Treebank Grammars and Other Infinite Parameter Models

Detlef Prescher, Remko Scha, Khalil Sima'an and Andreas Zollmann

Institute for Logic, Language & Computation
University of Amsterdam

Abstract. Syntactic disambiguators for natural language often use "Treebank Grammars": probabilistic grammars which are directly projected from an annotated corpus. In this paper we show that for describing these systems in the framework of Estimation Theory, we must generalize this theory so that it allows for an infinite number of parameters. Embracing this generalization will also bring the justification of statistical smoothing techniques within the scope of Estimation Theory.

1 Introduction

A serious problem in Natural Language Processing is *ambiguity*. Formal grammars of natural languages are usually designed to describe the complete set of syntactic possibilities that a language offers. They tend to assign a large set of possible syntactic structures to virtually every utterance, although only one of these would be perceived by a human language user. To resolve this ambiguity, a parser must be able to rank the different analyzes of an utterance, and select one of them as the most plausible one. To this end, natural language parsers use *probabilistic grammars* [1–5], which assign a probability to every syntactic analysis they generate. Disambiguation decisions are then based on the probabilities of the different alternative structures of an input utterance.

This paper discusses parsing/disambiguation systems which are based on probabilistic grammars. In particular, it investigates how the rules and/or the probabilities of such grammars may be learned from a *treebank*, i.e., from a corpus of utterances which have been annotated with "correct" syntactic structures. Since the early nineties, several systems have been developed which assume a *grammar formalism* (i.e., the *format* for the grammar rules), and then learn the actual rules as well as their probabilities from the corpus. Such systems may be said to learn a *treebank grammar*. Since their estimators learn the whole grammar (rather than just the rule probabilities of a given grammar), they deal with an infinite rule set: the set of all possible rules allowed by the grammar formalism. We will call this infinite rule set an "infinite grammar", although this is dangerous terminology: in the parlance of theoretical linguistics, finiteness is considered an essential property of grammars. Clearly, this is a non-trivial extension. Though the "infinite grammars" which are learned by implementable estimators are necessarily representable by finite means, there does not need to be a finite bound on the size of this representation.

We will show that treebank grammars based on sufficiently rich grammar formalisms can approximate arbitrary parse-tree probability distributions arbitrarily closely. But this power has a cost: if we allow the training corpus to grow indefinitely, the grammar that is learned may also grow arbitrarily large.

We conclude the paper by discussing the well-known "sparse data problem" in the context of natural language parsing. As we know from Zipf's law, new words, new categorizations of existing words, and new syntactic combinations should keep appearing as the corpus grows. This implies that probability estimation by means of frequency counts will *never* be an adequate predictor for future utterances. To deal with this problem, existing systems employ techniques for *smoothing* their probability estimates: they reserve probability mass for the unseen events and redistribute this mass on the basis of various heuristics [6–8]. To justify smoothing techniques in the framework of Estimation Theory, we need precisely the generalization that we introduced before: smoothing is concerned with estimating the values for an *infinite* dimensional parameter vector.

Before we discuss these matters in more detail, the next section introduces some basic concepts about formal and probabilistic grammars, and it reviews the current practice in treebank-based probabilistic parsing.

2 Current Practice in Corpus-Based NLP

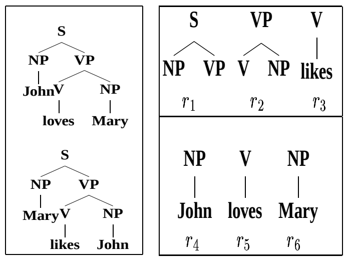
The first subsection below describes, in very general terms, the kind of grammar formalisms being used in probabilistic parsing/disambiguation systems. The second subsection then gives an informal impression about the way in which the rules and/or probabilities of such grammars are learned from treebanks. More precise investigations of that issue constitute the topic of the rest of the paper.

2.1 Probabilistic Grammars

Probabilistic grammars are based on *rewrite grammars*. In particular, many are based on *Context-Free Grammars* (CFGs). A CFG is a tuple $\langle \mathcal{N}, \mathcal{W}, \mathcal{R}, S \rangle$, where $\mathcal{N}$ and $\mathcal{W}$ are disjoint finite sets of *non-terminal* and *terminal* symbols, respectively; $S \in \mathcal{N}$ is a distinguished start symbol, and $\mathcal{R}$ is a finite set of *productions*, of the form $A \rightarrow \beta$, where $\alpha \in \mathcal{N}$ and β is a sequence of elements from $\mathcal{N} \cup \mathcal{W}$. A CFG defines a rewrite system, where S is rewritten, in consecutive steps, using the productions; at each step the result of rewriting is a sequence of symbols from $\mathcal{N} \cup \mathcal{W}$. Such a rewrite process, called a *derivation*, terminates when the resulting sequence consists of only terminal symbols; such a sequence is called a *sentence* of the language generated by the grammar.

The graphical representation of a CFG-derivation is at the same time the parse-tree of the sentence. In other grammar formalisms, such as *Tree-Substitution Grammars* (TSGs) and *Tree-Adjoining Grammars* (TAGs) [9], the one-to-one correspondence between sentence structure and derivation structure is given up. Grammars of this sort resemble CFGs, with one important difference: the rewrite process explicitly generates parse-trees rather than symbol sequences. The production rules now consist of a non-terminal symbol on the left hand side of the arrow, and a *an elementary-tree* on the right hand side; the rule specifies that the rewrite process may select a node which is labeled with this non-terminal, and expand it into a copy of this tree.

A *Probabilistic CFG* (PCFG) extends a CFG with a probability function $\pi : \mathcal{R} \rightarrow [0,1]$, which assigns to each production a probability value. The probability of derivation $r_1, \ldots, r_n$, involving productions $\forall i : r_i \in \mathcal{R}$, is calculated under an independence



Comments: Suppose the upper (lower) tree occurs 3 (resp. 7) times in the treebank. The relative frequency estimate for the productions extracted from this treebank would be: $\pi(r_1) = 1.0$, $\pi(r_2) = 1.0$, $\pi(r_3) = 0.7$, $\pi(r_4) = 0.5$, $\pi(r_5) = 0.3$ and $\pi(r_6) = 0.5$. Note that the counts of a production $A \rightarrow \alpha$ is normalized by the total count of productions with left hand side symbol A, e.g. $\pi(r_4) = \frac{frequency(r_4)}{frequency(r_4)+frequency(r_6)}$.

Fig. 1. (**left**) A treebank, (**right**) The treebank CFG (see comments on probabilities).

assumption between the individual productions, i.e. $p(r_1,\dots,r_n) = \prod_{i=1}^{n} \pi(r_i)$. The probability of a sentence u ($p(u)$) is given by the sum of the probabilities of all derivations that generate u.

Like PCFGs, *Stochastic TSGs* (STSGs) and *Stochastic TAGs* (STAGs) extend their conventional counterparts with a probability function over the productions (elementary-trees). Note, however, that these formalisms allow multiple derivations of the same parse-tree. Hence, the probability of a parse-tree t, $p(t)$, is now the sum of the probabilities of the derivations that generate t.

2.2 Treebank grammars and Language Models

Two issues play a role in how probabilistic grammars are actually built (1) the way the symbolic grammar is obtained, (2) how the probabilities of the grammar productions are estimated from an actual corpus. The earliest probabilistic parsing systems concentrated on estimating the probabilities of the productions of *broad-coverage, linguistically motivated, manually constructed grammars* e.g. [10]. In contrast, state-of-the-art approaches e.g. [11, 12, 3, 13] have abandoned the broad-coverage linguistic grammars for the sake of so-called *treebank grammars* (a term coined by Charniak [14]). The characteristic property of a treebank grammar is that its rules are projected directly from a corpus consisting of a sample of utterances that are manually annotated with the correct syntactic structures (a so-called *treebank*). See the example in Figure 1.

When seen from a broader perspective than parsing, treebank grammars were inspired by the statistical *corpus-based approaches* that underly *Statistical Language Models (SLMs)* that are so prevalent in speech-recognition [15]. In its purest form, a statistical language model stands for a probability distribution over a set of sequences of words (utterances). This statistical notion can be seen as an extension of the formal notion of a language, i.e. a set of utterances. In speech-recognition, a language model employs a probabilistic regular grammar (Finite State Automaton) where the probability of every word is conditioned on the preceding words in the utterance, i.e. a so called Markov grammar. Markov grammars can be seen as the first treebank grammars; an nth-order Markov grammar is obtained by collecting from a large corpus all the *(n+1)-grams* together with their frequency counts.

To arrive at a language model, a probabilistic grammar must generate a *single probability distribution* over the set of utterance-parse pairs such that the sum of the probabilities of all such pairs does not exceed one. One way to construct language models is through *generative probabilistic grammars*, where the probability of a production $A \rightarrow \alpha$ is conditioned on its left-hand side A, i.e. $\pi(A \rightarrow \alpha \mid A)$, which implies the following constraints on the probability estimates: $\sum_{\alpha} \pi(A \rightarrow \alpha \mid A) = 1$.

This paper studies Treebank and Corpus-based Grammars from the point of view of statistical Estimation Theory. We shall see that a satisfactory account of the treebank grammar approach requires some changes in this theory.

3 Standard Estimation Theory

In the following, we adhere to [16]. Our basic setting is a random experiment with an underlying sample space and a probability measure p. A *random experiment* is an experiment whose outcome cannot be predicted with certainty. A *random variable* X is as a measurement in the context of the random experiment; it is random in the sense that its value depends on the outcome of the experiment. Each time the experiment is run, an outcome of the experiment occurs, and X takes on a value $x \in S$. The range S of X is called the *sample space* of X. Assuming that S is countable, X is a discrete variable characterized by its *probability distribution* $p_X : S \to [0,1]$, where $p_X(x) := p(X = x)$ for all $x \in S$. Finally, we assume that X's distribution is parameterized, i.e., dependent on some *parameters*, i.e., non-random quantities in the random experiment that, once chosen, remain constant. In most cases, (one or more of) the parameters are unknown, and must be estimated from the outcome vector X. This is the most important of all statistical problems, and is the subject of this section.

3.1 Random Samples, Statistics, and Estimators

Let $X_1, \ldots, X_n$ be independent random variables with the same distribution as the variable X from the above setting. Then $\langle X_1, \ldots, X_n \rangle$ is called a *random sample* of X. The outcome of the variables in a particular trial, $\langle x_1, \ldots, x_n \rangle$, is called an *observation sequence*. A *statistic* is a random variable derived from the random sample $\langle X_1, \ldots, X_n \rangle$. Examples for statistics are the *sample mean* $\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i$ and the *sample variance* $s_n^2 = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \overline{X}_n)^2$.

Guessing the distribution of X from an observation sequence involves assumptions on what kind of distributions are admissible. More formally, estimation is based on a *model* $\mathcal{M}$—the set of admissible distributions. The 'true' distribution to be estimated from the observation sequence is assumed to be an instance of $\mathcal{M}$. In standard estimation theory, the model $\mathcal{M}$ is often characterized by a finite-dimensional set $\Theta \subseteq \mathbb{R}^k$ of *parameter vectors*, such that there is a one-to-one correspondence between Θ and $\mathcal{M}$. An example is the model of all Gaussian distributions, where each distribution is characterized by the parameters $\langle \mu, \sigma \rangle \in \mathbb{R} \times \mathbb{R}_+$ (μ being the distribution's expected value and σ its standard deviation). The process of estimation can now be described in terms of parameters. Let p_θ be the distribution corresponding to parameter vector $\theta \in \Theta$ and θ^* the parameter vector of the "true" distribution $p_X = p_{\theta^*}$.

An *estimator* est_n *of* θ^* is a statistic from the random sample $\langle X_1, \ldots, X_n \rangle$ whose range is Θ. Thus, the estimator is a random variable with a distribution, an expected value, and so on. The observed value $\text{est}_n(\langle x_1, \ldots, x_n \rangle)$ of the outcome of a particular random experiment is called the *estimate* of θ^* from the observations $\langle x_1, \ldots, x_n \rangle$. For example, the prominent *maximum-likelihood method* aims at maximizing the probability of the observation sequence: $\hat{\theta} = \arg\max_\theta \prod_{i=1}^{n} p_\theta(x_i)$. If such an estimate $\hat{\theta}$ exists for each possible observation sequence, the corresponding statistic $\hat{\theta}(\langle X_1, \ldots, X_n \rangle)$ is called a *maximum-likelihood estimator* of θ^*.

3.2 Properties of Estimators in Standard-Estimation Theory

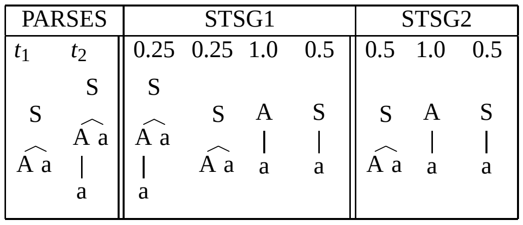
When investigating the properties of estimators, all distributions $p \in \mathcal{M}$ that could underly X are considered. This means that X is distributed according to a distribution p_θ depending on a parameter vector $\theta \in \Theta$. In other words, we implicitly assume in the following that $p_X = p_\theta$.

Bias and Consistency. The (random) *error* is the difference between the estimator and the parameter vector: $\mathrm{est}_n - \theta$. The expected value of the error is known as the *bias*: $\mathrm{bias}_\theta(\mathrm{est}_n) = \mathrm{E}(\mathrm{est}_n - \theta)$. If $\mathrm{bias}_\theta(\mathrm{est}_n) = 0$ (i.e., $\mathrm{E}(\mathrm{est}_n) = \theta$) for all $\theta \in \Theta$, then est_n is said to be *unbiased*. We also expect our estimators to improve, in some sense, as n increases. As the quality of an estimator can be measured by a *loss function* (e.g. by $\mathrm{loss}_\theta(\mathrm{est}_n) = ||\mathrm{est}_n - \theta||^2$), a sequence of estimators is called *consistent* if for each $\theta \in \Theta$, the expected loss approaches zero as n goes to infinity: $\lim_{n\to\infty} \mathrm{E}(\mathrm{loss}_\theta(\mathrm{est}_n)) = 0$.

Minimal Sufficiency. An estimator is a statistic with the parameter space Θ as its range. A basic property of a statistic is sufficiency: A statistic $U = h(X_1, \ldots, X_n)$ is called *sufficient for a parameter* θ if U contains all of the information about θ that is available in the entire random sample[1]. Obviously, the entire random sample is trivially sufficient. There is, however, usually a statistic U that is sufficient *and has a smaller dimension*, so that we can achieve real data reduction. Naturally, we would like to find the statistic U that has the smallest dimension possible.

In summary, estimators aim at guessing (the parameters of) the probability distribution that has been used to generate a given corpus. Standard estimation theory assumes that: (1) the parameter space is finite-dimensional, (2) the true distribution is in the model, and (3) estimation is/should be based on minimal sufficient statistics. In the following sections, we investigate whether these assumptions hold for treebank grammars.

4 The Parameters of Probabilistic Parsing



Comments: Both STSGs generate the same parse distribution $\{\langle t_1, 0.5\rangle, \langle t_2, 0.5\rangle\}$. (Note that the left STSG generates t_2 in two different derivations each contributing 0.25 to its probability: (1) (S$\rightarrow$ A a$\rightarrow$ a a) and (2) combine (S$\rightarrow$ A a) with (A$\rightarrow$a a).

Fig. 2. Two STSGs generating same parse distribution. See comments.

In order to describe the learning of probabilistic parsers from an Estimation Theory perspective, we first need to pin down the parameters that are being estimated. In

[1] Sufficiency is related to the concept of data reduction: If we can find a sufficient statistic U taking values in a $m-$dimensional space, then we reduced the original data vector $\langle X_1, \ldots, X_n\rangle$ (whose dimension n is usually large) to the statistic U (whose dimension m is usually much smaller) with no loss of information.

general, probability estimation from a corpus is used for two tasks: (1) estimating the production probabilities of an a priori fixed probabilistic grammar, or (2) estimating a probability distribution over parses. In the first case, the actual parameters are the *production probabilities*, while in the latter these are the *parse probabilities*. Remarkably, for estimating a certain probability distribution over parses one could employ different (kinds of) probabilistic grammars as shown in Figure 2. Hence, the two tasks, estimating a probabilistic grammar or estimating a parse distribution, are not necessarily equivalent.

What is then the appropriate kind of parameters for ambiguity resolution? Note that the role of the probabilities is to rank the different parses in order to facilitate the selection of the most probable one. Clearly, if we had access to estimates of parse probabilities (a parse distribution), we would have *sufficient* means for disambiguation. Hence, the parameters that are subject to estimation should be *parse probabilities*.

Note that for estimating production probabilities one must pin down a target probabilistic grammar prior to estimation. Pinning down the grammar prior to estimation implies a very strong assumption: the chosen grammar reflects the exact nature of natural language syntax. Clearly, this assumption does not always hold for existing formal grammars. If parse probabilities are the subject of estimation, why then does existing work in NLP highlight the estimation of production probabilities?

The reason is quite simple: the direct estimation of parse probabilities implies an *infinite* dimensional vector of parameters, i.e. the parses. This does not go well with Estimation Theory, which assumes a finite dimensional parameter vector ($\langle\theta_1,\ldots,\theta_n\rangle$). Work on probabilistic parsing tackles this infinity by assuming that a known probabilistic grammar, with a finite set of productions, generates the parse distribution. Let be given a set of parses $\mathcal{T}$ over finite sets of nonterminal $\mathcal{N}$ and terminal $\mathcal{W}$ symbols. If we assume now that the set of parses $\mathcal{T}$ is generated by a grammar G= $\langle\mathcal{N},\mathcal{W},\mathcal{R},S\rangle$, e.g. a CFG, then each parse t is generated by a G-derivation. For a given treebank $T_1,\ldots,T_x$, the Likelihood under a parse probability function p is given by $L(T_1,\ldots,T_x,\ p) = \prod_{i=1}^{x} p(T_i)$. If for all $1 \leq i \leq x$, T_i is derived by some derivation $r_1,\ldots r_n$, then $p(T_i) = \prod_{j=1}^{n} \pi(r_i)$. Therefore, the commonly used estimator, the Maximum-Likelihood of a treebank, simply reduces to the Maximum-Likelihood over a "corpus of independent grammar productions". This means that under the assumption that a given probabilistic grammar generates the distribution over $\mathcal{T}$, estimating the production probabilities of this grammar will also yield parse estimates. The question is, of course, whether the assumption of knowing the grammar productions is a reasonable one for natural language parsing? As we see in the sequel, this assumption has been found to be too strong for dealing with various problems, and a different approach has emerged, *Treebank Grammars*.

There are a few important realizations that arise in this section: (1) the actual goal is to estimate a (possibly) *infinite* parse distribution (of which the treebank is a finite sample), (2) by assuming that a given grammar generates the parse distribution, a structure is imposed on the infinite dimensional parse-space such that the dimension of the parameter vector is reduced to a finite dimension (the number of grammar productions).

5 Treebank Grammars and Estimation Theory

The common feature of treebank-based approaches to stochastic parsing is that they do not assume an *a priori* grammar, but learn a *treebank grammar* from a corpus. In what follows, we confront treebank-based approaches to stochastic parsing with estimation theory. We briefly review so-called *Data-Oriented Parsing* (DOP), one of the earliest and most radical approaches to treebank grammars: it suggests using an annotated corpus directly as a stochastic grammar [11]. After reviewing DOP, we point out the incompatibilities between DOP and Estimation Theory as a starting point for our further discussion concerning other treebank grammar approaches.

5.1 Data-Oriented Parsing

Like other treebank models, DOP acquires from a treebank TB a finite set $\mathcal{F}$ of productions, called *fragments*[2] or *subtrees*, together with their probability estimates. An important feature of DOP is that the set $\mathcal{F}$ consists of *all* fragments of the treebank trees. The set of fragments $\mathcal{F}$ is employed as an *STSG* with the same start symbol, nonterminal and terminal sets as the treebank trees (see section 2 for STSGs). Like other STSGs, a DOP grammar is based on fragment probabilities that allow the calculation of derivation and parse probabilities:

Fragment probability: To each $t \in \mathcal{F}$, a real number $\pi(t) \geq 0$ is assigned, such that for non-terminal label A, π induces a probability distribution on the set of fragments t whose root label $\mathrm{root}(t)$ is A. I.e.: $\sum_{t\,:\,\mathrm{root}(t)=A} \pi(t) = 1$.

Derivation probability: The probability of a derivation d of a parse-tree is the product of its fragment probabilities:

(1) $$p(d) = \prod_{t \in \mathcal{F}} \pi(t)^{f_t(d)}$$

(Here $f_t(d)$ is the number of times the fragment t occurs in the derivation d.)

Tree probability: The probability of a parse-tree x with a set of derivations $\mathcal{D}(x)$ is the sum of the probabilities of its derivations:

(2) $$p(x) = \sum_{d \in \mathcal{D}(x)} p(d)$$

5.2 Estimation Theory and DOP

Recall that the focus of estimation for parsing is a probability distribution over parses. The statistics employs a *corpus*, that has been generated in accordance with some unknown probability distribution, in order to infer that distribution.

Given a probability model comprising several distributions which might have generated the corpus, an *estimation method* selects one *instance* of the probability model as

[2] A connected subgraph of a treebank tree t is called a *fragment* iff it consists of one or more context-free productions.

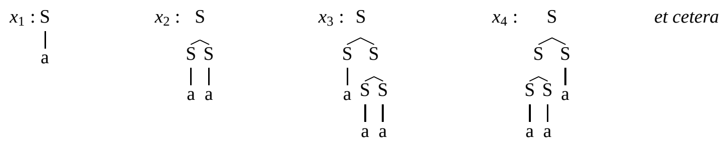
its best guess about the original distribution. From the perspective of DOP, a probability model simply bundles specific probability distributions on the set $\mathcal{X}$ of derivable parse trees, where each model instance p is induced by a function π on a given set $\mathcal{F}$ of tree fragments such that the equations (1) and (2) are satisfied. In other words, a *DOP model* is defined on the basis of a given set $\mathcal{F}$ of tree fragments such that

$$\mathcal{M}_{\mathrm{DOP}}(\mathcal{F}) = \left\{ p \in \mathcal{M}(\mathcal{X}) \;\middle|\; \exists \pi \text{ such that } \forall x \in \mathcal{X}:\; p(x) = \sum_{d \in \mathcal{D}(x)} \prod_{t \in \mathcal{F}} \pi(t)^{f_t(d)} \right\}$$

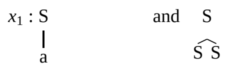
Moreover, given such a DOP model, an estimator est_n for this model can be described as a statistic from a random sample $\langle X_1, \ldots, X_n \rangle$, having the DOP model as its range. In other words, an observed value $\mathrm{est}_n(\langle x_1, \ldots, x_n \rangle)$ of this estimator is an instance of $\mathcal{M}_{\mathrm{DOP}}(\mathcal{F})$, thereby estimating this model instance by exploiting the corpus $\langle x_1, \ldots, x_n \rangle$. This result can be expressed by:

$$\langle x_1, \ldots, x_n \rangle \longmapsto \mathrm{est}_n(\langle x_1, \ldots, x_n \rangle) \quad \in \; \mathcal{M}_{\mathrm{DOP}}(\mathcal{F})$$

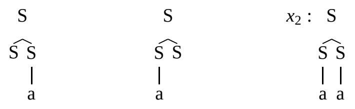
To study the asymptotic behavior of DOP, we will now investigate a sequence $\mathrm{est}_1, \mathrm{est}_2, \mathrm{est}_3, \ldots$ of DOP estimators. Starting with an example, the following figure displays the parse trees that are derivable by the context-free grammar with the rules $S \to SS$ and $S \to a$.



From the perspective of DOP, however, each context-free grammar *is a* DOP grammar. Each CFG rule can be interpreted as a local tree, and the set $\mathcal{F}$ of tree fragments of the DOP grammar simply consists of these local trees. In our example, $\mathcal{F}$ consists of the two fragments:



From the perspective of estimation theory, we are now assuming that a model instance of this DOP grammar is used for sampling. Then the specific corpus $\langle x_1, \ldots, x_n \rangle$ is clearly one of the possible observations of the random sample $\langle X_1, \ldots, X_n \rangle$. The crucial point is now that the set of fragments that are read off from $\langle x_1, \ldots, x_n \rangle$ includes fragments that are *not* in $\mathcal{F}$ — at least if we assume that the corpus size n is big enough. Here, we check e.g. for $n \geq 2$ that



are not fragments of the DOP grammar that generated the corpus. There is nothing special in the given example that stops us from generalizing this finding:

By assuming that the treebank is sampled from a DOP grammar that is interesting enough, i.e., that can be used to derive an infinite number of trees, then one aspect of the asymptotic behavior of DOP estimation is that *the symbolic DOP grammar grows as the treebank grows*. As an effect, in the limit of the treebank size, DOP risks learning an arbitrarily large grammar.

Although it is obvious that this phenomenon does not occur for simpler grammar formalisms (like PCFGs and probabilistic finite-state automata), we conjecture that it also occurs for other *higher-order* treebank grammars (like unification-based grammar formalisms). We think also that there is a connection to the task of estimating *a priori* grammars in the framework of *unsupervised learning*.

Summarizing the results of this section, we started out by learning tree distributions, and we find out that treebank grammars are indeed doing so. A surprise is, however, that DOP learns an arbitrarily large grammar (in the limit). We conjecture, however, that this is a more general problem for other kinds of treebank grammars. As we will point out in the next section, this is an extreme case of other common phenomena in Natural Language Processing.

6 Related Corpus-based Methods

As we have seen, theoretically speaking, DOP aims at estimating the probabilities of an infinite dimensional parameter vector. This implies that DOP estimation is incompatible with Estimation Theory. In this section we will show that this is not unique for DOP; other commonly used corpus-based methods based on Markov Models and other treebank grammars employ smoothing techniques that have the same property: they all assume an infinite grammar.

A common problem in speech and language processing is the problem of "unknown words", i.e. words that have not occurred in the training corpus. As the corpus grows, novel words will occur all the time. Open category words, such as proper nouns and compound nouns, are a common problem, but even novel verbs are made up on the fly all the time (e.g. "googling someone" for searching on the web). The situation is more severe with languages with rich morphology (e.g. Arabic or Turkish). The problem of unknown words has been linked to Zipf's law [17]: there is always a considerable tail of very low frequency phenomena to be expected to occur in the future. This problem has major consequences for the estimation of production probabilities, because one cannot determine the finite set of allowed words (terminal symbols) a priori to estimation.

For language models based on Markov processes over word sequences (e.g. Hidden Markov Models for speech-recognition) [15], it is not possible to fix a finite set of parameters (bigrams or trigrams). Therefore, Markov Grammars are usually obtained directly from a large corpus (n-grams together with their estimates). Theoretically speaking, we must assume an infinite dimensional parameter vector if we are to explain the asymptotic estimation properties of the corpus-based Markov models.

We may stretch the unknown word problem one step further to unknown category words, i.e. words for which some part-of-speech tag categories are not in the corpus. Similarly, unknown productions could occur: in the well-known Penn Wall Street Journal treebank [18] many productions occur only once, hinting at the fact that other novel productions are likely to occur in new utterances.

The problem of the “unknown events” constitutes the smallest common deviser for all language (and other) modeling activities. Various smoothing techniques have been developed for dealing with this problem [8, 7]. Smoothing techniques are used as follows:

Estimate the parameters of a (finite) grammar, including a special symbol UNKNOWN that stands for a category of unknown events.
Use a mapping from an input word to itself (i.e. own category) if it is known, or else to the UNKNOWN category.
Reserve and distribute probability mass to the map into UNKNOWN.

Theoretically speaking, the second step (the mapping) can only be described by an infinite set of rules that maps a novel word to its UNKNOWN.

7 Discussion and conclusion

As we have seen in the two preceding sections, it seems necessary and reasonable to lift certain restrictions on the grammar assumed to generate a natural language. Recall that for an ordinary CFG (cf. Section 2), the nonterminals $\mathcal{N}$, terminals $\mathcal{W}$, as well as the productions $\mathcal{R}$ are required to be finite sets. Which finiteness-restrictions should we abolish to fit the treebank models?

In Section 5, we observed that the number of productions (subtrees) of DOP treebank grammars may increase as training treebank size increases. In the limit, DOP then seems to estimate a infinite grammar; however, so far we were not able to express this behavior in a formal way. Allowing for $\mathcal{R}$ to be a *countable* set enables the generation of new tree distributions. Thus, the model $\mathcal{M}_{\text{DOP}}(\mathcal{R})$ (where the productions $\mathcal{R}$ are a (countable) set of DOP fragments) can now be chosen much richer. Consider the special case that $\mathcal{R}$ is chosen as the set of all fragments from a given countable set T of full parse trees. Then, given a DOP estimation method $\langle \text{est}_i \rangle_{i \in \mathbb{N}_+}$, the following desirable property holds:

for any infinite sequence $\langle t_i \rangle_{i \in \mathbb{N}_+}$ of observations $t_i \in T$, if the corresponding sequence $\langle est_i(\langle t_1, \ldots, t_i \rangle)\rangle_{i \in \mathbb{N}}$ of DOP estimates converges to a probability distribution, then that distribution is an instance of the model $\mathcal{M}_{DOP}(\mathcal{R})$.[3]

Furthermore, the increased generality of $\mathcal{M}_{\text{DOP}}(\mathcal{R})$ now allows estimators to learn tree distributions that could not be learned before, simply because they were not allowed to be generated due to the finiteness condition on the productions of the underlying grammar.

Another problem we encountered in Sections 5 and 6 was the handling of unknown words. Firstly, with growing training data, previously unseen words are bound to occur and keep occurring. No finite *a-priory* set of terminals can account for this. Secondly, smoothing techniques used in state-of-the-art statistical language modeling implicitly make use of an infinite number of unknown-word productions. Defining both $\mathcal{W}$ and $\mathcal{R}$

[3] This is true because $\mathcal{M}_{\text{DOP}}(\mathcal{R})$ is then identical to the set of all probability distributions over T.

as countable sets prepares the grounds for a proper formal treatment of these phenomena.

In certain grammar formalisms, also a countable set of *nonterminals* could be desirable. If $\mathcal{W}$, $\mathcal{N}$ and $\mathcal{R}$ are allowed to be countable, the resulting set of utterances will also be countable, since the set of possible derivations from the grammar—finite sequences of rules, i.e., of tuples over $\mathcal{W} \cup \mathcal{N}$—is countable.

Note that the sets of nonterminals and terminals used in practical NLP applications are in fact countably infinite: Terminals are often represented as words over the Latin alphabet, and nonterminals as such words pre- or postfixed with a special character. Note also that at any given time, practical estimation procedures assign non-zero weights only to a finite number of productions and thus make explicit use only of a finite number of nonterminal and terminal symbols, although the number of the different symbols and productions that might occur during training is (at least theoretically) infinite.

In light of the discoveries made in the preceding sections it is intriguing to review the development of probabilistic parsing. Early work on the estimation of parse distributions from treebanks started out by introducing grammar formalisms in order to avoid working with an infinite number of parameters. State-of-the-art treebank grammars adopted this strategy but, from a theoretical perspective, end up again with an infinite number of parameters (although all actual models only deal with a finite number of parameters).

The main explanation for this seemingly circular development is that the kind of grammars that are suitable for capturing natural language syntax are far more complex than any existing formal grammars. The fact that language is a continuously and rapidly evolving phenomenon raises a question as to whether any a priori fixed, finite grammar will be able to approximate language syntax (including phenomena as unknown words). From this perspective, it seems that the treebank grammar approach, together with smoothing techniques, can be seen as a *efficient* solution for this problem.

References

1. Salomaa, A.: Probabilistic and Weighted Grammars. Inf.Control **15** (1969) 529–544
2. Black, E., Jelinek, F., Lafferty, J., Magerman, D., Mercer, R., Roukos, S.: Towards History-based Grammars: Using Richer Models for Probabilistic Parsing. In: Proceedings of the 31st Annual Meeting of the ACL (*ACL*'93), Columbus, Ohio (1993)
3. Collins, M.: Three generative, lexicalized models for statistical parsing. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL) and the 8th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Madrid, Spain (1997) 16–23
4. Charniak, E.: A maximum entropy inspired parser. In: Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-00), Seattle, Washington, USA (2000) 132–139
5. Bod, R., Scha, R., Sima'an, K., eds.: Data Oriented Parsing. CSLI Publications, Stanford University, Stanford, California, USA (2003) not refereed.
6. Good, I.: The population frequencies of species and the estimation of population parameters. Biometrika **40** (1953) 237–264

7. Ney, H., Martin, S., Wessel, F.: Statistical language modeling using leaving-one-out. In Young, S., Bloothooft, G., eds.: Corpus-based Methods in Language and Speech Processing. Kluwer Academic, Dordrecht (1997) 174–207
8. Katz, S.: Estimation of probabilities from sparse data for the language model component of a speech recognizer. IEEE Transactions on Acoustics, Speech and Signal Processing (ASSP) **35(3)** (1987) 400–401
9. Joshi, A.: Tree Adjoining Grammars: How much context sensitivity is required to provide a reasonable structural description. In D. Dowty, I.K., Zwicky, A., eds.: Natural Language Parsing, Cambridge, U.K., Cambridge University Press (1985) 206–250
10. Fujisaki, T., Jelinek, F., Cocke, J., Black, E., Nishino, T.: A probabilistic method for sentence disambiguation. In: Proceedings of the 1st International Workshop on Parsing Technologies. (1989)
11. Scha, R.: Taaltheorie en taaltechnologie; competence en performance. In: Computer-toepassingen in de Neerlandistiek, LVVN-jaarboek, English translation as: Language Theory and Language Technology; Competence and Performance http://iaaa.nl/rs/LeerdamE.html (1990)
12. Bod, R.: Enriching Linguistics with Statistics: Performance models of Natural Language. PhD dissertation. ILLC dissertation series 1995-14, University of Amsterdam (1995)
13. Charniak, E.: A maximum-entropy-inspired parser. In: Report CS-99-12, Providence, Rhode Island (1999)
14. Charniak, E.: Tree-bank Grammars. In: Proceedings *AAAI'96*, Portland, Oregon (1996)
15. Jelinek, F.: Statistical Methods for Speech Recognition. Cambirdge: MIT Press (1997)
16. Siegrist, K.: Virtual Laboratories in Probability and Statistics, www.math.uah.edu/stat (2004)
17. Zipf, G.: Human behavior and the Principle of Least Effort: an Introduction To Human Ecology. Addisson-Wesley (1949)
18. Marcus, M., Santorini, B., Marcinkiewicz, M.: Building a large annotated corpus of English: The Penn treebank. Computational Linguistics **19** (1993) 313–330