A Probabilistic Model of Syntactic and Semantic Acquisition from Child-Directed Utterances and their Meanings

Tom Kwiatkowski[*] [†] tomk@cs.washington.edu

Sharon Goldwater[*] sgwater@inf.ed.ac.uk

Luke Zettlemoyer[†] lsz@cs.washington.edu

Mark Steedman[*] steedman@inf.ed.ac.uk

[*] ILCC, School of Informatics
University of Edinburgh
Edinburgh, EH8 9AB, UK

[†]Computer Science & Engineering
University of Washington
Seattle, WA, 98195, USA

Abstract

This paper presents an incremental probabilistic learner that models the acquistion of syntax and semantics from a corpus of child-directed utterances paired with possible representations of their meanings. These meaning representations approximate the contextual input available to the child; they do not specify the meanings of individual words or syntactic derivations. The learner then has to infer the meanings and syntactic properties of the words in the input along with a parsing model. We use the CCG grammatical framework and train a non-parametric Bayesian model of parse structure with online variational Bayesian expectation maximization. When tested on utterances from the CHILDES corpus, our learner outperforms a state-of-the-art semantic parser. In addition, it models such aspects of child acquisition as "fast mapping," while also countering previous criticisms of statistical syntactic learners.

1 Introduction

Children learn language by mapping the utterances they hear onto what they believe those utterances mean. The precise nature of the child's prelinguistic representation of meaning is not known. We assume for present purposes that it can be approximated by compositional logical representations such as (1), where the meaning is a logical expression that describes a relationship $have$ between the person you refers to and the object $another(x, cookie(x))$:

$$\begin{aligned} \textsf{Utterance}: &\ \text{you have another cookie} \qquad (1) \\ \textsf{Meaning}: &\ have(you, another(x, cookie(x))) \end{aligned}$$

Most situations will support a number of plausible meanings, so the child has to learn in the face of *propositional uncertainty*[1], from a set of contextually afforded meaning candidates, as here:

$$\begin{aligned} \textsf{Utterance}: &\ \text{you have another cookie} \\ \begin{matrix}\textsf{Candidate} \\ \textsf{Meanings}\end{matrix} &\begin{cases} have(you, another(x, cookie(x))) \\ eat(you, your(x, cake(x))) \\ want(i, another(x, cookie(x))) \end{cases} \end{aligned}$$

The task is then to learn, from a sequence of such (utterance, meaning-candidates) pairs, the correct lexicon and parsing model. Here we present a probabilistic account of this task with an emphasis on cognitive plausibility.

Our criteria for plausibility are that the learner must not require any language-specific information prior to learning and that the learning algorithm must be strictly *incremental*: it sees each training instance sequentially and exactly once. We define a Bayesian model of parse structure with Dirichlet process priors and train this on a set of (utterance, meaning-candidates) pairs derived from the CHILDES corpus (MacWhinney, 2000) using online variational Bayesian EM.

We evaluate the learnt grammar in three ways. First, we test the accuracy of the trained model in parsing unseen utterances onto gold standard annotations of their meaning. We show that it outperforms a state-of-the-art semantic parser (Kwiatkowski et al., 2010) when run with similar training conditions (i.e., neither system is given the corpus based initialization originally used by Kwiatkowski et al.). We then examine the learning curves of some individual words, showing that the model can learn word meanings on the basis of a single exposure, similar to the *fast mapping* phenomenon observed in children (Carey and Bartlett, 1978). Finally, we show that our

[1] Similar to *referential uncertainty* but relating to propositions rather than referents.

learner captures the step-like learning curves for word order regularities that Thornton and Tesan (2007) claim children show. This result counters Thornton and Tesan's criticism of statistical grammar learners—that they tend to exhibit gradual learning curves rather than the abrupt changes in linguistic competence observed in children.

1.1 Related Work

Models of syntactic acquisition, whether they have addressed the task of learning both syntax and semantics (Siskind, 1992; Villavicencio, 2002; Buttery, 2006) or syntax alone (Gibson and Wexler, 1994; Sakas and Fodor, 2001; Yang, 2002) have aimed to learn a single, correct, deterministic grammar. With the exception of Buttery (2006) they also adopt the Principles and Parameters grammatical framework, which assumes detailed knowledge of linguistic regularities[2]. Our approach contrasts with all previous models in assuming a very general kind of linguistic knowledge and a probabilistic grammar. Specifically, we use the probabilistic Combinatory Categorial Grammar (CCG) framework, and assume only that the learner has access to a small set of general combinatory schemata and a functional mapping from semantic type to syntactic category. Furthermore, this paper is the first to evaluate a model of child syntactic-semantic acquisition by parsing unseen data.

Models of child word learning have focused on semantics only, learning word meanings from utterances paired with either sets of concept symbols (Yu and Ballard, 2007; Frank et al., 2008; Fazly et al., 2010) or a compositional meaning representation of the type used here (Siskind, 1996). The models of Alishahi and Stevenson (2008) and Maurits et al. (2009) learn, as well as word-meanings, orderings for verb-argument structures but not the full parsing model that we learn here.

Semantic parser induction as addressed by Zettlemoyer and Collins (2005, 2007, 2009), Kate and Mooney (2007), Wong and Mooney (2006, 2007), Lu et al. (2008), Chen et al. (2010), Kwiatkowski et al. (2010, 2011) and Börschinger et al. (2011) has the same task definition as the one addressed by this paper. However, the learning approaches presented in those previous papers are not designed to be cognitively plausible, using batch training algorithms, multiple passes over the data, and language specific initialisations (lists of noun phrases and additional corpus statistics), all of which we dispense with here. In particular, our approach is closely related that of Kwiatkowski et al. (2010) but, whereas that work required careful initialisation and multiple passes over the training data to learn a discriminative parsing model, here we learn a generative parsing model without either.

1.2 Overview of the approach

Our approach takes, as input, a corpus of (utterance, meaning-candidates) pairs $\{(s_i, \{m\}_i) : i = 1, \ldots, N\}$, and learns a CCG lexicon Λ and the probability of each *production* $a \rightarrow b$ that could be used in a parse. Together, these define a probabilistic parser that can be used to find the most probable meaning for any new sentence.

We learn both the lexicon and production probabilities from allowable parses of the training pairs. The set of allowable parses $\{t\}$ for a single (utterance, meaning-candidates) pair consists of those parses that map the utterance onto one of the meanings. This set is generated with the functional mapping $\mathcal{T}$:

$$\{t\} = \mathcal{T}(s, m), \tag{2}$$

which is defined, following Kwiatkowski et al. (2010), using only the CCG combinators and a mapping from semantic type to syntactic category (presented in in Section 4).

The CCG lexicon Λ is learnt by reading off the lexical items used in all parses of all training pairs. Production probabilities are learnt in conjunction with Λ through the use of an incremental parameter estimation algorithm, online Variational Bayesian EM, as described in Section 5.

Before presenting the probabilistic model, the mapping $\mathcal{T}$, and the parameter training algorithm, we first provide some background on the meaning representations we use and on CCG.

2 Background

2.1 Meaning Representations

We represent the meanings of utterances in first-order predicate logic using the lambda-calculus. An example logical expression (henceforth also referred to as a lambda expression) is:

$$like(eve, mummy) \tag{3}$$

[2] This linguistic use of the term "parameter" is distinct from the statistical use found elsewhere in this paper.

which expresses a logical relationship $like$ between the entity eve and the entity $mummy$. In Section 6.1 we will see how logical expressions like this are created for a set of child-directed utterances (to use in training our model).

The lambda-calculus uses λ operators to define functions. These may be used to represent functional meanings of utterances but they may also be used as a ‘glue language’, to compose elements of first order logical expressions. For example, the function $\lambda x\lambda y.like(y, x)$ can be combined with the object $mummy$ to give the phrasal meaning $\lambda y.like(y, mummy)$ through the lambda-calculus operation of *function application.*

2.2 CCG

Combinatory Categorial Grammar (CCG; Steedman 2000) is a strongly lexicalised linguistic formalism that tightly couples syntax and semantics. Each CCG lexical item in the lexicon Λ is a triple, written as word $\vdash$ $\mathsf{syntactic\ category}$: $logical\ expression$. Examples are:

$$\begin{aligned} \text{You} &\vdash \mathsf{NP} : you \\ \text{read} &\vdash \mathsf{S\backslash NP/NP} : \lambda x\lambda y.read(y, x) \\ \text{the} &\vdash \mathsf{NP/N} : \lambda f.the(x, f(x)) \\ \text{book} &\vdash \mathsf{N} : \lambda x.book(x) \end{aligned}$$

A full CCG category $\mathsf{X} : h$ has syntactic category X and logical expression h. Syntactic categories may be atomic (e.g., S or NP) or complex (e.g., $\mathsf{(S\backslash NP)/NP}$). Slash operators in complex categories define functions from the range on the right of the slash to the result on the left in much the same way as lambda operators do in the lambda-calculus. The direction of the slash defines the linear order of function and argument.

CCG uses a small set of *combinatory rules* to concurrently build syntactic parses and semantic representations. Two example combinatory rules are forward (>) and backward (<) *application*:

$$\begin{array}{llll} X/Y : f \quad Y : g & \Rightarrow & X : f(g) & (>) \\ Y : g \quad X\backslash Y : f & \Rightarrow & X : f(g) & (<) \end{array}$$

Given the lexicon above, the phrase “You read the book” can be parsed using these rules, as illustrated in Figure 1 (with additional notation discussed in the following section)..

CCG also includes combinatory rules of forward ($> \mathbf{B}$) and backward ($< \mathbf{B}$) *composition*:

$$\begin{array}{ll} X/Y : f \quad Y/Z : g \Rightarrow X/Z : \lambda x.f(g(x)) & (> \mathbf{B}) \\ Y\backslash Z : g \quad X\backslash Y : f \Rightarrow X\backslash Z : \lambda x.f(g(x)) & (< \mathbf{B}) \end{array}$$

3 Modelling Derivations

The objective of our learning algorithm is to learn the correct parameterisation of a probabilistic model $P(s, m, t)$ over (utterance, meaning, derivation) triples. This model assigns a probability to each of the *grammar productions* $a \to b$ used to build the derivation tree t. The probability of any given CCG derivation t with sentence s and semantics m is calculated as the product of all of its production probabilities.

$$P(s, m, t) = \prod_{a \to b \in t} P(b|a) \qquad (4)$$

For example, the derivation in Figure 1 contains 13 productions, and its probability is the product of the 13 production probabilities. Grammar productions may be either *syntactic*—used to build a syntactic derivation tree, or *lexical*—used to generate logical expressions and words at the leaves of this tree.

A syntactic production $\mathsf{C_h} \to \mathcal{R}$ expands a head node $\mathsf{C_h}$ into a result $\mathcal{R}$ that is either an ordered pair of syntactic parse nodes $\langle \mathsf{C_l}, \mathsf{C_r} \rangle$ (for a binary production) or a single parse node (for a unary production). Only two unary syntactic productions are allowed in the grammar: $\mathsf{START} \to \mathsf{A}$ to generate A as the top syntactic node of a parse tree and $\mathsf{A} \to [\mathsf{A}]_{\mathsf{lex}}$ to indicate that A is a leaf node in the syntactic derivation and should be used to generate a logical expression and word. Syntactic derivations are built by recursively applying syntactic productions to non-leaf nodes in the derivation tree. Each syntactic production $\mathsf{C_h} \to \mathcal{R}$ has conditional probability $P(\mathcal{R}|\mathsf{C_h})$. There are 3 binary and 5 unary syntactic productions in Figure 1.

Lexical productions have two forms. *Logical expressions* are produced from leaf nodes in the syntactic derivation tree $\mathsf{A_{lex}} \to m$ with conditional probability $P(m|\mathsf{A_{lex}})$. *Words* are then produced from these logical expressions with conditional probability $P(w|m)$. An example logical production from Figure 1 is $[\mathsf{NP}]_{\mathsf{lex}} \to you$. An example word production is $you \to$ You.

Every production $a \to b$ used in a parse tree t is chosen from the set of productions that could be used to expand a head node a. If there are a finite K productions that could expand a then a K-dimensional *Multinomial distribution* parameterised by θ_a can be used to model the categorical

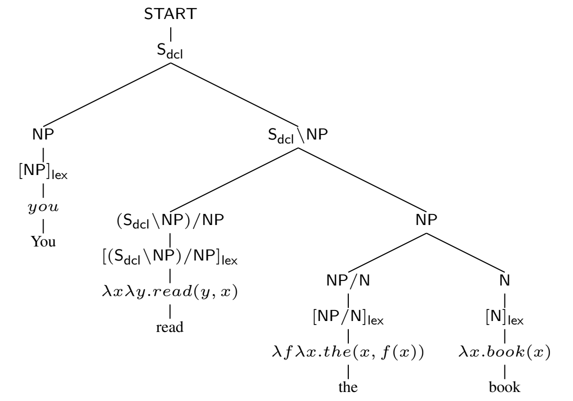

Figure 1: Derivation of sentence `You read the book` with meaning $read(you, the(x, book(x)))$.

choice of production:

$$b \sim \text{Multinomial}(\theta_a) \tag{5}$$

However, before training a model of language acquisition the dimensionality and contents of both the syntactic grammar and lexicon are unknown. In order to maintain a probability model with cover over the countably infinite number of possible productions, we define a Dirichlet Process (DP) prior for each possible production head a. For the production head a, $DP(\alpha_a, H_a)$ assigns some probability mass to all possible production targets $\{b\}$ covered by the base distribution H_a.

It is possible to use the DP as an infinite prior from which the parameter set of a finite dimensional Multinomial may be drawn provided that we can choose a suitable partition of $\{b\}$. When calculating the probability of an (s, m, t) triple, the choice of this partition is easy. For any given production head a there is a finite set of usable production targets $\{b_1, \ldots, b_{k-1}\}$ in t. We create a partition that includes one entry for each of these along with a final entry $\{b_k, \ldots\}$ that includes all other ways in which a could be expanded in different contexts. Then, by applying the distribution G_a drawn from the DP to this partition, we get a parameter vector θ_a that is equivalent to a draw from a k dimensional Dirichlet distribution:

$$\begin{aligned} G_a &\sim DP(\alpha_a, H_a) && (6) \\ \theta_a &= (G_a(b_1), \ldots, G_a(b_{k-1}), G_a(\{b_k, \ldots\}) \\ &\sim \text{Dir}(\alpha_a H(b_1), \ldots, \alpha_a H_a(b_{k-1}), && (7) \\ &\qquad \alpha_a H_a(\{b_k, \ldots\})) \end{aligned}$$

Together, Equations 4-7 describe the joint distribution $P(\mathbf{X}, \mathbf{S}, \theta)$ over the observed training data $\mathbf{X} = \{(s_i, \{m\}_i) : i = 1, \ldots, N\}$, the latent variables $\mathbf{S}$ (containing the productions used in each parse t) and the parsing parameters θ.

4 Generating Parses

The previous section defined a parameterisation over parses assuming that the CCG lexicon Λ was known. In practice Λ is empty prior to training and must be populated with the lexical items from parses t consistent with training pairs $(s, \{m\})$.

The set of allowed parses $\{t\}$ is defined by the function $\mathcal{T}$ from Equation 2. Here we review the *splitting procedure* of Kwiatkowski et al. (2010) that is used to generate CCG lexical items and describe how it is used by $\mathcal{T}$ to create a packed chart representation of all parses $\{t\}$ that are consistent with s and at least one of the meaning representations in $\{m\}$. In this section we assume that s is paired at each point with only a single meaning m. Later we will show how $\mathcal{T}$ is used multiple times to create the set of parses consistent with s and a set of candidate meanings $\{m\}$.

The splitting procedure takes as input a CCG category $\mathsf{X}{:}h$, such as $\mathsf{NP} : a(x, cookie(x))$, and returns a set of *category splits*. Each category split is a pair of CCG categories $(\mathsf{C_l} : m_l, \mathsf{C_r} : m_r)$ that can be recombined to give $\mathsf{X} : h$ using one of the CCG combinators in Section 2.2. The CCG category splitting procedure has two parts: *logical splitting* of the category semantics h; and *syntactic splitting* of the syntactic category X. Each logical split of h is a pair of lambda expressions (f, g) in the following set:

$$\{(f, g) \mid h = f(g) \;\vee\; h = \lambda x.f(g(x))\}, \tag{8}$$

which means that f and g can be recombined using either *function application* or *function composition* to give the original lambda expression h. An example split of the lambda expression $h = a(x, cookie(x))$ is the pair

$$(\lambda y.a(x, y(x)), \lambda x.cookie(x)), \tag{9}$$

where $\lambda y.a(x, y(x))$ applied to $\lambda x.cookie(x)$ returns the original expression $a(x, cookie(x))$.

Syntactic splitting assigns linear order and syntactic categories to the two lambda expressions f and g. The initial syntactic category X is split by a reversal of the CCG application combinators in Section 2.2 if f and g can be recombined to give

Syntactic Category	Semantic Type	Example Phrase
$\mathsf{S}_{\mathsf{dcl}}$	$\langle ev, t\rangle$	I took it $\vdash \mathsf{S}_{\mathsf{dcl}}: \lambda e.took(i, it, e)$
S_{t}	t	I'm angry $\vdash \mathsf{S}_{\mathsf{t}}: angry(i)$
S_{wh}	$\langle e, \langle ev, t\rangle\rangle$	Who took it? $\vdash \mathsf{S}_{\mathsf{wh}}: \lambda x \lambda e.took(x, it, e)$
S_{q}	$\langle ev, t\rangle$	Did you take it? $\vdash \mathsf{S}_{\mathsf{q}}: \lambda e.Q(take(you, it, e))$
N	$\langle e, t\rangle$	cookie $\vdash \mathsf{N}: \lambda x.cookie(x)$
NP	e	John $\vdash \mathsf{NP}: john$
PP	$\langle ev, t\rangle$	on John $\vdash \mathsf{PP}: \lambda e.on(john, e)$

Figure 2: Atomic Syntactic Categories.

h with function application:

$$\{(X/Y : f \;\; Y : g), \qquad (10)$$
$$(Y : g \;\; : X\backslash Y : f) | h = f(g)\}$$

or by a reversal of the CCG composition combinators if f and g can be recombined to give h with function composition:

$$\{(X/Z : f \;\; Z/Y : g, \qquad (11)$$
$$(Z\backslash Y : g \;\; : X\backslash Z : f) | h = \lambda x.f(g(x))\}$$

Unknown category names in the result of a split (Y in (10) and Z in (11)) are labelled via a functional mapping cat from semantic type T to syntactic category:

$$\texttt{cat}(T) = \left\{ \begin{array}{ll} \texttt{Atomic}(T) & \text{if } T \in \text{Figure 2} \\ \texttt{cat}(T_1)/\texttt{cat}(T_2) & \text{if } T = \langle T_1, T_2\rangle \\ \texttt{cat}(T_1)\backslash\texttt{cat}(T_2) & \text{if } T = \langle T_1, T_2\rangle \end{array} \right\}$$

which uses the Atomic function illustrated in Figure 2 to map semantic-type to basic CCG syntactic category. As an example, the logical split in (9) supports two CCG category splits, one for each of the CCG application rules.

$$(\mathsf{NP/N}: \lambda y.a(x, y(x)),\ \mathsf{N}: \lambda x.cookie(x)) \quad (12)$$
$$(\mathsf{N}: \lambda x.cookie(x),\ \mathsf{NP\backslash N}: \lambda y.a(x, y(x))) \quad (13)$$

The parse generation algorithm $\mathcal{T}$ uses the function split to generate all CCG category pairs that are an allowed split of an input category $\mathsf{X}: h$:

$$\{(\mathsf{C_l}: m_l, \mathsf{C_r}: m_r)\} = \texttt{split}(\mathsf{X}: h),$$

and then packs a chart representation of $\{t\}$ in a top-down fashion starting with a single cell entry $\mathsf{C_m} : m$ for the top node shared by all parses $\{t\}$. For the utterance and meaning in (1) the top parse node, spanning the entire word-string, is

$$\mathsf{S}: have(you, another(x, cookie(x))).$$

$\mathcal{T}$ cycles over all cell entries in increasingly small spans and populates the chart with their splits. For any cell entry $\mathsf{X}: h$ spanning more than one word $\mathcal{T}$ generates a set of pairs representing the splits of $\mathsf{X}: h$. For each split $(\mathsf{C_l}: m_l, \mathsf{C_r}: m_r)$ and every binary partition $(w_{i:k}, w_{k:j})$ of the word-span $\mathcal{T}$ creates two new cell entries in the chart: $(\mathsf{C_l} : m_l)_{i:k}$ and $(\mathsf{C_r}: m_r)_{k:j}$.

```
Input  : Sentence $[w_1, \ldots, w_n]$, top node $\mathsf{C_m}: m$
Output: Packed parse chart Ch containing $\{t\}$
Ch $= [\ [\{\}_1, \ldots, \{\}_n]_1,\ \ldots\ , [\{\}_1, \ldots, \{\}_n]_n\ ]$
Ch$[1][n-1] = \mathsf{C_m}: m$
for $i = n, \ldots, 2;\ j = 1 \ldots (n-i)+1$ do
    for $\mathsf{X}: h \in$ Ch$[j][i]$ do
        for $(\mathsf{C_l}: m_l, \mathsf{C_r}: m_r) \in$ split$(\mathsf{X}: h)$ do
            for $k = 1, \ldots, i-1$ do
                Ch$[j][k] \leftarrow \mathsf{C_l}: m_l$
                Ch$[j+k][i-k] \leftarrow \mathsf{C_r}: m_r$
```

Algorithm 1: Generating $\{t\}$ with $\mathcal{T}$.

Algorithm 1 shows how the learner uses $\mathcal{T}$ to generate a packed chart representation of $\{t\}$ in the chart Ch. The function $\mathcal{T}$ massively overgenerates parses for any given natural language. The probabilistic parsing model introduced in Section 3 is used to choose the best parse from the overgenerated set.

5 Training

5.1 Parameter Estimation

The probabilistic model of the grammar describes a distribution over the observed training data $\mathbf{X}$, latent variables $\mathbf{S}$, and parameters θ. The goal of training is to estimate the posterior distribution:

$$p(\mathbf{S}, \theta | \mathbf{X}) = \frac{p(\mathbf{S}, \mathbf{X} | \theta) p(\theta)}{p(\mathbf{X})} \qquad (14)$$

which we do with online Variational Bayesian Expectation Maximisation (oVBEM; Sato (2001), Hoffman et al. (2010)). oVBEM is an online

Bayesian extension of the EM algorithm that accumulates observation pseudocounts $n_{a \rightarrow b}$ for each of the productions $a \rightarrow b$ in the grammar. These pseudocounts define the posterior over production probabilities as follows:

$$(\theta_{a\rightarrow b_1}, \ldots, \theta_{a\rightarrow b_{\{k,\ldots\}}})) \mid \mathbf{X}, \mathbf{S} \sim \qquad (15)$$
$$\mathrm{Dir}(\alpha H(b_1) + n_{a\rightarrow b_1}, \ldots, \sum_{j=k}^{\infty} \alpha H(b_j) + n_{a\rightarrow b_j})$$

These pseudocounts are computed in two steps:

oVBE-step For the training pair $(s_i, \{m\}_i)$ which supports the set of parses $\{t\}$, the expectation $E_{\{t\}}[a \rightarrow b]$ of each production $a \rightarrow b$ is calculated by creating a packed chart representation of $\{t\}$ and running the inside-outside algorithm. This is similar to the E-step in standard EM apart from the fact that each production is scored with the current *expectation* of its parameter weight $\hat{\theta}^{i-1}_{a\rightarrow b}$, where:

$$\hat{\theta}^{i-1}_{a\rightarrow b} = \frac{e^{\Psi\left(\alpha_a H_a(a\rightarrow b) + n^{i-1}_{a\rightarrow b}\right)}}{e^{\Psi\left(\sum_{\{b'\}}^{K} \alpha_a H_a(a\rightarrow b') + n^{i-1}_{a\rightarrow b'}\right)}} \qquad (16)$$

and Ψ is the digamma function (Beal, 2003).

oVBM-step The expectations from the oVBE step are used to update the pseudocounts in Equation 15 as follows,

$$n^{i}_{a\rightarrow b} = n^{i-1}_{a\rightarrow b} + \eta_i (N \times E_{\{t\}}[a \rightarrow b] - n^{i-1}_{a\rightarrow b}) \qquad (17)$$

where η_i is the learning rate and N is the size of the dataset.

5.2 The Training Algorithm

Now the training algorithm used to learn the lexicon Λ and pseudocounts $\{n_{a\rightarrow b}\}$ can be defined. The algorithm, shown in Algorithm 2, passes over the training data only once and one training instance at a time. For each $(s_i, \{m\}_i)$ it uses the function $\mathcal{T}$ $|\{m\}_i|$ times to generate a set of consistent parses $\{t\}'$. The lexicon is populated by using the `lex` function to read all of the lexical items off from the derivations in each $\{t\}'$. In the parameter update step, the training algorithm updates the pseudocounts associated with each of the productions $a \rightarrow b$ that have ever been seen during training according to Equation (17).

Only non-zero pseudocounts are stored in our model. The count vector is expanded with a new entry every time a new production is used. While the parameter update step cycles over all productions in $\{t\}$ it is not neccessary to store $\{t\}$, just the set of productions that it uses.

```
Input  : Corpus $D = \{(s_i, \{m\}_i) | i = 1, \ldots, N\}$,
         Function $\mathcal{T}$, Semantics to syntactic category mapping cat, function lex to read
         lexical items off derivations.
Output: Lexicon $\Lambda$, Pseudocounts $\{n_{a\rightarrow b}\}$.
$\Lambda = \{\}, \ \{t\} = \{\}$
for $i = 1, \ldots, N$ do
    $\{t\}_i = \{\}$
    for $m' \in \{m\}_i$ do
        $\mathsf{C}_{\mathsf{m'}} = \mathtt{cat}(m')$
        $\{t\}' = \mathcal{T}(s_i, \mathsf{C}_{\mathsf{m'}} : m')$
        $\{t\}_i = \{t\}_i \cup \{t\}', \ \{t\} = \{t\} \cup \{t\}'$
        $\Lambda = \Lambda \cup \mathtt{lex}(\{t\}')$
    for $a \rightarrow b \in \{t\}$ do
        $n^{i}_{a\rightarrow b} = n^{i-1}_{a\rightarrow b} + \eta_i (N \times E_{\{t\}_i}[a \rightarrow b] - n^{i-1}_{a\rightarrow b})$
```

Algorithm 2: Learning Λ and $\{n_{a\rightarrow b}\}$

6 Experimental Setup

6.1 Data

The Eve corpus, collected by Brown (1973), contains 14,124 English utterances spoken to a single child between the ages of 18 and 27 months. These have been hand annotated by Sagae et al. (2004) with labelled syntactic dependency graphs. An example annotation is shown in Figure 3.

While these annotations are designed to represent syntactic information, the parent-child relationships in the parse can also be viewed as a proxy for the predicate-argument structure of the semantics. We developed a template based deterministic procedure for mapping this predicate-argument structure onto logical expressions of the type discussed in Section 2.1. For example, the dependency graph in Figure 3 is automatically transformed into the logical expression

$$\lambda e.have(you, another(y, cookie(y)), e) \wedge on(the(z, table(z)), e), \qquad (18)$$

where e is a Davidsonian event variable used to deal with adverbial and prepositional attachments. The deterministic mapping to logical expressions uses 19 templates, three of which are used in this example: one for the verb and its arguments, one for the prepositional attachment and one (used twice) for the quantifier-noun constructions.

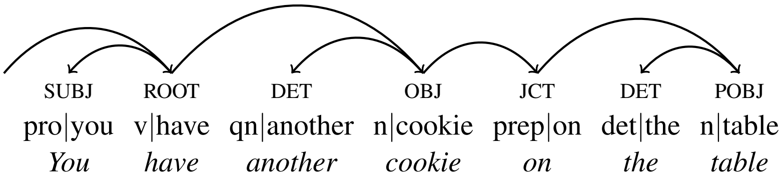
SUBJ ROOT DET OBJ JCT DET POBJ
pro|you v|have qn|another n|cookie prep|on det|the n|table
You have another cookie on the table

Figure 3: Syntactic dependency graph from Eve corpus.

This mapping from graph to logical expression makes use of a predefined dictionary of allowed, typed, logical constants. The mapping is successful for 31% of the child-directed utterances in the Eve corpus[3]. The remaining data is mostly accounted for by one-word utterances that have no straightforward interpretation in our typed logical language (e.g. *what; okay; alright; no; yeah; hmm; yes; uhhuh; mhm; thankyou*), missing verbal arguments that cannot be properly guessed from the context (largely in imperative sentences such as *drink the water*), and complex noun constructions that are hard to match with a small set of templates (e.g. as *top to a jar*). We also remove the small number of utterances containing more than 10 words for reasons of computational efficiency (see discussion in Section 8).

Following Alishahi and Stevenson (2010), we generate a context set $\{m\}_i$ for each utterance s_i by pairing that utterance with its correct logical expression along with the logical expressions of the preceding and following $(|\{m\}_i| - 1)/2$ utterances.

6.2 Base Distributions and Learning Rate

Each of the production heads a in the grammar requires a base distribution H_a and concentration parameter α_a. For word-productions the base distribution is a geometric distribution over character strings and spaces. For syntactic-productions the base distribution is defined in terms of the new category to be named by `cat` and the probability of splitting the rule by reversing either the *application* or *composition* combinators.

Semantic-productions' base distributions are defined by a probabilistic branching process conditioned on the type of the syntactic category. This distribution prefers less complex logical expressions. All concentration parameters are set to 1.0. The learning rate for parameter updates is $\eta_i = (0.8 + i)^{-0.5}$.

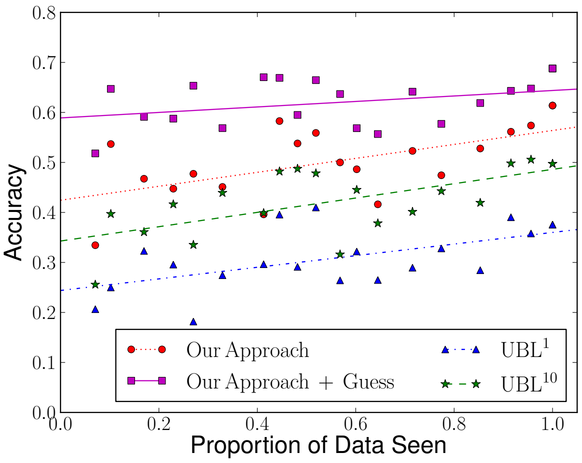

Figure 4: Meaning Prediction: Train on files $1, \ldots, n$ test on file $n + 1$.

7 Experiments

7.1 Parsing Unseen Sentences

We test the parsing model that is learnt by training on the first i files of the longitudinally ordered Eve corpus and testing on file $i + 1$, for $i = 1 \ldots 19$. For each utterance s' in the test file we use the parsing model to predict a meaning m^* and compare this to the target meaning m'. We report the proportion of utterances for which the prediction m^* is returned correctly both with and without word-meaning guessing. When a word has never been seen at training time our parser has the ability to 'guess' a typed logical meaning with placeholders for constant and predicate names.

For comparison we use the UBL semantic parser of Kwiatkowski et al. (2010) trained in a similar setting—i.e., with no language specific initialisation[4]. Figure 4 shows accuracy for our approach with and without guessing, for UBL

[3] Available at www.tomkwiat.com/resources.html

[4] Kwiatkowski et al. (2010) initialise lexical weights in their learning algorithm using corpus-wide alignment statistics across words and meaning elements. Instead we run UBL with small positive weight for all lexical items. When run with Giza++ parameter initialisations, UBL^{10} achieves 48.1% across folds compared to 49.2% for our approach.

when run over the training data once (UBL^1) and for UBL when run over the training data 10 times (UBL^{10}) as in Kwiatkowski et al. (2010). Each of the points represents accuracy on one of the 19 test files. All of these results are from parsers trained on utterances paired with a single candidate meaning. The lines of best fit show the upward trend in parser performance over time.

Despite only seeing each training instance once, our approach, due to its broader lexical search strategy, outperforms both versions of UBL which performs a greedy search in the space of lexicons and requires initialisation with cooccurence statistics between words and logical constants to guide this search. These statistics are not justified in a model of language acquisition and so they are not used here. The low performance of all systems is due largely to the sparsity of the data with 32.9% of all sentences containing a previously unseen word.

7.2 Word Learning

Due to the sparsity of the data, the training algorithm needs to be able to learn word-meanings on the basis of very few exposures. This is also a desirable feature from the perspective of modelling language acquisition as Carey and Bartlett (1978) have shown that children have the ability to learn word meanings on the basis of one, or very few, exposures through the process of *fast mapping*.

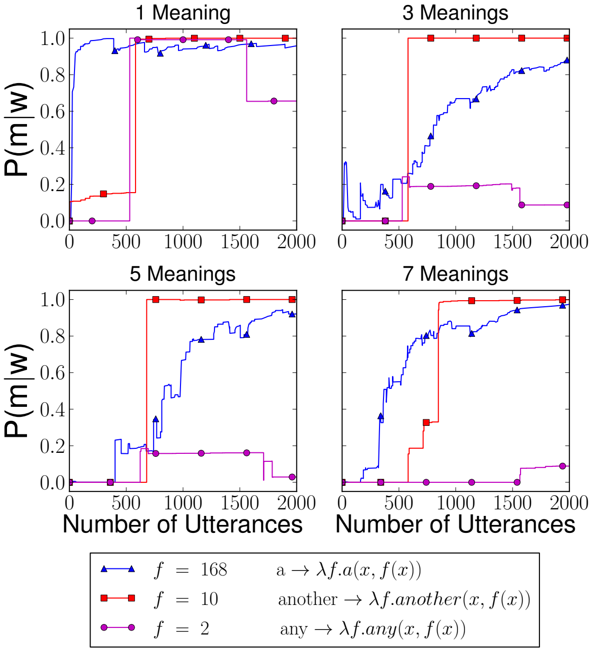

Figure 5: Learning quantifiers with frequency *f*.

Figure 5 shows the posterior probability of the correct meanings for the quantifiers ‘a’, ‘another’ and ‘any’ over the course of training with 1, 3, 5 and 7 candidate meanings for each utterance[5]. These three words are all of the same class but have very different frequencies in the training subset shown (168, 10 and 2 respectively). In all training settings, the word ‘a’ is learnt gradually from many observations but the rarer words ‘another’ and ‘any’ are learnt (when they are learnt) through large updates to the posterior on the basis of few observations. These large updates result from a *syntactic bootstrapping* effect (Gleitman, 1990). When the model has great confidence about the derivation in which an unseen lexical item occurs, the pseudocounts for that lexical item get a large update under Equation 17. This large update has a greater effect on rare words which are associated with small amounts of probability mass than it does on common ones that have already accumulated large pseudocounts. The fast learning of rare words later in learning correlates with observations of word learning in children.

7.3 Word Order Learning

Figure 6 shows the posterior probability of the correct SVO word order learnt from increasing amounts of training data. This is calculated by summing over all lexical items containing transitive verb semantics and sampling in the space of parse trees that could have generated them. With no propositional uncertainty in the training data the correct word order is learnt very quickly and stabilises. As the amount of propositional uncertainty increases, the rate at which this rule is learnt decreases. However, even in the face of ambiguous training data, the model can learn the correct word-order rule. The distribution over word orders also exhibits initial uncertainty, followed by a sharp convergence to the correct analysis. This ability to learn syntactic regularities abruptly means that our system is not subject to the criticisms that Thornton and Tesan (2007) levelled at statistical models of language acquisition—that their learning rates are too gradual.

[5] The term ‘fast mapping’ is generally used to refer to noun learning. We chose to examine quantifier learning here as there is a greater variation in quantifier frequencies. Fast mapping of nouns is also achieved.

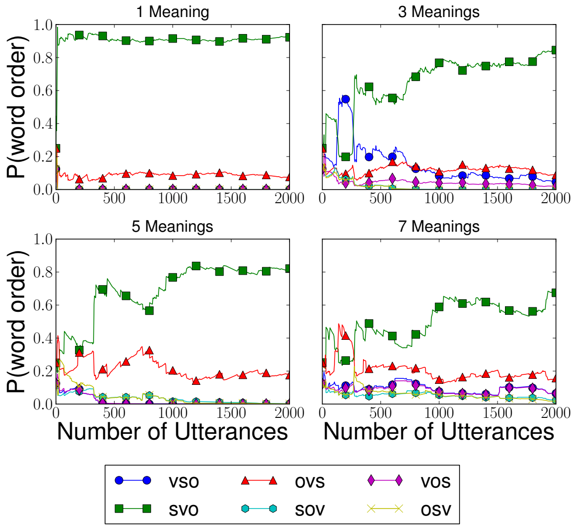

Figure 6: Learning SVO word order.

8 Discussion

We have presented an incremental model of language acquisition that learns a probabilistic CCG grammar from utterances paired with one or more potential meanings. The model assumes no language-specific knowledge, but does assume that the learner has access to language-universal correspondences between syntactic and semantic types, as well as a Bayesian prior encouraging grammars with heavy reuse of existing rules and lexical items. We have shown that this model not only outperforms a state-of-the-art semantic parser, but also exhibits learning curves similar to children's: lexical items can be acquired on a single exposure and word order is learnt suddenly rather than gradually.

Although we use a Bayesian model, our approach is different from many of the Bayesian models proposed in cognitive science and language acquisition (Xu and Tenenbaum, 2007; Goldwater et al., 2009; Frank et al., 2009; Griffiths and Tenenbaum, 2006; Griffiths, 2005; Perfors et al., 2011). These models are intended as *ideal observer* analyses, demonstrating what would be learned by a probabilistically optimal learner. Our learner uses a more cognitively plausible but approximate online learning algorithm. In this way, it is similar to other cognitively plausible approximate Bayesian learners (Pearl et al., 2010; Sanborn et al., 2010; Shi et al., 2010).

Of course, despite the incremental nature of our learning algorithm, there are still many aspects that could be criticized as cognitively implausible. In particular, it generates all parses consistent with each training instance, which can be both memory- and processor-intensive. It is unlikely that children do this once they have learnt at least some of the target language. In future, we plan to investigate more efficient parameter estimation methods. One possibility would be an approximate oVBEM algorithm in which the expectations in Equation 17 are calculated according to a high probability subset of the parses $\{t\}$. Another option would be particle filtering, which has been investigated as a cognitively plausible method for approximate Bayesian inference (Shi et al., 2010; Levy et al., 2009; Sanborn et al., 2010).

As a crude approximation to the context in which an utterance is heard, the logical representations of meaning that we present to the learner are also open to criticism. However, Steedman (2002) argues that children do have access to structured meaning representations from a much older apparatus used for planning actions and we wish to eventually ground these in sensory input.

Despite the limitations listed above, our approach makes several important contributions to the computational study of language acquisition. It is the first model to learn syntax and semantics concurrently; previous systems (Villavicencio, 2002; Buttery, 2006) learnt categorial grammars from sentences where all word meanings were known. Our model is also the first to be evaluated by parsing sentences onto their meanings, in contrast to the work mentioned above and that of Gibson and Wexler (1994), Siskind (1992) Sakas and Fodor (2001), and Yang (2002). These all evaluate their learners on the basis of a small number of predefined syntactic parameters.

Finally, our work addresses a misunderstanding about statistical learners—that their learning curves must be gradual (Thornton and Tesan, 2007). By demonstrating sudden learning of word order and fast mapping, our model shows that statistical learners can account for sudden changes in children's grammars. In future, we hope to extend these results by examining other learning behaviors and testing the model on other languages.

9 Acknowledgements

We thank Mark Johnson for suggesting an analysis of learning rates. This work was funded by the ERC Advanced Fellowship 24952 GramPlus and EU IP grant EC-FP7-270273 Xperience.

References

Alishahi and Stevenson, S. (2008). A computational model for early argument structure acquisition. *Cognitive Science*, 32:5:789–834.

Alishahi, A. and Stevenson, S. (2010). Learning general properties of semantic roles from usage data: a computational model. *Language and Cognitive Processes*, 25:1.

Beal, M. J. (2003). Variational algorithms for approximate Bayesian inference. Technical report, Gatsby Institute, UCL.

Börschinger, B., Jones, B. K., and Johnson, M. (2011). Reducing grounded learning tasks to grammatical inference. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1416–1425, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Brown, R. (1973). *A First Language: the Early Stages*. Harvard University Press, Cambridge MA.

Buttery, P. J. (2006). Computational models for first language acquisition. Technical Report UCAM-CL-TR-675, University of Cambridge, Computer Laboratory.

Carey, S. and Bartlett, E. (1978). Acquring a single new word. *Papers and Reports on Child Language Development*, 15.

Chen, D. L., Kim, J., and Mooney, R. J. (2010). Training a multilingual sportscaster: Using perceptual context to learn language. *J. Artif. Intell. Res. (JAIR)*, 37:397–435.

Fazly, A., Alishahi, A., and Stevenson, S. (2010). A probabilistic computational model of cross-situational word learning. *Cognitive Science*, 34(6):1017–1063.

Frank, M., Goodman, S., and Tenenbaum, J. (2009). Using speakers referential intentions to model early cross-situational word learning. *Psychological Science*, 20(5):578–585.

Frank, M. C., Goodman, N. D., and Tenenbaum, J. B. (2008). A bayesian framework for cross-situational word-learning. *Advances in Neural Information Processing Systems 20*.

Gibson, E. and Wexler, K. (1994). Triggers. *Linguistic Inquiry*, 25:355–407.

Gleitman, L. (1990). The structural sources of verb meanings. *Language Acquisition*, 1:1–55.

Goldwater, S., Griffiths, T. L., and Johnson, M. (2009). A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.

Griffiths, T. L., . T. J. B. (2005). Structure and strength in causal induction. *Cognitive Psychology*, 51:354–384.

Griffiths, T. L. and Tenenbaum, J. B. (2006). Optimal predictions in everyday cognition. *Psychological Science*.

Hoffman, M., Blei, D. M., and Bach, F. (2010). Online learning for latent dirichlet allocation. In *NIPS*.

Kate, R. J. and Mooney, R. J. (2007). Learning language semantics from ambiguous supervision. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2010). Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the Conference on Emperical Methods in Natural Language Processing*.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2011). Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the Conference on Emperical Methods in Natural Language Processing*.

Levy, R., Reali, F., and Griffiths, T. (2009). Modeling the effects of memory on human online sentence processing with particle filters. In *Advances in Neural Information Processing Systems 21*.

Lu, W., Ng, H. T., Lee, W. S., and Zettlemoyer, L. S. (2008). A generative model for parsing natural language to meaning representations. In *Proceedings of The Conference on Empirical Methods in Natural Language Processing*.

MacWhinney, B. (2000). *The CHILDES project: tools for analyzing talk*. Lawrence Erlbaum, Mahwah, NJ u.a. EN.

Maurits, L., Perfors, A., and Navarro, D. (2009). Joint acquisition of word order and word reference. In *Proceedings of the 31th Annual Conference of the Cognitive Science Society*.

Pearl, L., Goldwater, S., and Steyvers, M. (2010). How ideal are we? Incorporating human limitations into Bayesian models of word segmentation. pages 315–326, Somerville, MA. Cascadilla Press.

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

Sagae, K., MacWhinney, B., and Lavie, A. (2004). Adding syntactic annotations to transcripts of parent-child dialogs. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*. Lisbon, LREC.

Sakas, W. and Fodor, J. D. (2001). The structural triggers learner. In Bertolo, S., editor, *Language Acquisition and Learnability*, pages 172–233. Cambridge University Press, Cambridge.

Sanborn, A. N., Griffiths, T. L., and Navarro, D. J. (2010). Rational approximations to rational models: Alternative algorithms for category learning. *Psychological Review*.

Sato, M. (2001). Online model selection based on the variational bayes. *Neural Computation*, 13(7):1649–1681.

Shi, L., Griffiths, T. L., Feldman, N. H., and Sanborn, A. N. (2010). Exemplar models as a mechanism for performing bayesian inference. *Psychonomic Bulletin & Review*, 17(4):443–464.

Siskind, J. M. (1992). *Naive Physics, Event Perception, Lexical Semantics, and Language Acquisition*. PhD thesis, Massachusetts Institute of Technology.

Siskind, J. M. (1996). A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, 61(1-2):1–38.

Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA.

Steedman, M. (2002). Plans, affordances, and combinatory grammar. *Linguistics and Philosophy*, 25.

Thornton, R. and Tesan, G. (2007). Categorical acquisition: Parameter setting in universal grammar. *Biolinguistics*, 1.

Villavicencio, A. (2002). The acquisition of a unification-based generalised categorial grammar. Technical Report UCAM-CL-TR-533, University of Cambridge, Computer Laboratory.

Wong, Y. W. and Mooney, R. (2006). Learning for semantic parsing with statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL*.

Wong, Y. W. and Mooney, R. (2007). Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the Association for Computational Linguistics*.

Xu, F. and Tenenbaum, J. B. (2007). Word learning as Bayesian inference. *Psychological Review*, 114:245–272.

Yang, C. (2002). *Knowledge and Learning in Natural Language*. Oxford University Press, Oxford.

Yu, C. and Ballard, D. H. (2007). A unified model of early word learning: Integrating statistical and social cues. *Neurocomputing*, 70(13-15):2149 – 2165.

Zettlemoyer, L. S. and Collins, M. (2005). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.

Zettlemoyer, L. S. and Collins, M. (2007). Online learning of relaxed CCG grammars for parsing to logical form. In *Proc. of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Zettlemoyer, L. S. and Collins, M. (2009). Learning context-dependent mappings from sentences to logical form. In *Proceedings of The Joint Conference of the Association for Computational Linguistics and International Joint Conference on Natural Language Processing*.