johnson and Postal 1980; McConnell-Ginet and Chierchia 2000; Mel'čuk 1988; Moortgat 1997; Pollard and Sag 1994; Sag 2012; Stabler 1997; Steedman 2000).[2] The development of lexically-driven approaches to grammar leads naturally to the suggestion that much of language learning might be reduced to the problem of learning the lexicon (see, e.g., Chomsky 1993).

However, natural language also exhibits constituents that do not appear to be arguments of any lexical item. Consider the sentence *While preparing dinner, John thoughtlessly put the loaf of bread in his kitchen cupboard*. In this sentence, the phrases *while preparing dinner* and *thoughtlessly* specify additional information about the time and manner of the putting event, but they do not seem to be required by any other constituent and the sentence is well-formed and interpretable without them. These phrases also differ in a number of other ways from the core arguments of the verb. For instance, while the argument-phrase specifying the doer of the putting event (i.e., *John*) must appear in the subject position of the sentence (**put the loaf of bread John in his kitchen cupboard*), these other phrases can appear in a greater variety of positions (*John thoughtlessly put the loaf of bread in his kitchen cupboard, while preparing dinner*). We will refer to such non-argument phrases as *modifiers*.

The existence of such (apparent) non-argument-driven structure raises a fundamental question. If there are both lexical and non-lexical

[2] We note that an alternate tradition of *constructivist* theories argue that argument structure is not associated with particular lexical roots (a position sometimes known as *projectivism*) but rather is a consequence of the functional structure into which roots are inserted during syntactic derivation (see Marantz 2013, for discussion). To the degree that differences between arguments and modifiers in such frameworks still give rise to the distributional differences we discuss below, our results are also consistent with these theories.

modes of composition, how do learners determine when and how each are used? Consider the phrase *in his kitchen*. In the sentence *John put the loaf of bread in his kitchen*, this phrase is an argument, while it is a modifier in the sentence *John made the loaf of bread in his kitchen*. Adult speakers understand these structural differences despite such superficial similarities between the constructions. How do they come by this knowledge?

In this paper, we use computational modeling to address this question. We argue that the statistics of natural language corpora provide evidence that would allow learners to distinguish between argument and non-argument modes of composition in many cases. This evidence is complementary to other forms of evidence available to learners that have been discussed in the context of the argument-modifier distinction in the linguistic literature (such as semantic differences) and can be leveraged by appropriately equipped learners to determine the *argument* or *modifier* status of individual phrases.

In Section 2, we propose that modifiers tend to differ from lexically specified arguments in three ways that have distributional consequences (*inter alia*): **iterability** vs. **finiteness**, **optionality** vs. **obligatoriness**, and **structural flexibility** vs. **structural fixity**. In Section 2.1, we describe two models of lexicon learning designed to minimally capture these differences: the *argument-only model* and the *argument-modifier model*. All formal details can be found in the appendices of the paper.

In Section 3, we describe how lexicon learning under both models can be formulated in terms of a *tradeoff* between two simplicity biases that favor small lexica (*simple-lexicon bias*) and simple derivations (*simple-derivation bias*), respectively. Adopting this tradeoff-based approach, we first show in Section 5.1 that the argument-modifier model is able to recover the argument status of many constituents in a gold-standard corpus, indicating that it captures some aspect of the argument-modifier distinction as discussed in the linguistics literature. We then show in Section 5.2 that the argument-modifier model is able to provide explanations of the input corpus that are more optimal in terms of both the small-lexicon and simple-derivation biases. These results imply that there is clear distributional evidence indicating the argument-modifier status of

many phrases and that this evidence could be leveraged by learners who make use of a tradeoff between derivational and lexical simplicity.

ARGUMENTS AND MODIFIERS 2

Historically, some distinction between arguments and modifiers (sometimes called *adjuncts*) has been assumed by nearly all theories of syntax and semantics and a number of theoretical mechanisms have been proposed to handle the distinction. Furthermore, many different syntactic and semantic tests have been proposed for distinguishing between the two kinds of phrase (see Bergen *et al.* 2015, for detailed review of this literature). In this paper, we operationalize the argument-modifier distinction by focusing on one particular question: Which constituents in a sentence are there because they were required by some lexical item, and which are not lexically required? In this paper, *argument structure* will refer to any lexically-specified constraint or requirement on constituent co-occurrence. We intend this general notion of argument structure to potentially include many kinds of lexically-specified constraint that have been proposed over the years in different grammatical traditions. Thus, it includes verb-argument structure but, also, the lexical requirements of other categories such as prepositions or nouns.

The difference between arguments and modifiers is often cast in semantic terms. While we do not deny that there are important differences in the way that these types of constituent contribute to the meaning of sentences, in this paper we focus solely on differences between the two types of phrase that affect the distribution of constituents in language.

In lexically-specified grammar formalisms, lexical items list their arguments and (typically) where these arguments appear with respect to the selecting item. This architecture has three critical properties which have important distributional consequences. First, lexical items in such formalisms usually specify only a small number of argument

positions (**finiteness**).[3] Second, lexical arguments are typically obligatory in such systems (**obligatoriness**), though some mechanisms for handling optional arguments are usually provided. Third and finally, particular arguments are required to appear in fixed relationship to the selecting lexical item (**structural fixity**). In languages like English, this typically corresponds to their structural position with respect to their selecting head. In other languages, this may correspond to a grammatical relation which is encoded in other ways (e.g., case).

By contrast, the types of constituents which have been traditionally identified as modifiers differ in each of these three properties. An unbounded number of modifiers can often be added to a constituent (**finiteness** vs. **iterability**); modifiers tend to be optional (**obligatoriness** vs. **optionality**); and modifiers often occur in a greater variety of structural relationships with their head (**structural fixity** vs. **structural flexibility**). These three dimensions of variation summarize a large number of properties and linguistic tests that have been discussed in the literature (Borsley 1999; Comrie 1993; Creissels 2014; Croft 2001; Forker 2014; Haegeman 1994; Haspelmath 2014; Hornstein and Lightfoot 1981; Koenig *et al.* 2003; Kroeger 2004; Matthews 1981; Przepiórkowski 1999a,b; Radford 1988; Rákosi 2006; Schütze 1995; Schütze and Gibson 1999; Tallerman 2015; Tutunjian and Boland 2008; Vater 1978; Wichmann 2014; Zwicky 1993).[4]

We emphasize that these properties are not definitional and do not represent necessary and sufficient conditions on argumenthood. Instead, they are tendencies: Arguments are sometimes optional (e.g., *John ate/John ate the cake*) and in some cases there is more than one structural realization of the same arguments of some lexical item (e.g., *John gave Mary the book/John gave the book to Mary*). At the same time, there are often strong constraints on the structural position of modifiers (e.g., *John gave Mary the book quickly/*John gave quickly Mary the book*) and there are constructions in which modifiers are obligatory (e.g., *These ovens clean easily*).[5] Nevertheless, the three properties do roughly summarize a number of linguistic tests for argument-/modifierhood often dis-

[3] See Przepiórkowski (2017) for an exception.

[4] See Bergen *et al.* (2015) for a detailed review of this literature.

[5] We thank an anonymous reviewer for this example.

cussed in the literature. We propose that these statistical tendencies can be used by suitably equipped learning to determine the argument/modifier status of many constituents and, thus, provide a useful source of evidence for lexicon learning that is complementary to other sources of evidence that have been discussed in the literature.

Tree-substitution and sister-adjunction grammars 2.1

In this paper, we use *probabilistic tree-substitution grammars* as our model of lexical argument structure. A tree-substitution grammar formalizes the lexicon as an inventory of stored tree fragments, such as those shown in Figure 1 (Bod 1998; Joshi and Levy 1975; Scha 1990, 1992). This figure shows the inventory of elementary trees that we will use as examples below.[6] Each tree fragment encodes the category and structural position of argument phrases that must be present in a complete sentence which is derived using the fragment. In a tree-substitution grammar, lexical fragments are combined via the SUBSTITUTE operation, which replaces a node at the frontier of a derivation with another tree fragment from the lexicon – subject to the condition that the category of the frontier node and the root category of the substituted fragment are identical. The SUBSTITUTE operation is applied recursively until no substitutable nodes remain at the frontier, and a complete sentence has been derived.

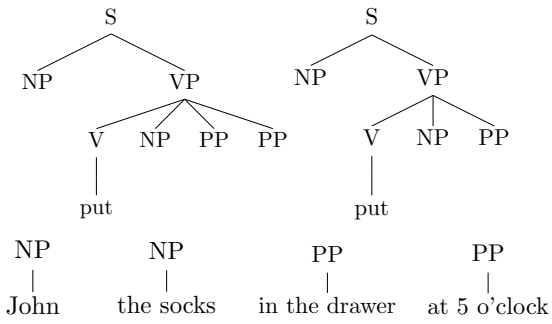

Figure 1:
Inventory of tree fragments

[6] Note that the internal constituent structure of the noun and prepositional phrases (NP and PP) has been suppressed.

Tree-substitution grammars capture the three core properties of argument structure discussed Section 2. Each lexical fragment can only possess a fixed (and in practice small) number of leaf variables (**finiteness**). All such variables must be filled in a complete derivation (**obligatoriness**); and finally, the position of each argument phrase is fixed relative to the lexical item which selects for it (**structural fixity**).

To model modification, we make use of an extension of tree-substitution grammars which introduces a second structure-building operation, *sister-adjunction* (Chiang 2000; Chiang and Bikel 2002; Rambow *et al.* 1995; Schabes and Shieber 1994). While SUBSTITUTE must be licensed by the presence of an argument node at the frontier, SISTER-ADJOIN can insert a constituent as the sister to any node in an existing tree. The formalism is strongly equivalent to (unlexicalized) tree-insertion grammar and, therefore, has the same weak generative capacity as context-free grammar (Schabes and Waters 1995).[7]

To derive the complete tree for a sentence using a set of fragments such as those shown in Figure 1, the generative process starts from a single nonterminal node of category S (i.e., the start symbol), and then recursively samples arguments and modifiers according to the following procedure. For each node f with nonterminal category A on the frontier of our derivation, we perform the following two steps. First, we choose an elementary tree t with category A from our lexicon and, for each position before or after a node on the interior of t, we sister-adjoin zero or more new nonterminal nodes, representing modifier phrases. Second, we substitute f – now with modifier category nodes – into the derivation at node n (see discussion of Figure 2 below). This process then repeats on any nonterminal nodes now on the frontier of the tree. In particular, if we have sister-adjoined a modifier node with category X, its internal structure will be determined recursively by choosing an elementary tree of category X from the lexicon.

The SISTER-ADJOIN operation formalizes the three core ways in which modifiers differ from arguments: (i) The decision to insert or not insert a modifier does not change the well-formedness of a generated structure with respect to the satisfaction of lexical argument

[7] We note that these formalisms have different strong generative capacity, however.

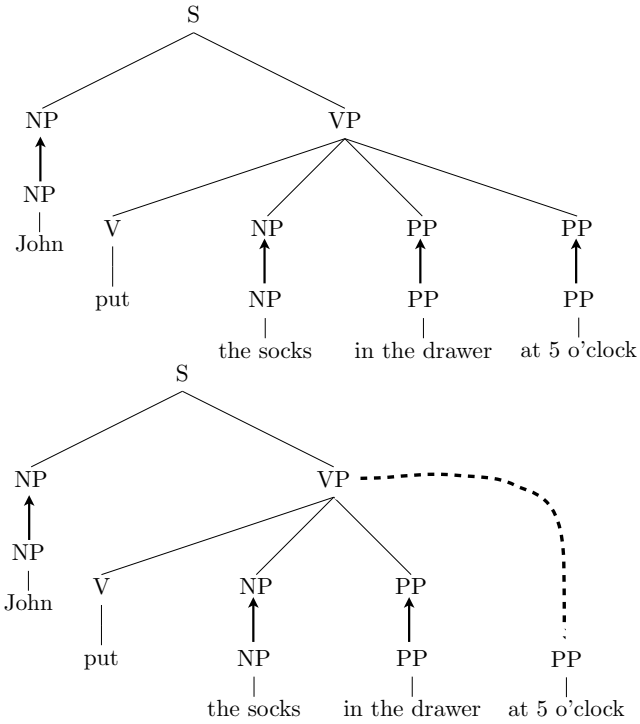

Figure 2:
TSG versus SAG derivations

requirements (**optionality**) (ii) SISTER-ADJOIN can insert any number of modifiers at a position in a derivation (**iterability**), and (iii) SISTER-ADJOIN can insert a modifier at any position in a constituent (**structural flexibility**).

Figure 2 illustrates two derivations of the same tree, one in a standard tree-substitution grammar (TSG) without sister-adjunction, and one in the model extended with SISTER-ADJOIN, which we term *sister-adjunction grammar* (SAG). The tree-substitution grammar derivation, at the top of the figure, uses an elementary tree with four leaf nonterminals as the backbone for the derivation. The four phrases filling these arguments are then substituted into the elementary tree, as indicated by arrows. Note that in tree-substitution grammars the prepositional phrase, *at 5 o'clock*, which is a temporal modifier, enters the derivation through an argument node. However, the sister-adjunction grammar in the lower part of the figure is able to insert this modifier using SISTER-ADJOIN (indicated using dotted lines) and, therefore, uses an elementary tree with only three leaf nonterminals as the back-

bone of this derivation. This difference will mean that tree-substitution grammars will require a greater number of tree fragments in the lexicon to account for variability that could otherwise be accounted for using modification.

3 HANDLING UNCERTAINTY: TRADEOFF-BASED LEARNING OF LEXICA

Neither language learners nor linguists have a priori knowledge of the set of lexical items in a language, their particular argument structures, or the argument/modifier status of individual phrases in the input. Rather, the set of lexical argument structures in a language must be learned from linguistic input, and the derivation of particular sentences must be inferred on a case-by-case basis. In this paper, we adopt a probabilistic approach to these problems of learning and inference, specifying prior distributions over lexicons and derivations for both the argument-only model and the argument-modifier model, and using probabilistic conditioning to infer language-specific lexicons and utterance-specific derivations from input data. We give formal definitions of our prior distributions, and algorithms for estimating conditional probabilities in Section 6. In this section, we give an intuitive overview of the ideas behind the framework.

Following earlier work, we propose that lexicon learning is guided by two prior biases for simplicity (especially Brent 1999; De Marcken 1996a,b; Goldwater 2006; Johnson *et al.* 2007; O'Donnell 2011, 2015). The first, the *simple lexicon bias*, provides an a priori measure of the quality of lexicons, favoring those with fewer, more reusable lexical items. The second, the *simple derivation bias*, provides an a priori measure of the quality of the derivations of individual sentences, favoring simpler derivations involving smaller numbers of lexical items, and lexical items with higher probability. These two biases lead to a tradeoff: For a fixed set of sentences, if we increase the average reusability of lexical items, then we must also increase the average number of lexical items used in any derivation. Likewise, if we decrease the average number of lexical items used per derivation, we must, on average, increase the size of the lexicon. The inference problem is to jointly find a

set of lexical items and sentence derivations that best explains the distribution of forms in the input data, subject to these two prior biases.

Our two prior biases are a special case of the standard Bayesian prior/likelihood tradeoff applied to the problem of lexical storage. The preference for more reusable lexical items is encoded by the prior over lexical items and the preference for smaller derivations results from the likelihood, which favors derivations in which fewer random choices are made. In the two sections below, we provide additional intuitions about the behavior of our models when applied to input datasets and details about their implementation.

3.1 *Simplicity biases and inference*

As just discussed, our models encode two simplicity biases. The *simple lexicon bias* favors smaller lexicons containing more reusable lexical items. Following Goldwater (2006), Johnson *et al.* (2007), and others, we formalize this bias using a distribution from Bayesian nonparametric statistics known as the *Pitman-Yor Process* (Pitman and Yor 1995). A Pitman-Yor process $\mathrm{PYP}(G_0, a, b)$ is a distribution over lexical items that is specified with three parameters, G_0, a, and b. The first parameter, G_0, is a prior distribution over possible tree fragments that can be stored as lexical items. The other two parameters – known as the concentration parameter b and discount parameter a – are real-valued such that $0 \geq a \geq 1$ and $b > -a$.

A Pitman-Yor process operates as follows. The first time we sample from $\mathrm{PYP}(G_0, a, b)$, a new lexical item will be chosen according to G_0, stored internally by the Pitman-Yor process, and returned to the caller. On subsequent invocations, either a previously sampled lexical item i will be returned with probability $\frac{n_i - a}{N + b}$, or a new lexical item will be sampled from G_0, stored, and returned, with probability $\frac{aK + b}{N + b}$, where n_i is the number of times that lexical item i was previously sampled, N is the total number of lexical items sampled so far (i.e., $N = \sum_j n_j$), and K is the number of distinct lexical items that have been previously sampled (i.e., the number of lexical *types*). Notice that these definitions favor smaller numbers of lexical items and induce a rich-get-richer dynamic whereby lexical items that are used more often are more likely to be reused.

The *simple derivation bias* favors derivations for individual sentences that use small numbers of more probable lexical items. In both the argument-only model and argument-modifier model, this bias is captured by our assumption that the probability of a derivation is the product of the probabilities of the lexical tree fragments used to construct it. Because probabilities must be numbers between 0 and 1, the probability of a derivation decreases quickly (geometrically) as the number of fragments it contains increases. However, this can be mitigated somewhat if the fragments are highly probable (i.e., have probability close to 1).

Applying these two simplicity biases to tree-substitution grammar, we arrive at what we call the argument-only model (see Bod *et al.* 2003; Cohn *et al.* 2010; O'Donnell 2011, 2015; Post and Gildea 2013, for related models). To better understand the inferential behavior of the argument-only model, it is useful to consider a toy example. Figure 3 shows three possible solutions to the problem of inferring the correct set of stored tree fragments for a toy corpus consisting of three sentences.

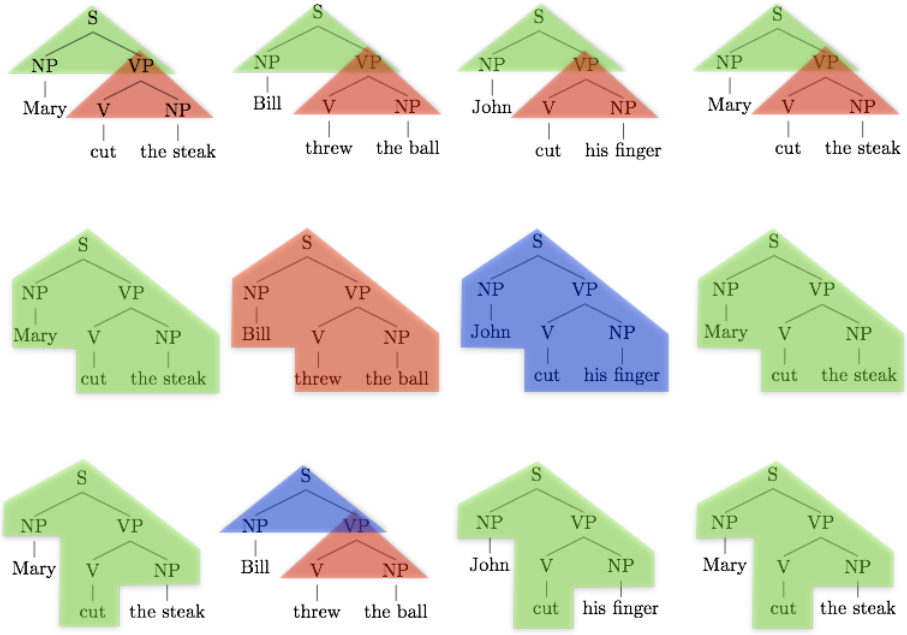

Figure 3: Inference in the argument-only model

Row I of Figure 3 shows the result of storing and using only the smallest, most abstract fragments of sentence structure. In this case, each particular lexical item will be highly reusable, and the lexicon will be maximally compact. However, the derivations of individual sentences will necessarily make use of many lexical fragments and will therefore be more complex. Row II of the figure shows the solution at the other extreme. In this case, every utterance is stored in its entirety. This solution will result in extremely large lexicons with lexical items of limited reusability. However, individual sentences which recur in the data will be derivable with a single lexical item, resulting in potentially low-cost derivations if particular sentences recur in the input. Row III of Figure 3 shows an intermediate solution which is more optimal with respect to this dataset. By storing lexical fragments which express argument structures of intermediate complexity, this solution produces a more compact lexicon than the solution in Row II, and simpler derivations than the solution in Row I, providing a globally better explanation of the input forms. The inference problem solved by the argument-only model is to find such optimal sets of tree fragments given an input corpus.

A similar pair of simplicity biases is used to define the distribution over modifiers. Recall that SISTER-ADJOIN inserts modifier category nodes into derivations and that these nodes are then filled using SUBSTITUTE. We place a Pitman-Yor process prior over the set of possible modifier node categories. This prior will bias the model towards using a small set of category types when sampling modifiers. For example, the modifier model might prefer to hypothesize that only adjective and adverb phrases are likely to be modifiers rather than adjective, adverb, noun, and verb phrases. A second simplicity bias favors inserting only a small number of modifiers into derivations. This bias is captured by the assumption that the probability of deriving a sequence of modifiers is the product of probabilities of the individual modifiers in this sequence. Because this product drops off geometrically in the number of modifiers, the model will prefer derivations which contain a small number of modifiers.

Applying all of the simplicity biases to sister-adjunction grammar, we arrive at the argument-modifier model. During inference, the argument-modifier model will attempt to find an optimal set of reusable argument-structure fragments by categorizing individual

nodes in input data trees as either (i) internal to a stored tree fragment, (ii) built by substitution of a lexical item at a frontier node, or (iii) built by sister-adjunction. In general, the model will categorize a node as a modifier when doing so will result in a simpler representation of the input corpus, that is, when it allows the input corpus to be explained using a smaller set of lexical items. Intuitively, the SISTER-ADJOIN operation allows the model to prune out constituents when doing so will lead to more compact and generalizable lexical items.

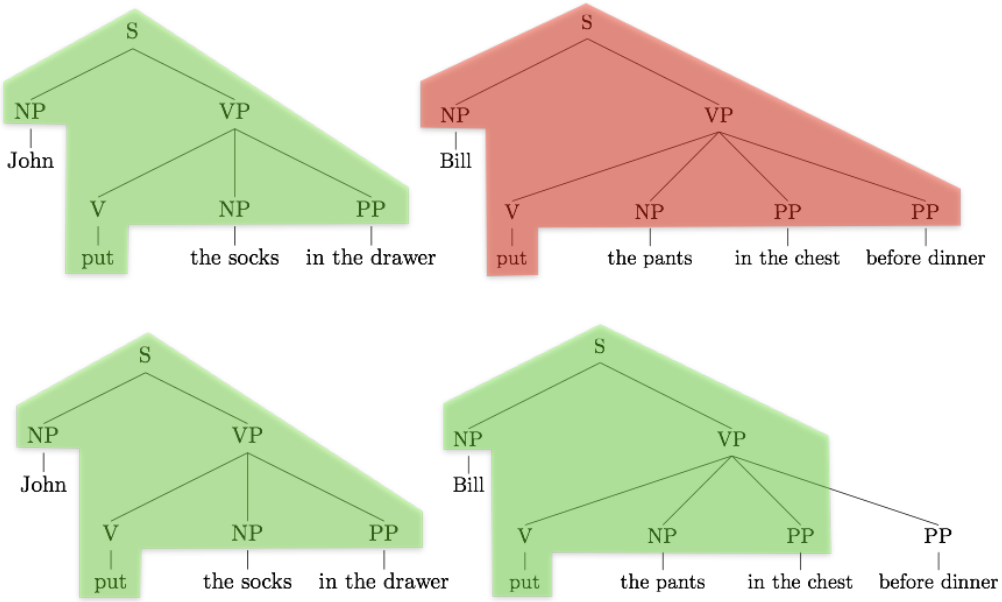

Figure 4: The argument-only model versus the argument-modifier model

Consider Figure 4. If a model posits that there are no modifiers in these sentences, then it will not identify the shared structure between two uses of the verb *put*, and will derive them using distinct sets of elementary trees, as on the top line of Figure 4. On the other hand, if it posits that the PP *before dinner* is a modifier, then it will be able to derive the core structure of these sentences using a single elementary tree, as on the bottom line of Figure 4. Nodes will be identified as modifiers when, like this PP, their removal from the sentence's argument structure leads to simpler derivations of the sentences in the corpus and greater amounts of sharing in the lexicon.

Inference 3.2

To perform inference, we developed a local Gibbs sampler which generalizes the one proposed by Cohn *et al.* (2010). This sampler jointly explores the space of elementary trees and substitution/adjunction attributions for a corpus consisting of parsed sentences. At each iteration, the sampler determines for each node in the corpus whether (i) the node is internal to an elementary tree, (ii) the node is the root of a tree which was inserted by substitution, or (iii) the node is the root of a tree which was inserted by sister-adjunction. The sampler randomly selects a node in the corpus and resamples its label from the full conditional posterior given the current hypothesis for the rest of the nodes in the corpus and the elementary tree set.

SIMPLICITY AND EVALUATION METRICS 4

Before presenting our results in the next section, we make some observations about the relationship between our learning framework and the broader literature. The tradeoff-based approach that we adopt here can be understood as an instantiation of the classical linguistic notion of an *evaluation metric* (Chomsky 1951 [1979], 1955 [1975], 1964).[8] Although we make use of probability theory to capture our two kinds of simplicity, our framework is closely related to other approaches that operationalize simplicity using the idea of description-length or succinctness (e.g., Berwick 1982, 1985; Brent 1997, 1999; Cartwright and Brent 1994; De Marcken 1996a,b; Grünwald 2007; Hsu and Chater 2010; Hsu *et al.* 2011, 2013; Li and Vitányi 2008; Perfors *et al.* 2011; Phillips and Pearl 2014; Rissanen 1978; Stolcke and Omohundro 1994; Wolff 1977, 1980, 1982; Yang and Piantadosi 2022, inter alia). Such approaches go back at least to Chomsky (1951 [1979]) in linguistics and have been widely discussed in philosophy, statistics, and cognitive science, both with respect to their normative justification as well as their appropriateness for describing human psychology (see, Li and

[8] Also see discussion in Goldsmith (2011) and Rasin and Katzir (2016).

Vitányi 2008, for an overview of many historical threads in statistics, mathematics, and computer science).

Perhaps the most general treatment comes from the theory of Solomonoff induction, which uses a distribution over the set of all possible computer programs to define simplicity preferences related to those used in this work (Grünwald 2007; Li and Vitányi 2008; Rissanen 1978; Solomonoff 1978, 1964a,b). In this framework, theories (i.e. computable distributions over observations) are preferred when they are both simple to describe and provide simple descriptions of the data. It has been proven that this distribution can be used to asymptotically learn any computable theory, given a sufficient amount of data, and as a result it has been proposed as a universal, normative account of learning. The relation between this work and theoretical and empirical problems of language learning are also beginning to be understood in more detail (see, e.g., Hsu and Chater 2010; Hsu *et al.* 2011, 2013, for recent discussion). In cognitive science, there is a large and growing body of work suggesting that human inductive biases are captured by models making use of similar simplicity biases (see, e.g., Feldman 2000; Goodman *et al.* 2008; Piantadosi 2011, 2021, for examples from concept learning).

However, any formal definition of simplicity is dependent on the formalism, representation, or machine model with respect to which it is defined (Li and Vitányi 2008). Therefore, proposals about simplicity are substantive parts of theories of learning and must be evaluated together with other aspects of such theories. There remain several different frameworks implementing the simplicity-based approach – including the Bayesian framework, adopted here, and the minimum description length framework (Grünwald 2007; Rissanen 1978). It remains for future work to achieve a more fine-grained theoretical and empirical understanding of similarities and differences amongst various approaches to learning-via-simplicity.

5 SIMULATIONS

In this section, we will use the computational models introduced above to evaluate two questions. First, do the statistics of natural language corpora provide evidence for the argument or modifier status of indi-

vidual phrases that is usable by a learner that optimizes a tradeoff between lexical and derivational simplicity? Second, why is the argument-modifier model a superior model of the input data under these simplicity biases.

In order to address these questions, we will perform two sets of analyses. In the first, we will look at whether the argument-modifier model learns a distinction between arguments and non-arguments which agrees with a hand-annotated corpus. We will show that the argument-modifier model classifies arguments and non-arguments in a manner that aligns with traditional linguistic assumptions. Thus, we conclude that the argument/modifier status of individual phrases is evidenced in the input. This provides evidence in favor of both our formalization of the distinction and in favor of tradeoff-based learning.

In the second set of analyses, we will examine how the argument-modifier model infers the argument status of constituents using simplicity. We will show that the argument-modifier model learns a simpler representation of the input data than the argument-only model. We illustrate how this arises as a result of the representational and inferential assumptions discussed above.

5.1 *Gold Standard Evaluation*

Our first set of analyses examine the ability of the argument-modifier model to correctly classify constituents as arguments or modifiers. As we discussed above, the model was designed to capture three differences between arguments and modifiers that affect their syntactic distribution: **obligatoriness/optionality**, **finiteness/iterability**, and **fixity/flexibility**. If the argument-modifier model is able to correctly distinguish modifier and argument phrases in the training corpus, we can conclude that these three distributional differences provide a signal to appropriately equipped learners.

We trained the argument-modifier model on sections 2–21 of the Wall Street Journal portion of the Penn Treebank (Marcus *et al.* 1999). The input consisted of approximately 40,000 parsed sentences, without any further annotations for argument or modifier status. Under the Penn Treebank's tree annotation scheme, arguments and modifiers are not distinguished from each other by their hierarchical relations

in the parse tree (or in any other way). In particular, the arguments and modifiers of a phrase are most often siblings in the tree. Thus, the argument-modifier model could not directly use any information in the input corpus to simply read off each sentence's argument and modifier structure.

In order to evaluate the accuracy of the argument-modifier model classification of arguments and modifiers, we require a gold standard which provides annotations for arguments and modifiers in the Penn Treebank. Unfortunately, no such resource provides a classification of all nodes in the Penn Treebank (or CHILDES, MacWhinney 2000, which we use in our next study). However, for a subset of the phrases in the Penn Treebank, such information is available in the PropBank corpus (Palmer *et al.* 2005) which provides annotations of argument and modifier structure for all of the verbal predicates in the Wall Street Journal portion of the corpus. As noted in Palmer *et al.* (2005), the annotation of modifiers in PropBank is non-standard in certain cases. In particular, `NEG` and `MOD` categories are annotated as modifiers. We therefore exclude these categories from our analyses. PropBank does not annotate the arguments or modifiers of expressions which are not verbal predicates. Our model evaluations were performed by running the Gibbs sampler described in Section 3.2 for 100 iterations, and selecting the node labelings which were output on the final iteration.

For the purpose of our analyses, all sister-adjoined nodes are classified as modifiers, and all other nodes (i.e. nodes which are internal to an elementary tree or at the leaf of one) are classified as non-modifiers. We compared the model's labels to those provided by PropBank, on the subset of nodes for which PropBank provides annotations.

To show that differences in the distributions of argument and modifier phrases provide a valuable source of evidence for lexicon acquisition, we first establish that our model is able to correctly classify phrases at a rate which is better than chance. To demonstrate this, we computed the precision (i.e. number of correctly identified modifier nodes divided by the total number of modifier nodes identified by the model) and recall (i.e. number of correctly identified modifier nodes divided by the total number of modifier nodes in the gold-standard) of the model and compared it with two baselines. The first baseline randomly classifies each node as internal to an elementary tree, the leaf of an elementary tree, or a modifier with equal probability. Note

that prior to receiving any training data, the model has no information about which phrase types are likely to be modifiers and which are likely to be arguments. The random baseline therefore represents the model's knowledge of the argument/modifier distinction prior to training, and any improvement in the model's classification of modifiers must be attributed to information contained in the input data.

The second baseline treats every node as a modifier. We introduce this baseline in order to illustrate some basic facts about PropBank. Table 1 shows precision and recall in identifying the modifiers of verbal predicates in the corpus. The argument-modifier model is compared to three baselines: an all-modifier baseline, in which every node is labeled as a modifier, a random baseline, and a version of the model that does not use context to predict modifiers. PropBank annotated 179,058 nodes in the corpus for their argument/modifier status. These nodes represent approximately 10% of the total nodes in the corpus. Among the annotated nodes, 45,507 (25%) are modifiers, meaning that 25% of the guesses of the all-modifier baseline are correct.

Precision measures accuracy of modifier-predictions. Table 1 shows that the argument-modifier model is significantly more accurate than the random and the all-modifier baselines, demonstrating that the training data has provided information which allows the model to correctly classify many constituents.

Model	Precision	Recall	Accuracy
all-modifier	0.25	1	0.25
all-argument	N/A	0	0.75
random	0.29	0.23	0.66
SAG	**0.66**	**0.52**	**0.81**

Table 1:
Precision and recall
of the argument-modifier model

Recall measures the coverage of gold-standard modifier nodes achieved by the models. Again, the argument-modifier model achieved significantly higher coverage than the random baseline, indicating that the training data contains enough information to increase the number of true modifiers that the model recognizes.

In order to better understand what the argument-modifier model learned about the modifiers of verbal predicates, the evaluations against PropBank were further broken down by the category of the

Table 2: Labelings for modifiers of VP nodes, broken down by child category

VP Parent				
Child category	Model	Precision	Recall	PropBank
ADVP	random	0.95	0.23	12,385
ADVP	SAG	0.95	0.47	12,385
NP	random	0.04	0.23	3,345
NP	SAG	0.47	0.57	3,345
PP	random	0.49	0.22	18,841
PP	SAG	0.56	0.54	18,841
SBAR	random	0.40	0.22	4,552
SBAR	SAG	0.84	0.63	4,552

modifier. Table 2 shows the results for the phrase types which occur most frequently as verbal modifiers: adverb phrases (ADVPs), noun phrases (NPs), prepositional phrases (PPs), and subordinate clauses (SBARs). Together these categories of constituent account for more than 85% of the modifiers in the training corpus.

For the phrase categories of adverb phrases (ADVPs) and prepositional phrases (PPs), the model doubles the recall of the random baseline, and roughly maintains its baseline precision. Adverb phrases are typical modifiers when they appear within a verb phrase (VP). Out of 13,197 ADVPs annotated by PropBank, 12,384 are modifiers. Prepositional phrases are also frequently modifiers when they appear in this context. Out of 38,861 PPs annotated by PropBank, 18,839 are modifiers. The increase in the model's recall therefore indicates that the model learned to correctly classify many of these ADVP and PP modifiers.

In contrast to adverb and prepositional phrases, noun phrases (NPs) which appear within verb phrases are typically arguments to the verb. Out of 92,965 NPs annotated by PropBank, only 3,306 appear as modifiers. Exceptions to this generalization are cases where a noun phrase is used as an adverbial modifier, such as the noun phrase *last night* in *They played the game last night*. The precision of the model increased by a factor of 10 for NPs, indicating that it incorrectly classified many fewer non-modifier NPs. In addition, the model's precision more than doubles the baseline.

Phrases belonging to the category of subordinate clauses SBAR can serve either as arguments or modifiers. For example, in the sentence

John said that he would be late, the subordinate clause *that he would be late* is an argument of the verb *said*. By contrast, in the sentence *The woman laughed when she heard the joke*, the clause *when she heard the joke* is a temporal modifier of the verb *laughed*. Out of 13,617 SBAR phrases annotated by PropBank, 4,551 are modifiers. The model's precision and recall on SBAR phrases was more than twice that of the random baseline, showing that the model classified fewer clausal arguments as modifiers, and correctly identified a greater number of clausal modifiers.

As we mentioned above, certain categories of constituents have highly stereotyped argument-modifier status when they appear as children of other categories. For example, adverb phrase (ADVP) children of verb phrases (VP) and adjective phrase (JJ) children of noun phrases (NP) are both typically modifiers of their parent constituents.

PropBank only provides argument-modifier annotations for the children of verb phrases (VP), and therefore we do not have a gold standard for modifiers occurring outside of VPs. Nonetheless, it is possible to use the stereotyped behavior of these categories to examine the model's performance on the children of non-VP nodes. Tables 3–6 show the model's classification of constituents which were children of sentence-level constituents (S), prepositional phrases (PPs), noun phrases (NPs), and subordinate clauses (SBARs), respectively. In each of these cases, the category of child constituents is highly indicative of their argument-modifier status.

For sentence-level (S) constituents, we analyzed three categories of child phrase: noun phrases (NPs), verb phrases (VPs), and (ADVPs). These are the three most common categories which have stereotyped argument/modifier behavior when they appear as children of nodes. Of these three phrase types, noun and verb phrase are typically not modifiers, whereas adverb phrases typically are. For example, in *Usually, John wears a coat*, the adverb *Usually* is a modifier of the sentence while *John* and *wears a coat* are not modifiers. Table 3 shows how often the model labeled the children of sentence-level constituents as modifiers nodes. The model accords with intuition here, most often labeling adverb phrases but not noun or verb phrases as modifiers.

For prepositional phrases (PPs), we considered four categories of child constituent: adverb phrases (ADVPs), noun phrases (NPs), prepositions (INs), and *to* (TOs). Of these phrase types, only adverb phrases

Table 3: Labels for children of S nodes

S parent				
Child category	Model	#Guessed	Corpus total	Typically modifier
ADVP	random	1,393	6,063	Y
ADVP	SAG	2,331	6,063	Y
NP	random	16,654	93,076	N
NP	SAG	1,738	93,076	N
VP	random	16,005	89,984	N
VP	SAG	572	89,984	N

typically modify the parent prepositional phrase. For example, in the prepositional phrase *immediately after the opening*, the adverb phrase *immediately* is a modifier while the prepositional head *after* and noun phrase *the opening* are not. In accord with these intuitions, Table 4 demonstrates that the model classifies most adverb phrase children of prepositional phrases as modifiers, but treats prepositional heads and noun phrases as non-modifiers.

Table 4: Labels for children of PP nodes

PP parent				
Child category	Model	#Guessed	Corpus total	Typically modifier
ADVP	random	216	1,109	Y
ADVP	SAG	547	1,109	Y
IN	random	13,972	83,848	N
IN	SAG	672	83,848	N
NP	random	15,060	88,556	N
NP	SAG	496	88,556	N
TO	random	1,484	8,654	N
TO	SAG	64	8,654	N

We considered four categories of subconstituents for noun phrases (NPs): determiners (e.g., *the*, *a*; DT), adjectives (JJ), other noun phrases, and prepositional phrases. Determiners are unlikely to modify noun phrases, while adjectives typically do modify them. For example, in the noun phrase *the big chair*, the determiner *the* is not a modifier, while the adjective *big* modifies the noun *chair*. Prepositional phrases are often modifiers (e.g., in *the resort by the sea*, the prepositional phrase *by the sea* modifies the noun *resort*), although in some cases,

Table 5: Labels for children of NP nodes

NP parent				
Child category	Model	#Guessed	Corpus total	Typically modifier
DT	random	15,791	77,553	N
DT	SAG	1,701	77,553	N
JJ	random	10,544	45,812	Y
JJ	SAG	9,717	45,812	Y
PP	random	7,652	43,420	Y
PP	SAG	3,226	43,420	Y

such as deverbal nominalizations, they are typically treated as arguments of the head noun (e.g., in the noun phrase *the destruction of the city*, the prepositional phrase *of the city* is an argument of the head noun; see, e.g., Chomsky 1970).

Table 5 shows the modifier-classification rates of noun phrase children. The model correctly identifies determiners as non-modifiers. However, for adjectives, the most prototypical modifiers of noun phrase, the model's performance is weaker: The number of JJs classified as modifiers is approximately the same as the random baseline. The number of PPs classified as modifiers decreased by more than half relative to the random baseline, though the implications of this are unclear: As discussed above, PPs appear frequently as the modifiers of noun phrases but also as arguments. It should be noted that the Penn treebank is notorious for having many complex noun phrases consisting of long sequences of noun compounds annotated with a single flat structure. This likely affected the ability of the model to distinguish amongst the children of NP nodes.

Table 6: Labels for children of SBAR nodes

SBAR parent				
Child category	Model	#Guessed	Corpus total	Typically modifier
S	random	4,873	29396	N
S	SAG	101	29396	N
WHADVP	random	421	2521	N
WHADVP	SAG	38	2521	N
WHNP	random	1,383	8505	N
WHNP	SAG	79	8505	N

The category SBAR is used to mark subordinate clauses in the Penn treebank. Here we consider the following categories of children: sentence-level constituents (Ss) and wh-expressions (WHADVPs and WHNPs) which are used to introduce subordinate clauses (e.g., the word *when* in the sentence *The woman laughed when she heard the joke*). None of these types of constituent is typically thought of as modifying subordinate clauses. Table 6 shows, consistent with this intuition, that the model treats all three categories as non-modifiers.

5.1.1 Discussion

We have presented two sets of results in this section. First, we have shown that the argument-modifier model's accuracy at classifying arguments and non-arguments substantially improves over a random baseline. Second, we have shown that among phrases that are not labeled in the gold standard (i.e., all phrase types but verb phrases), the argument-modifier model learns an argument/non-argument classification which appears linguistically reasonable for most major phrasal categories.

These results have two consequences for the arguments in this paper. The argument-modifier model is built on the assumption of three distributional differences between lexical argument-structure-derived phrases and modifier phrases: **finiteness** v. **iterability**, **obligatoriness** v. **optionality**, and **structural fixity** v. **structural flexibility**. Since the argument-modifier model made use of these properties in order to classify phrases in the input corpus as arguments or non-arguments, its performance on the gold standard shows that we have captured some linguistically relevant properties of arguments and modifiers using these properties.

The results also show that the distributional information contained in the input corpus is often sufficient for recovering the argumenthood of specific constituents. The argument-modifier model does not have any a priori knowledge about which types of phrases are likely to be arguments, and it leverages only distributional information in order to infer the status of individual phrases. Thus, its performance in categorizing arguments and non-arguments must be attributable to the distributional information contained in the corpus. This distributional information is leveraged by the model by trading

off the simple lexicon and simple derivation biases. We note that our study is an example of an *ideal learner analysis* (Pearl and Goldwater 2016); that is, the model is highly idealized and not intended to veridically represent a child language learner. Therefore, our results do not demonstrate that children use distributional information to identify the argument or modifier status of individual constituents. Instead, they indicate that such information would be available to any learner that made similar assumptions about the relationship between simplicity and learning.

5.2 Lexicon learning, arguments structure, and simplicity

In the previous section, we showed that the argument-modifier model is able to correctly recover the modifier status of many constituents using only the pattern of co-occurrences between constituents in the training set. In this section, we show how this performance is the result of the simplicity biases outlined in Section 3. As discussed in that section, our framework makes use of two competing simplicity biases. The *simple lexicon bias* favors small numbers of highly reusable lexical items and the *simple derivation bias* favors derivations of individual forms using small numbers of lexical items. Typically, these two biases lead to a tradeoff. Smaller, more reusable lexical items mean more complex derivations and vice versa. However in this section, we present simulations demonstrating that compared to the argument-only model, the argument-modifier model learns a more compact generalizable lexicon, while also providing simpler derivations for individual forms. If we fix a particular dataset as well as fix a particular model (argument-only model or argument-modifier model and all parameters), there is at least one optimal (i.e., most probable), solution for that model-dataset combination. Any lexicon/derivation set that increases one kind of simplicity with respect to this optimum will necessarily decrease the other. Thus, our results show that the argument-modifier model is overall a better model for the data since it is able to find a solution which is superior under both measures.

To see how it is possible that the argument-modifier model is able to optimize *both* kinds of simplicity simultaneously consider a verb phrase (VP) headed by a verb like *put*. In simplest form, *put* requires

two VP-internal arguments – a noun phrase (NP) expressing the object which was put somewhere, and a prepositional phrase (PP) expressing the destination – *put his socks in the suitcase*. Across particular uses of this simple *put*-construction, the VP node will often have the following sequence of children: V NP PP. However, because modifiers are optional, iterable, and appear at a variety of positions within a constituent, they can greatly increase the number of different observed sequences of children of the VP node: *put his socks suddenly in the suitcase* [V NP ADVP PP], *put his socks in the suitcase suddenly without warning* [V NP PP ADVP PP], etc. The argument-modifier model is able to explain away the presence of these additional phrases using the SISTER-ADJOIN operation, and is driven to do so because this leads to a lexicon of argument-structure fragments and derivations of individual forms which better optimizes both simplicity biases.

In the analyses in this section, we provide empirical support for this argument. To demonstrate the point, we show that the argument-modifier model can account for the same data as the argument-only model with a more compact lexicon and simpler derivations of each sentence. We show this on both training and holdout data drawn from two corpora: the Wall Street Journal portion of the Penn Treebank, and the Brown (1973) portion of the CHILDES database (MacWhinney 2000). For the WSJ, the model was trained on the 40,000 parsed sentences from sections 2-21 (the same sentences that were used in the gold standard analyses). The CHILDES sections used here consist of approximately 30,000 child-directed utterances which were recorded between ages 1;6 to 5;1. Sentence fragments and *wh*-questions were excluded from our analyses, though the results do not differ substantially when fragments and questions are included.

The training regime was the same as in the gold standard analysis: The models received parse trees for each sentence as input. Because the CHILDES database does not provide parses, we used the corpus of parsed CHILDES sentences developed by Pearl and Sprouse (2013). We include these CHILDES analyses below because it is more likely than newspaper text to be representative of the input received by a typical natural language learner. Note that we did not include the CHILDES corpus in the previous evaluation because we do not have gold-standard annotations of the argument/modifier status of any phrases in this corpus. The differences between the two corpora

can be illustrated by several simple statistics. On average, the sentences in the WSJ corpus contain 25 words, while the sentences in CHILDES contain 6.5 words. The parse trees in the WSJ contain 71 nodes on average, while those in CHILDES contain 19 nodes. Finally, the average maximum depth (i.e., the longest distance from the root node to a leaf) of the parse trees in the WSJ is 10, while the average depth in CHILDES is 5. These statistics show that the sentences in the WSJ are significantly longer and more syntactically complex than those in CHILDES.

Lexical and derivational simplicity in the training set 5.2.1

In this section, we compare the ways in which the argument-modifier model and the argument-only model represent the input training data for the WSJ and CHILDES corpora. We first examine the bias for reusable lexical items. Figure 5 shows the cumulative frequencies of the $1,000$ most often stored tree fragments in the lexicons of the argument-modifier model and argument-only model, as learned on the CHILDES (left) and WSJ (right) training sets. We computed these values by first ranking tree fragments by frequency of occurrence in the lexicon; this resulted in a rank for each type of tree fragment, with lower rank corresponding to greater frequency. Then, for all tree fragments below a given rank (e.g., for the tree fragments below rank 100, corresponding to the 100 most common tree fragments), we computed the sum of the frequencies of these fragments.[9] The figure shows that the most reusable tree fragments learned by the argument-modifier model are used more often across sentences in the training corpus than the most reusable tree fragments learned by the argument-only model. The difference is more pronounced in the WSJ training set. This is likely due to the greater sentence complexity and greater number of modifiers in newspaper text compared to child-directed speech.

[9] Tree fragments which were rooted at part-of-speech nodes (pre-terminals) were excluded from this and subsequent analyses. A subtree which is rooted at a part of speech necessarily consists of exactly two nodes (the part of speech and the terminal string which it is a parent of). As a result, there is only one parse for such subtrees, and both models will always parse such a subtree in an identical manner.

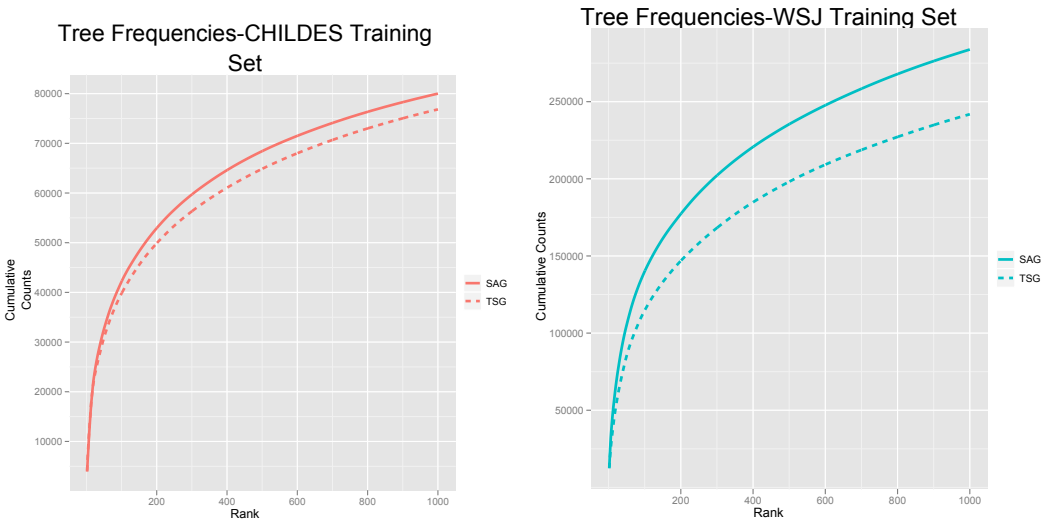

Figure 5: Cumulative Frequencies

We next examine which model was able to provide simpler derivations of individual sentences. One way to measure this is to look at the complexity of stored tree fragments learned by each model. If a model stores tree fragments which are larger (on average), then it must account for each sentence using fewer fragments (on average). Figure 6 shows the cumulative average number of nodes (left) and average depth (right) of the 1,000 most common elementary trees learned by the argument-modifier model and the argument-only model on the CHILDES and WSJ corpora. These figures show that the elementary trees learned by the argument-modifier model are more complex than those learned by the argument-only model, and therefore that the derivation of individual sentences involve fewer lexical items (on average). The difference in tree fragment complexity is greater for the WSJ corpus than for CHILDES, most likely because the parse trees in the WSJ corpus contain a greater number of nodes and have greater depth than those in CHILDES.

Figure 7 shows the cumulative proportion of nodes in the training corpus which are accounted for by the 1,000 most common stored tree fragments learned by the argument-modifier model and the argument-only model. Because it learns both more reusable and larger stored

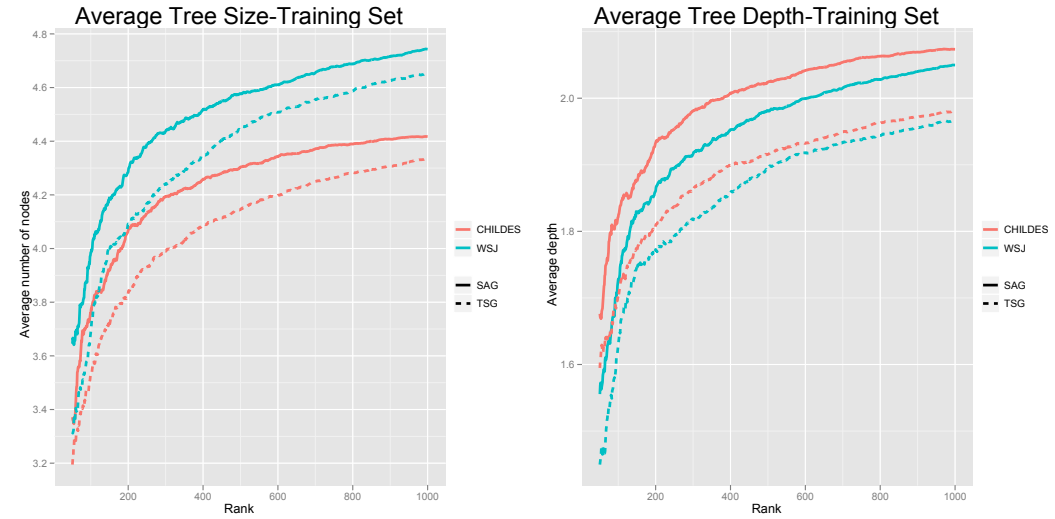

Figure 6: Complexity of stored tree fragments

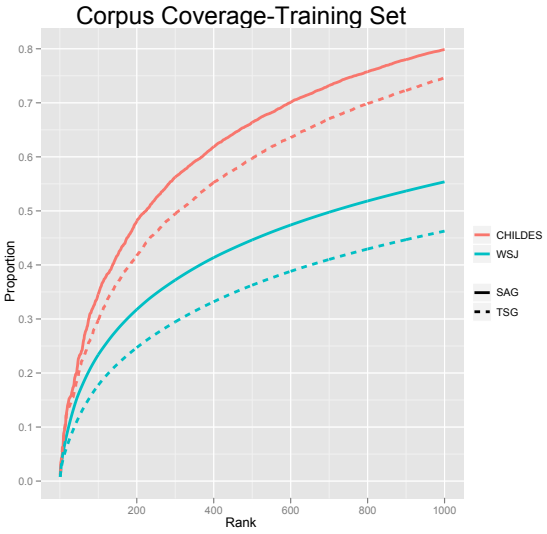

Figure 7: Cumulative coverage

tree fragments, the argument-modifier model is able to account for the training data using a smaller number of stored items.

5.2.2 Lexical and derivational simplicity in new sentences

The previous analyses demonstrate that the argument-modifier model learns a more parsimonious representation of the input than the argument-only model. An important caveat, however, is that the argument-modifier model is a more complex grammatical formalism than the argument-only model. Whereas the argument-only model only has a single composition operation (SUBSTITUTE), the argument-modifier model has two composition operations (SUBSTITUTE and SISTER-ADJOIN). This means that the model has more degrees of freedom in explaining an input training set. As a result, it is possible that the argument-modifier model's performance is due to *overfitting*. A standard method to diagnose overfitting is to evaluate the model on novel data. If the model is overfit on the training data, then it will have captured spurious regularities in its input, and will therefore generalize poorly to new data.

In order to determine whether the parsimony advantages of the argument-modifier model generalize to novel data, we divided the CHILDES and WSJ corpora into training and test portions. The training portion was used as input to the argument-modifier model and argument-only model, while the test portion was used for evaluating the generalizability of these grammars. For the WSJ corpus, we used the standard split: training on sections 2–21 and testing on section 23. For the CHILDES corpus, we randomly selected 80% of the sentences for training, and used the remaining 20% for test.

Our evaluation of the argument-modifier model follows the method in the previous section. We compare the argument-modifier model to the argument-only model, and conduct similar analyses of fragment reusability and derivation complexity (fragment size). In order to perform these analyses, we applied our sampler to the test portions of the two corpora without incorporating any new tree fragments into the set of learned tree fragments. That is, after training we froze the set of lexical fragments (and associated counts) and did not allow any learning from the test set during inference. Thus each sentence in the test corpus was analyzed as if it were the next observed

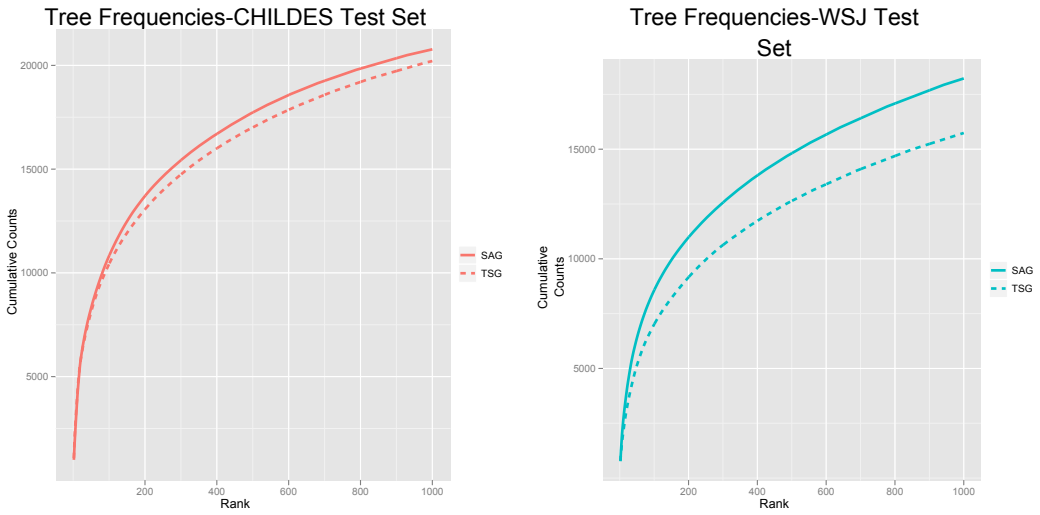

Figure 8: Cumulative frequencies (generalization)

sentence after training. The analyses below are otherwise identical to those in the previous section.

We first examine the bias for reusable lexical items. Figure 8 shows the cumulative frequencies of the 1,000 most common tree fragments from the lexicons of the argument-modifier model and the argument-only model, as inferred on the CHILDES (left) and WSJ (right) test sets. The figure shows that the commonly stored tree fragments learned by the argument-modifier model are used more often across sentences in the test corpus. The difference is again more pronounced in the WSJ test set due the greater sentence complexity and number of modifiers in newspaper text compared to child-directed speech.

We next turn to the bias for simple derivations of individual sentences. As in the previous simplicity analyses, we measure derivation complexity by examining the size of tree fragments used to account for test sentences. Larger tree fragments imply fewer fragments per derivation. Figure 9 shows the cumulative average number of nodes (left) and average depth (right) of the 1,000 most common elementary trees used to account for the new sentences by the argument-modifier model and the argument-only model on the CHILDES and WSJ test corpora. These figures show that the elementary trees learned by the

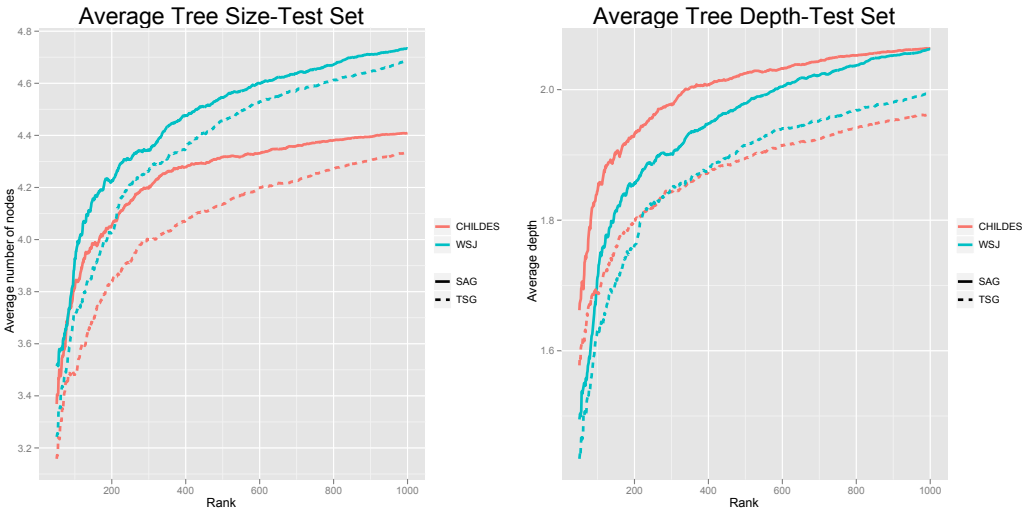

Figure 9: Complexity of stored tree fragments (generalization)

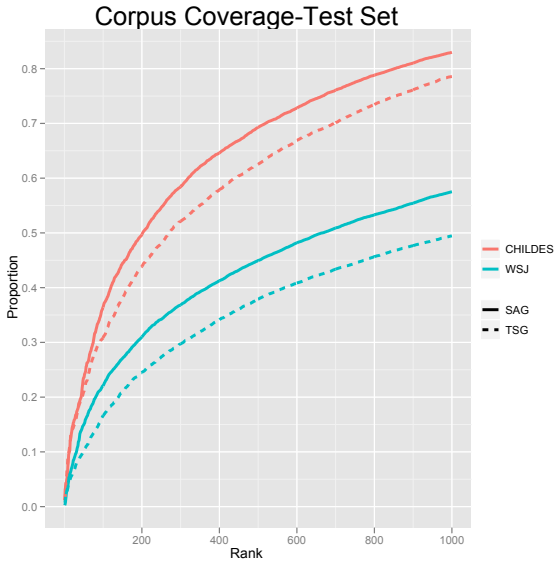

Figure 10: Cumulative coverage (generalization)

argument-modifier model are more complex than those learned by the argument-only model and, therefore, that the derivations of individual sentences are simpler.

Thus, the argument-modifier model's advantage in both kinds of simplicity transfers to the case of new sentences. This is further confirmed in Figure 10 which shows the cumulative proportion of nodes in the training corpus that are accounted for by the 1,000 most common stored tree fragments learned by the two models.

Discussion 5.2.3

The preceding analyses indicate that the sister-adjunction model is able to learn both more reusable lexical items, and simpler derivations of each sentence than the tree-substitution model. As we discussed previously, the inference performed in learning the set of lexical fragments for the argument-only model can be understood in terms of a tradeoff. All else being equal, smaller tree fragments are more reusable, leading to smaller lexica. However, larger tree fragments lead to simpler derivations, since fewer are needed per derivation. For a given corpus, if a particular model is at or near an optimum, increasing the reusability of lexical items in an otherwise optimal model will necessarily increase the complexity of derivations, and decreasing the complexity of derivations will necessarily increase the size of the lexicon. Nevertheless, the argument-modifier model is better in both simplicity measures, indicating that it provides a globally superior account of the input data be learning a smaller lexicon of larger tree fragments.

To understand these results better, again consider the example sentences in Figure 4. The argument-modifier model is able to use a single elementary tree (stretching from the root `S` node to the verb *put*) to derive the core of both sentences. In contrast, as Figure 4 shows, the argument-only model will require two distinct elementary trees, one with three arguments under the `VP` node (for the first sentence) and one with four arguments (for the second). Thus, because the argument-modifier model can compose an optional `PP` such as *at 5 o'clock* separately from a sentence's core argument structure, it can re-use the same elementary tree to derive a greater number sentences in the corpus. This explains how the argument-modifier model can use

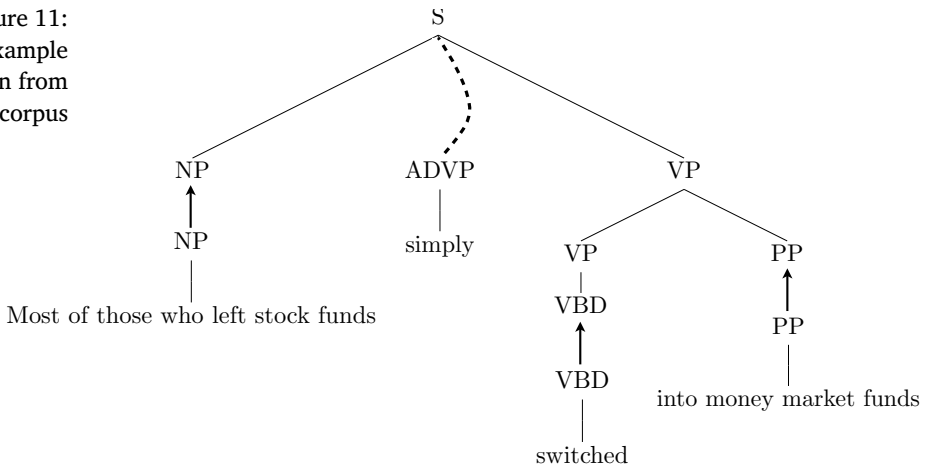

Figure 11: Example derivation from the WSJ corpus

its sister-adjunction operation to find more reusable elementary trees than the argument-only model. It is driven to do so by the the prior preference for a smaller lexicon.

Figure 11 illustrates a representative example from the WSJ corpus. By using SISTER-ADJOIN to account for the ADVP node separately from the rest of the sentence's derivation, the argument-modifier model was able to use a common depth-three elementary tree to derive the backbone of the sentence. By contrast, the argument-only model must include the ADVP node in an elementary tree; this elementary tree is much less common in the corpus.

6 CONCLUSION

In this paper, we have studied the learnability of the argument-modifier status of phrases. We have formulated the distinction as one between lexical and non-lexical modes of composition which give rise to three differences between the two types of constituents which have distributional consequences: **iterability** v. **finiteness**), **optionality** v. **obligatoriness**, and **structural flexibility** v. **structural fixity**. We then proposed that the modifier or argument status of individual phrases could be learned based on optimizing a tradeoff between two competing biases: the *simple lexicon bias* and the *simple derivation bias*.

Our first set of gold standard results indicate that our formalization of the distinction accords with the traditional distinction between arguments and modifiers. Furthermore, our results show that linguistic input contains a strong distributional signal to the modifier/argument status of individual phrases – at least for a learner making use of the tradeoff between lexical and derivational simplicity.

Our second set of results illustrate why the argument-modifier model is able to identify the status of individual phrases. The addition of the SISTER-ADJUNCTION operation allows the model to put derivations into a kind of normal form for which the optimal lexicon contains of both more complex and more reusable fragments. Thus, the argument-modifier model achieves a greater degree of lexical and derivational simplicity simultaneously.

Taken together, these results show that there is considerable distributional evidence for the traditional argument-modifier distinction, but that a simplicity-based learner equipped with lexical and extra-lexical modes of composition could make use of this evidence to acquire the pattern of arguments and modifiers in their language. This result is complementary to traditional linguistic argumentation about the distinction. Our formulation of the problem has deliberately ignored any semantic or, in fact, any non-distributional aspects of the argument-modifier distinction. Any such systematically correlated information should only make the learning problem easier.

APPENDIX

FORMALIZATION OF THE MODELS

The argument-modifier model extends earlier work on induction of Bayesian TSGs (Cohn *et al.* 2010; O'Donnell 2011, 2015; O'Donnell *et al.* 2011; Post and Gildea 2009). The Pitman-Yor Process allows the complexity of the lexicon to grow with more input sentences, while still enforcing a bias for more compact lexicons (Pitman and Yor 1995). As discussed in Section 3.1, the model has two components: (i) A distribution over elementary trees, similar to earlier models of Bayesian TSG induction, and (ii) a distribution over modifiers.

Algorithm 1 provides pseudocode for the generative model. Note that throughout, we will use the notation c_p to refer to the nonterminal category of a node p.

For each node p, the distribution over elementary trees rooted at that node is given by:

$$G_{c_p}|a_{c_p}, b_{c_p}, P_E \sim \text{PYP}(a_{c_p}, b_{c_p}, P_E(\cdot|c_p)) \tag{1}$$

where $P_E(\cdot|c_p)$ is a context free distribution over elementary trees with root label c_p. The hyperparameters a_{c_p}, b_{c_p} are set to $a_{c_p} = 0,\ b_{c_p} = 1$ for this paper.[10]

The context-free distribution over elementary trees $P_E(e|c)$ is defined by:

$$P_E(e|c) = \prod_{i \in I(e)} (1 - s_{c_i}) \prod_{f \in F(e)} s_{c_f} \prod_{c' \to \alpha \in e} P_{\text{cfg}}(\alpha|c'), \tag{2}$$

where $I(e)$ is the set of internal nodes in e, $F(e)$ is the set of frontier nodes, s_c is the probability that we stop expanding at a node labeled c, and $P_{\text{cfg}}(\alpha|c')$ is the probability of the context-free rule expanding category c' to the sequence α, $c' \to \alpha$. For this paper, the parameters s_c are set to 0.5. The distribution $P_{\text{cfg}}(\alpha|c')$ is defined using a distribution that is similar to the Infinite PCFG (Finkel *et al.* 2007; Liang *et al.* 2007),[11] which provides a Dirichlet process prior for PCFG rules.[12]

[10] Given these parameter values, the prior reduces to the model known as a Dirichlet process. Since our implementation allows for other values of a we present the more general version of the mathematics.

[11] Our base distribution over PCFG rules differs from the Infinite PCFG as presented in Liang *et al.* (2007) in a number of ways. First, rather than being a hierarchical Dirichlet process model, our set of nonterminal categories is fixed to be equal to the set of nonterminal categories in the treebank. Second, our rules are not fixed to be in Chomsky normal form, but rather the length of the right-hand side of each rule is sampled from a geometric distribution, and each child symbol is drawn conditioned on the parent symbol and then entire left context of the symbol, which is backed-off using the scheme of Teh (2006) and Goldwater *et al.* (2006).

[12] We use this nonparametric prior so that in addition to learning a distribution over elementary trees, we can also learn a distribution over context-free rules. The inferred distribution over context-free rules may substantially differ from the maximum-likelihood estimate derived from the corpus, as nodes that the model

Algorithm 1: Sister-adjunction grammar

$\beta \sim Dir(1, ..., 1)$ [draw prior over nonterminals]
for each nonterminal sequence $c_1, ..., c_n$:
  $P_{\text{rhs}}(c_1, ..., c_n) = \frac{1}{2^n} \prod_i \beta_{c_i}$ [define base distribution for pcfg prior]
for each nonterminal c:
  $P_{\text{cfg}}(\cdot|c) \sim \text{DP}(a, P_{\text{rhs}}(\cdot))$ [draw distributions over CF rules]
for each nonterminal c:
  for each elementary tree e rooted at c:
    $F(e)$ = frontier of e, $I(e)$ = interior nodes of e
    $P_E(e|c) = \prod_{i \in I(e)} (1 - s_{c_i}) \prod_{f \in F(e)} s_{c_f} \prod_{c' \to \alpha \in e} P_{\text{cfg}}(\alpha|c')$
  $G_c \sim \text{PYP}(a_c, b_c, P_E(\cdot|c))$ [draw distributions over elementary trees]

$\theta \sim Dir(1, ..., 1)$ [draw base distribution over nonterminals]
for each sequence of nonterminals $C = q_l, ..., q_1$: [draw modifier distributions]
  if length$(C) == 1$
    $H_C \sim \text{DP}(\alpha, \text{Multinomial}(\theta))$
  else
    $H_C \sim \text{DP}(\alpha, H_{C'})$, where $C' = q_{l-1}, ..., q_1$

for each node f on the frontier of the parse tree:
  $e \sim G_{c_f}$ [sample an elementary tree rooted at category c_f]
  substitute e at f
  for each internal node p in e:
    for each argument child d_i of p:
      j$=1$
      $C = c_{d_1}, s_{1,1}, ..., c_{d_i}, c_p$ [C is the context for d_i]
      $s_{i,j} \sim H_C$ [draw from the modifier distribution for d_i]
      while $s_{i,j} \neq STOP$ [continue until drawing a STOPsymbol]
        sister-adjoin a node labeled $s_{i,j}$ between d_i, d_{i+1}
        j$+=1$
        $C = c_{d_1}, s_{1,1}, ..., c_{d_i}, s_{i,1}, ..., s_{i,j-1}, c_p$ [add sampled modifier to the context]
        $s_{i,j} \sim H_C$

A similar base distribution for elementary trees is used in Cohn *et al.* (2010) and Post and Gildea (2009). The base distribution over elementary trees thus will be biased towards small elementary trees which use frequent context-free expansions.

In addition to defining a distribution over elementary trees, we also define a distribution which governs modification via sister-adjunction. To sample a modifier, we first decide whether or not to sister-adjoin into location l in a tree. Following this step, we sample a modifier category (e.g., a PP) conditioned on the location l's *context*: its parent and left siblings. Because contexts are sparse, we use a backoff scheme based on hierarchical Dirichlet processes similar to the ngram backoff schemes defined in Teh (2006) and Goldwater *et al.* (2006). Let e be an elementary tree that has been substituted into the parse tree, and let p be an internal node in e. The node p will have $n \geq 1$ children derived by argument substitution: $d_1, ..., d_n$. In order to sister-adjoin between two of these children d_i, d_{i+1}, we recursively sample nonterminals $s_{i,1}, ..., s_{i,k}$ until we sample a $STOP$ symbol:

$$P_a(s_{i,1}, ..., s_{i,k}, STOP|C_0) = (\prod_{j=1}^{k} P_a(s_{i,j}|C_j)) \cdot P_a(STOP|C_{k+1}) \tag{3}$$

where $C_j = c_{d_1}, s_{1,1}, ..., c_{d_i}, s_{i,1}, ..., s_{i,j-1}, c_p$ is the context for the jth modifier between these children. The distribution over sister-adjoined nonterminals is defined using a hierarchical Dirichlet process to implement backoff in a prefix tree over contexts. Given the context $C = q_l, ..., q_1$ (where $l > 1$), we define the distribution H_C over sister-adjoined nonterminals $s_{i,j}$ by:

$$H_C \sim \mathrm{DP}(\alpha, H_{C'}), \tag{4}$$

where $C' = q_{l-1}, ..., q_1$. A sample is drawn from the root of the hierarchy when the context C is of length 1 (and hence the backed-off context is empty). A Dirichlet-multinomial distribution is used as the

---

labels as modifiers are not included in the derivation of an elementary tree. This approach is also suitable to the unsupervised setting (as in Cohn *et al.* 2010), in which the derived trees in the corpus are not observed.

prior in this case:

$$\theta \sim Dir(1, ..., 1)$$
$$H_C \sim \text{DP}(\alpha, \text{Multinomial}(\theta))$$

where $C = q_1$ and θ is a vector with entries for each nonterminal and an entry for the $STOP$ symbol. The backoff scheme for sampling modifiers is illustrated in Figure 12.

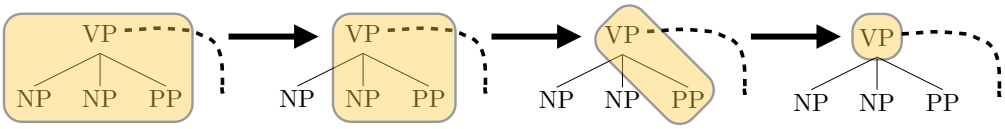

Figure 12: This illustrates the procedure for sampling a modifier at the right edge of a VP. The distribution over modifiers is conditioned on the modifier's context, which contains its VP parent and left siblings, as illustrated on the left of the figure. This distribution is estimated by successively backing off to smaller contexts

REFERENCES

Leon Bergen, Edward Gibson, and Timothy J. O'Donnell (2015), A learnability analysis of argument and modifier structure, *lingbuzz*, (lingbuzz/002502).

Robert C. Berwick (1982), *Locality principles and the acquisition of syntactic knowledge*, Ph.D. thesis, Massachusetts Institute of Technology.

Robert C. Berwick (1985), *The acquisition of syntactic knowledge*, The MIT Press, Cambridge, Massachusetts and London, England.

Rens Bod (1998), *Beyond grammar: An experience-based theory of language*, CSLI Publications, Palo Alto, CA.

Rens Bod, Remko Scha, and Khalil Sima'an, editors (2003), *Data-oriented parsing*, CSLI, Palo Alto, CA.

Robert D. Borsley (1999), *Syntactic theory: A unified approach*, Edward Arnold, London, England.

Michael R. Brent (1997), Toward a unified model of lexical acquisition and lexical access, *Journal of Psycholinguistic Research*, 26(3):363–375.

Michael R. BRENT (1999), An efficient, probabilistically sound algorithm for segmentation and word discovery, *Machine Learning*, 34:71–105.

Joan BRESNAN (2001), *Lexical functional syntax*, Wiley-Blackwell, Oxford, England.

Roger W. BROWN (1973), *A first language: The early stages*, Harvard University Press, Cambridge, MA.

Timothy A. CARTWRIGHT and Michael R. BRENT (1994), Segmenting speech without a lexicon: Evidence for a bootstrapping model of lexical acquisition, in *Proceedings of the 16th Annual Meeting of the Cognitive Science Society*.

David CHIANG (2000), Statistical parsing with an automatically-extracted tree adjoining grammar, in *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics.

David CHIANG and Daniel BIKEL (2002), Recovering latent information in treebanks, in *Proceedings of COLING 2002*.

Noam CHOMSKY (1951 [1979]), *Morphophonemics of modern Hebrew*, Garland Publishing, New York, NY.

Noam CHOMSKY (1955 [1975]), *The logical structure of linguistic theory*, Plenum Press, New York, NY.

Noam CHOMSKY (1964), *Current issues in linguistic theory*, Janua Linguarum: Studia Memoriae Nicolai van Wijk Dedicata, Mouton, The Hague, The Netherlands.

Noam CHOMSKY (1970), Remarks on nominalization, in Roderick JACOBS and Peter ROSENBAUM, editors, *Readings in English Transformational Grammar*, Ginn and Company, Waltham, MA.

Noam CHOMSKY (1993), A minimalist program for linguistic theory, in Kenneth L. HALE and Samuel Jay KEYSER, editors, *The View from Building 20: Essays in Honor of Sylvain Bromberger*, pp. 1–52, The MIT Press, Cambridge, Massachusetts and London, England.

Noam CHOMSKY (1995a), Bare phrase structure, in Gerth WEBELHUTH, editor, *Government and Binding Theory and the Minimalist Program*, pp. 383–349, Blackwell.

Noam CHOMSKY (1995b), *The minimalist program*, The MIT Press, Cambridge, MA.

Trevor COHN, Phil BLUNSOM, and Sharon GOLDWATER (2010), Inducing tree-substitution grammars, *Journal of Machine Learning Research*, 11:3053–3096.

Bernard COMRIE (1993), Argument structure, in Joachim JACOBS, Arnim VON STECHOW, Wolfgang STERNEFELD, and Theo VENNEMAN, editors, *Syntax: An International Handbook*, pp. 905–914, Walter de Gruyter, Berlin, Germany.

Denis Creissels (2014), Cross-linguistic variation in the treatment of beneficiaries and the argument *vs*. adjunct distinction, *Linguistic Discovery*, 12(2).

William Croft (2001), *Radical construction grammar: Syntactic theory in typological perspective*, Oxford University Press, Oxford, England.

Peter Culicover and Ray Jackendoff (2005), *Simpler syntax*, Oxford University Press, Oxford, England.

Carl De Marcken (1996a), The unsupervised acquisition of a lexicon from continuous speech, Technical Report AI-memo-1558, CBCL-memo-129, Massachusetts Institute of Technology – Artificial Intelligence Laboratory.

Carl De Marcken (1996b), *Unsupervised language acquisition*, Ph.D. thesis, Massachusetts Institute of Technology.

Jacob Feldman (2000), Minimization of Boolean complexity in human concept learning, *Nature*, 407(6804):630–633.

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning (2007), The infinite tree, in *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

Diana Forker (2014), A canonical approach to the argument/adjunct distinction, *Linguistic Discovery*, 12(2).

L. T. F. Gamut (1991), *Logic, language, and meaning volume II: Intensional logic and logical grammar*, University of Chicago Press, Chicago, IL.

Gerald Gazdar, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag (1985), *Generalized phrase structure grammar*, Harvard University Press, Cambridge, MA.

John Anton Goldsmith (2011), The evaluation metric in generative grammar, in *Proceedings of the 50th Anniversay Celebration of the MIT Department of Linguistics*.

Sharon Goldwater (2006), *Nonparametric bayesian models of lexical acquisition*, Ph.D. thesis, Brown University.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson (2006), Interpolating between types and tokens by estimating power-law generators, in *Advances in Neural Information Processing Systems 18*.

Noah D. Goodman, Joshua B. Tenenbaum, Jacob Feldman, and Thomas L. Griffiths (2008), A rational analysis of rule-based concept learning, *Cognitive Science*, 32(1):108–154.

Peter D. Grünwald (2007), *The minimum description length principle*, The MIT Press, Cambridge, MA.

Liliane Haegeman (1994), *Introduction to government and binding theory*, Blackwell, Oxford, England.

Martin HASPELMATH (2014), Arguments and adjuncts as language-particular syntactic categories and as comparative concepts, *Linguistic Discovery*, 12(2).

Irene HEIM and Angelika KRATZER (1998), *Semantics in generative grammar*, Blackwell Publishing, Malden, MA.

Norbert HORNSTEIN and David W. LIGHTFOOT (1981), *Introduction to explanation in linguistics: The logical problem of language acquisition*, Addison Wesley Longman, Upper Saddle River, NJ.

Anne S. HSU and Nick CHATER (2010), The logical problem of language acquisition goes probabilistic: No negative evidence as a window on language acquisition, *Cognitive Science*, 34:972–1016.

Anne S. HSU, Nick CHATER, and Paul M. B. VITÁNYI (2011), The probabilistic analysis of language acquisition: Theoretical, computational, and experimental analysis, *Cognition*, 120:380–390.

Anne S. HSU, Nick CHATER, and Paul M. B. VITÁNYI (2013), Language learning for positive evidence reconsidered: A simplicity-based approach, *Topics in Cognitive Science*, 5:35–55.

Rodney HUDDLESTON and Geoffrey K. PULLUM (2002), *The cambridge grammar of English language*, Cambridge University Press, Cambridge, England.

Ray JACKENDOFF (2002), *Foundations of language*, Oxford University Press, New York, NY.

David E. JOHNSON and Paul M. POSTAL (1980), *Arc pair grammar*, Princeton University Press, Princeton, NJ.

Mark JOHNSON, Thomas L. GRIFFITHS, and Sharon GOLDWATER (2007), Adaptor Grammars: A framework for specifying compositional nonparametric Bayesian models, in *Advances in Neural Information Processing Systems 19*, MIT Press, Cambridge, MA.

Aravind K. JOSHI and Leon S. LEVY (1975), Tree adjunct grammars, *Journal of Computer and System Sciences*, 10:136–163.

Jean-Pierre KOENIG, Gail MAUNER, and Breton BIENVENUE (2003), Arguments for adjuncts, *Cognition*, 89:67–103.

Paul R. KROEGER (2004), *Analyzing syntax: A lexical-functional approach*, Cambridge University Press, Cambridge, England.

Ming LI and Paul M. B. VITÁNYI (2008), *An introduction to Kolmogorov complexity and its applications*, Springer, Berlin, Germany, third edition.

Percy LIANG, Slav PETROV, Michael I. JORDAN, and Dan KLEIN (2007), The infinite PCFG using hierarchical Dirichlet processes, in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 688–697.

Brian MacWhinney (2000), *The childes project: Tools for analyzing talk.*, Lawrence Erlbaum Associates, Mahwah, NJ.

Alec Marantz (2013), Verbal argument structure: Events and participants, *Lingua*, 130:152–168.

Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor (1999), Treebank-3, Technical report, Linguistic Data Consortium, Philadelphia.

Peter H. Matthews (1981), *Syntax*, Cambridge University Press, Cambridge, England.

Sally McConnell-Ginet and Gennaro Chierchia (2000), *Meaning and grammar: An introduction to semantics*, MIT Press, Cambridge, MA.

Igor Mel'čuk (1988), *Dependency syntax : Theory and practice*, The SUNY Press, Albany, NY.

Michael Moortgat (1997), Categorial type logics, in *Handbook of Logic and Language*, pp. 93–177, Elsevier.

Timothy J. O'Donnell (2011), *Productivity and reuse in language*, Ph.D. thesis, Harvard University, Cambridge, MA.

Timothy J. O'Donnell (2015), *Productivity and reuse in language: A theory of linguistic computation and storage*, The MIT Press, Cambridge, MA.

Timothy J. O'Donnell, Jesse Snedeker, Joshua B. Tenenbaum, and Noah D. Goodman (2011), Productivity and reuse in language, in *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, Boston, MA.

Martha Palmer, P. Kingsbury, and Daniel Gildea (2005), The proposition bank: An annotated corpus of semantic roles, *Computational Linguistics*, 31(1):71–106.

Lisa Pearl and Sharon Goldwater (2016), Statistical learning, inductive bias, and Bayesian inference in language acquisition, in Jeffrey Lidz, William Snyder, and Joe Pater, editors, *The Oxford Handbook of Developmental Linguistics*, Oxford University Press, Oxford, England.

Lisa Pearl and Jon Sprouse (2013), Syntactic islands and learning biases: Combining experimental syntax and computational modeling to investigate the language acquisition problem, *Language Acquisition*, 20:23–68.

Amy Perfors, Joshua B. Tenenbaum, and Terry Regier (2011), The learnability of abstract syntactic principles, *Cognition*, 118(3):306–338.

Lawrence Phillips and Lisa Pearl (2014), The utility of cognitive plausibility in language acquisition modeling: Evidence from word segmentation, manuscript.

Steven Thomas Piantadosi (2011), *Learning and the language of thought*, Ph.D. thesis, Massachusetts Institute of Technology.

Steven Thomas PIANTADOSI (2021), The computational origin of representation, *Minds and Machines*, 31:1–58.

Jim PITMAN and Marc YOR (1995), The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator, Technical report, Department of Statistics University of California, Berkeley.

Carl POLLARD and Ivan A. SAG (1994), *Head-driven phrase structure grammar*, University of Chicago Press, Chicago, IL.

Matt POST and Daniel GILDEA (2009), Bayesian learning of a tree substitution grammar, in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*.

Matt POST and Daniel GILDEA (2013), Bayesian tree substitution grammars as a usage-based approach, *Language and Speech*, 56(3):291–308.

Adam PRZEPIÓRKOWSKI (1999a), *Case assignment and the complement/adjunct dichotomy*, Ph.D. thesis, Neuphilologischen Fakultät der Universität Tübingen, Tübingen.

Adam PRZEPIÓRKOWSKI (1999b), On case assignment and "adjuncts as complements", in Gerth WEBELHUTH, Jean-Pierre KOENIG, and A. KATHOL, editors, *Lexical and Constructional Aspects of Linguistic Explanation*, pp. 223–245, CSLI Publications.

Adam PRZEPIÓRKOWSKI (2017), Hierarchical lexicon and the argument/adjunct distinction, in *Proceedings of the Lexical Functional Grammar 2017 (LFG'17) Conference*, University of Konstanz.

Andrew RADFORD (1988), *Transformational grammar: A first course*, Cambridge University Press, Cambridge, England.

György RÁKOSI (2006), *Dative experiencer predicates in Hungarian*, Ph.D. thesis, Universiteit Utrecht.

Owen RAMBOW, K. VIJAY-SHANKER, and David WEIR (1995), D-tree grammars, in *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, Association for Computational Linguistics.

Ezer RASIN and Roni KATZIR (2016), On evaluation metrics in optimality theory, *Linguistic Inquiry*, 47(2):235–282.

Jorma RISSANEN (1978), Modeling by shortest data description, *Automaticata*, 14(5):465–471.

Ivan A. SAG (2012), Sign-based Construction Grammar: An informal synopsis, in Hans BOAS and Ivan A. SAG, editors, *Sign-Based Construction Grammar*, pp. 101–107, CSLI Publications, Palo Alto, CA.

Remko SCHA (1990), Taaltheorie en taaltechnologie; competence en performance, in R. DE KORT and G.L.J. LEERDAM, editors, *Computertoepassingen in de Neerlandistiek*, pp. 7–22, Landelijke Vereniging van Neerland.

Remko Scha (1992), Virtuele grammatica's en creatieve algoritmes, *Gramma/TTT*, 1(1):57–77.

Yves Schabes and Stuart M. Shieber (1994), An alternative conception of tree-adjoining derivation, *Computational Linguistics*, 20(1):91–124.

Yves Schabes and Richard C. Waters (1995), Tree insertion grammar: A cubic-time parsable formalism that lexicalizes context-free grammar without changing the trees produced, *Computational Linguistics*, 21(4):479–513.

Carson T. Schütze (1995), PP attachment and argumenthood, Technical report, Papers on language processing and acquisition, MIT working papers in linguistics, Cambridge, Ma.

Carson T. Schütze and Edward Gibson (1999), Argumenthood and English prepositional phrase attachment, *Journal of Memory and Language*, 40:409–431.

Ray Solomonoff (1978), Complexity-based induction systems: comparisons and convergence theorems, *IEEE Transactions on Information Theory*, 24(4):422–432.

Ray J. Solomonoff (1964a), A formal theory of inductive inference. Part I, *Information and Control*, 7(1):1–22.

Ray J. Solomonoff (1964b), A formal theory of inductive inference. Part II, *Information and Control*, 7(2):224–254.

Edward P. Stabler (1997), Derivational minimalism, in *Logical Aspects of Computational Linguistics*, Springer, Berlin, Germany.

Mark Steedman (2000), *The syntactic process*, MIT Press, Cambridge, MA.

Andreas Stolcke and Stephen Omohundro (1994), Inducing probabilistic grammars by Bayesian model merging, in *Proceedings of the International Conference on Grammatical Inference*.

Maggie Tallerman (2015), *Understanding syntax*, Routledge, London, England, fourth edition.

Yee Whye Teh (2006), A Bayesian interpretation of interpolated Kneser-Ney, Technical Report TRA2/06, National University of Singapore, School of Computing.

Damon Tutunjian and Julie E. Boland (2008), Do we need a distinction between arguments and adjuncts? Evidence from psycholinguistic studies of comprehension, *Language and Linguistics Compass*, 2(4):631–646.

Heinz Vater (1978), On the possibility of distinguishing between complements and adjuncts, in *Valence, semantic case and grammatical relations*, pp. 21–45, John Benjamins.

Søren Wichmann (2014), Arguments and adjuncts cross-linguistically: A brief introduction, *Linguistic Discovery*, 12(2).

J. Gerard WOLFF (1977), The discovery of segments in natural language, *British Journal of Psychology*, 68:97–106.

J. Gerard WOLFF (1980), Language acquisition and the discovery of phrase structure, *Language and Speech*, 23(3):255–269.

J. Gerard WOLFF (1982), Language acquisition, data compression, and generalisation, *Language and Communication*, 2(1):57–89.

Yuan YANG and Steven Thomas PIANTADOSI (2022), One model for the learning of language, *Proceedings of the National Academy of Sciences*, 119(5).

Arnold M. ZWICKY (1993), Heads, bases, and functors, in Greville G. CORBETT, Norman M. FRASER, and Scott MCGLASHAN, editors, *Heads in Grammatical Theory*, pp. 292–315, Cambridge University Press, Cambridge, England.

Leon Bergen
lbergen@ucsd.edu

Department of Linguistics, University of California San Diego, San Diego, California

Edward Gibson
0000-0002-5912-883X
gibson@mit.edu

Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts

Timothy J. O'Donnell
0000-0002-5711-977X
timothy.odonnell@mcgill.ca

McGill University,
Canada CIFAR AI Chair, Mila

Leon Bergen, Edward Gibson, and Timothy J. O'Donnell (2022), *Simplicity and learning to distinguish arguments from modifiers*, Journal of Language Modelling, 10(2):241–286
https://dx.doi.org/10.15398/jlm.v10i2.263