Rank Consistent Estimation: The DOP Case

Thuy Linh Nguyen tlnguyen@science.uva.nl

Amsterdam, November 1, 2004

Abstract

The goal of an estimator is to approximate the unknown distribution of the language from its partial evidence. In this thesis, a *rank consistent* estimator is defined as an estimator that preserves the ranking frequencies of all the full parse trees in the treebank proved to be rank consistent with respect to the training treebank. The *rank consistency* property adopts Laplace's Principle of Insufficient Reason for statistical parsing: a *rank consistent* estimator assigns the same probability to all trees that occur the same number of times in the training data.

This thesis presents the first non-trivial DOP estimator where the treebank is not only considered as a stochastic generating system but also a sample of the stochastic process. In this thesis, the existing DOP definitions of probability and derivation of full parse trees are generalized to subtrees. Fragments in the treebank's fragment corpus are assigned weights so that their probabilities are proportional to their relative frequencies. The estimator is proved to be *rank consistent*.

The theoretical property of the model is substantiated by empirical results. The new estimator outperforms the DOP1 estimator on the OVIS corpus.

Acknowledgements

First of all, I would like to acknowledge my advisor, Khalil Sima'an for being a very patient advisor and for his guidance and support throughout this thesis. I enjoyed the hours of thesis discussion as well as the course "Probabilistic Grammars and Data Oriented Parsing" he taught. I am very grateful for having had the opportunity to learn from him.

I would like to express my gratitude to Professor Dick de Jongth who helped me to overcome many difficulties during my graduate study in ILLC.

I am grateful to Prof. Remko Scha, Prof. Dick de Jongth and Yoav Seginer for serving on my qualifying exam committee, especially to Prof. Remko Scha who provided valuable feedback.

Many thanks to Andreas Zollmann for having always believed in my work, also for useful thesis suggestions and support during my stay in Amsterdam.

Last but not the least, I want to thank my mother who always loves and cares for me.

Contents

Chapter 1

Introduction

Since its inception, one of the primary goals of Artificial Intelligence has been the development of computational methods for language understanding [Brill & Mooney, 1997]. The Natural Language Processing (NLP) field covers not only lexical and grammatical information but also semantic, pragmatic and general world knowledge. In order to understand natural language utterances, one should not solve all of these problems at once but attack each one separately. The focus of this thesis is the syntactical analysis in NLP.

Syntactic analysis is a sub-field in NLP, it determines the grammatical structure of a sentence, that is how words are grouped into constituents such as noun phrase or verb phrase. Because of the syntactic and semantic preferences and constraints, people rarely notice the ambiguity and quickly settle on the interpretation based on the context. However, computers are unable to capture the context and the semantics of a sentence as humans do. This makes parsing natural language a difficult task due to the fact that the number of possible parses for a sentence increases with the length of the sentence. We want to select a few parses which are more likely to be correct from the parses which are syntactically possible.

One example of parsing with ambiguity we often see in the literature is to determine the parse of the sentence "He saw the dog with a telescope". There are at least two possible analyses of the sentence as in Figure (1.1) and the parser needs to determine tree (a) to be the correct parse tree of the sentence. Using *statistical parsing*, the parser can be used to choose from among many parses of the input sentence which ones are most likely based on their distributional behaviors. In statistical parsing, the probability distribution of the data in a language is determined by a *probabilistic grammar*. Informally speaking, a probabilistic grammar consists of a set of *rules* which are assigned *weights* to determine the probabilities of the generated sentence parses and *combination operations* with combination probabilities specify how sentence parses are generated.

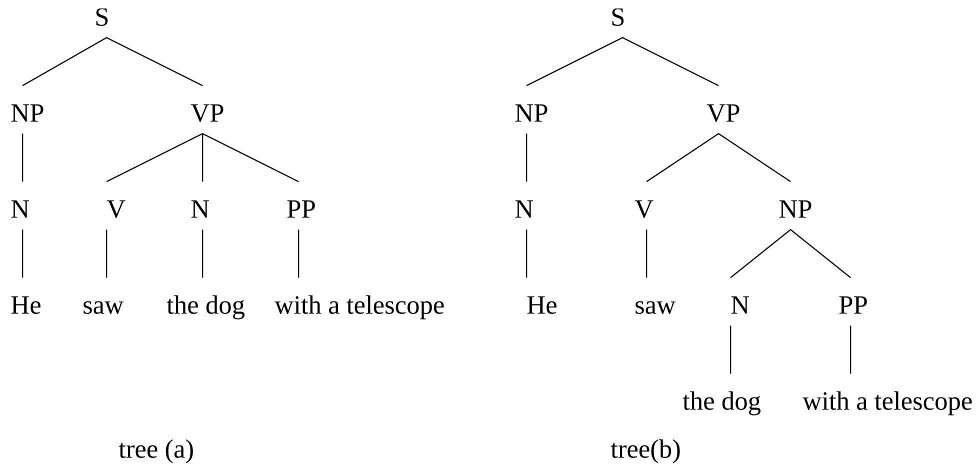


Figure 1.1: Examples of parsing with ambiguity

In the early stage of statistical parsing, grammar rules of the parser were built by linguists. However, this is a very difficult task due to the complexity of human languages. The grammar rules instead can be projected from the examples of the language. A collection of such examples of correct parses is referred to as *treebank* or *corpus*.

There are several approaches in statistical parsing. The simplest probabilistic model is Probabilistic Context Free Grammar (PCFG). PCFG was first studied in the late 1960s and early 1970s [Booth & Thompson, 1973]. It is simply a Context Free Grammar with probabilities added to the rules, indicating how likely different alternative rewriting rules are. There has been some work on probabilistic versions of other grammatical frameworks such as History Based Grammar [Black *et al*, 1993], Probabilistic Tree Adjoining Grammar [Resnik, 1992].

The probabilistic model described in this thesis is Data-Oriented Parsing (DOP). DOP was first introduced by [Scha, 1990] and formalized by [Bod, 1992]. It is a probabilistic model that works out statistics over subtrees (or *fragments*) in the treebank where the treebank is assumed to present the body of the parses that one has previously experienced. DOP fits into the statistical parsing framework: each treebank is a representation of a Stochastic Tree Substitution Grammar (STSG) that determines the probability distribution of the language. The STSG's rules are the treebank's fragments. These fragments are assigned weights between zero and one. Fragments are combined to obtain new parse trees that do not occur in the treebank. The probability of a parse tree depends on the weights of fragments that build it up.

1.1 Problem definition

The goal of statistical parsing is to approximate the language's probability distribution from a treebank as a finite collection of full parse trees. Laplace's "Principle of Insufficient Reason" attempts to supply a criterion of choice: two events are to be assigned the same probability if there are no reasons to think otherwise. In statistical parsing, the available information of a parse tree is its frequency in the treebank. A statistical parsing model should meet a necessary property of a good learner: whenever one tree appears in the corpus with higher relative frequency than another tree, the model should always assign the former tree higher probability than the latter tree. In chapter 3, we define this property as *rank consistent*.

In Data-Oriented Parsing, the estimator estimates the underlying STSG from the treebank. Thus the treebank should be considered as a stochastic generating system (to estimate fragments' weights) as well as a sample of the estimation's result (the treebank is also a representation of the language). However, in all the existing DOP estimators, the treebank is only used as a stochastic generating system not as a sample for estimating the underlying stochastic grammar. As a result, the outcome probability distribution of the estimations are not consistent with the distributions of trees in the treebank, i.e. they are not *rank consistent*. The failure to preserve the ranking of full parse trees in the treebank directly relates to other shortcomings of the existing models. [Johnson, 2002] proved that the original DOP model, DOP1, is inconsistent.

1.2 Contribution

This thesis proposes a *rank consistent* DOP estimator where the treebank is not only used as a stochastic generative system but also a sample of the estimated distribution. Perhaps the simplest statistical approach is the DOP maximum likelihood estimation which assigns probabilities to trees in the treebank as their relative frequencies and zero probability to all trees outside the treebank. In term of practicality, assigning all unseen events a zero probability would cause an *overfitting* problem which is clearly unsatisfactory.

To satisfy *rank consistency* and avoid overfitting, in our estimator the treebank is not only regarded as a collection of full parse trees but also a collection of fragments. To support this new view of the treebank, the DOP's existing probability definition and its derivation definition for full parse trees are generalized for subtrees. The frequency-distributions of fragments are the representations of their parse tree probabilities. The weights of fragments are calculated such that parse tree probabilities of fragments are *proportional* to their relative frequencies. The new estimator is proved to be *rank consistent*.

This thesis also studies the relationship of a *rank consistent* estimation and a consistent estimation as the training corpus size grows to the limit. The consistency is a necessary property in estimation theory. However, the fact that an estimation is consistent or not depends on the choice of the estimation error function.

1.3 Outline

In the following chapter, we introduce the basic concepts of statistical parsing and the Data Oriented Parsing framework. Also, we review the existing DOP models and their shortcomings which were also presented in previous works.

In chapter 3, we discuss the estimation theory's consistency property in statistical parsing. Also, we define the *rank consistency* property of an estimator and study the relationship between the consistency property and the *rank consistency* property.

The new estimator, DOPα, is presented in chapter 4. We present how to assign weights to fragments such that their probabilities are *proportional* to their relative frequencies.

The theoretical property of DOPα is substantiated by the empirical results in chapter 5.

Finally, chapter 6 gives the conclusion of the thesis and possible directions for future work.

Chapter 2

Background & Notations

In this chapter, the readers first will be made familiar with NLP terminologies and notations that we are going to use throughout the thesis. In section (2.2), the Data-Oriented Parsing framework is introduced with a focus on DOP1 [Bod, 1995]. Also, we review some of other DOP variants and their shortcomings, such as [Buratto, 2003] and [Zollmann, 2004].

2.1 Natural Language Processing Terminologies and Notations

2.1.1 Terminologies

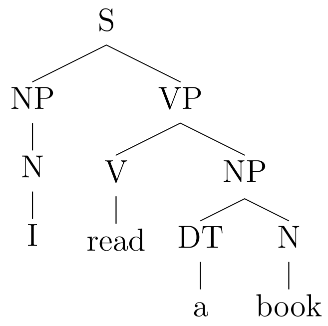
In formal grammars, words are also referred to as *terminal* symbols; *categories* or *constituent labels* such as "NP", "VP", . . . , are also referred to as *nonterminal* symbols. In the set of non-terminal symbols, there is a distinguished *sentence category* or *start symbol*. The start symbol is often denoted as "S".

In this thesis, when we mention a *tree* we mean a *phrase structure tree* where the root and non-leaf nodes are nonterminal symbols, the leaf nodes can be non-terminal or terminal symbols.

A *full parse tree* is a tree such that its root is the start symbol and all its leaves are terminal symbols.

An *utterance* or a *sentence* is a sequence of words. A *full parse tree of a sentence* is a full parse tree such that the sequence of its leaves forms the sentence.

For example, the full parse tree of the sentence "I read a book" is



where “I”, “read”, “a”, “book” are terminal symbols; “S”, “NP”, “VP”, “N”, “V”, “DT” are non-terminal symbols.

A *treebank* or a *corpus* is a sequence of full parse trees. The number of occurrences of a parse tree in the treebank is significant for the probability estimation that acts on the treebank.

An example of a treebank that consists of the two parse trees of the sentences “I read a book” and “He bought the magazine” is given in Figure (2.1).

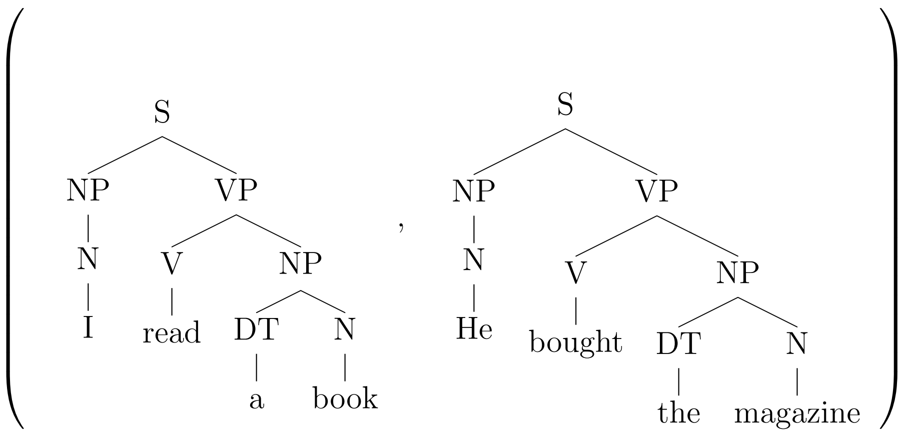


Figure 2.1: A treebank example

A *probability distribution*, $\mathrm{P}_{\mathbb{S}}$, over the set $\mathbb{S}$ is a function that assigns a probability to each element in $\mathbb{S}$ such that the probabilities of all elements in $\mathbb{S}$ sum up to one.

$$\mathrm{P}_{\mathbb{S}} : \mathbb{S} \to \mathbb{R} \qquad \sum_{a \in \mathbb{S}} \mathrm{P}_{\mathbb{S}}(a) = 1$$

When the set $\mathbb{S}$ is clear from the context, we write the probability of an element a in the set $\mathbb{S}$ as $\mathrm{P}(a)$.

A *language* is a probability distribution over a set of full parse trees. If the set of full parse trees is Ω, we denote the language as P^*_{Ω}.

An *estimator* estimates the language from a given treebank. The outcome of an *estimation* process is a probability distribution over a set of full parse trees. Statistical parsing assumes that the language was generated by a probabilistic grammar. Thus the *estimation* process actually determines the combination operators, the grammar's rules and the rules' weights.

2.1.2 Notations

Throughout this thesis, new notations and definitions are introduced whenever needed. However, we present here the most commonly used notations.

Let X be a multi-set and x be an element of X. We define

- $\boldsymbol{C}(X)$ as the total number of occurrences of all elements in X.
- $\boldsymbol{C}_X(x)$ as the number of occurrences of x in X.
- $\boldsymbol{rf}_X(x)$ as the relative frequency of x in X. This means that

$$\boldsymbol{rf}_X(x) = \frac{\boldsymbol{C}_X(x)}{\boldsymbol{C}(X)}$$

Examples: Let TC be a training corpus. $\boldsymbol{C}(\text{TC})$ is the number of full parse trees in TC. If t is a full parse tree, $\boldsymbol{C}_{\text{TC}}(t)$ is the number of times t appears in TC.

2.2 Data-Oriented Parsing

As mentioned in the previous chapter, DOP was first introduced in [Scha, 1990]. Later, DOP was formalized by [Bod, 1993]. A DOP model consists of:

- a representation of utterance analyses.
- a definition of fragments, the smallest units used for building new utterance analyses.
- composition operations that define how fragments are combined to new utterance analyses.
- a probabilistic model.

The DOP framework allows a wide range of instantiations. How the fragments are combined and how the probability of a new tree is computed from the fragments are left open. The first DOP instantiation, DOP1, was introduced by [Bod, 1995]. There are several variants of the DOP model improving upon DOP1.

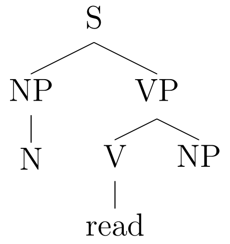
[Hoogweg, 2000] enriched DOP1 with an insertion operation to make the model more robust. [Bonnema *et al.*, 1999] used the assumption that every *derivation* of a tree is equally likely and derived the weights of the fragments based on this assumption. Back-off DOP [Sima'an & Buratto, 2003] considers a derivation of length two of a fragment as a back-off of the fragment. The weights of the fragments are discounted to the smaller fragments in the derivations of length two in a way that is similar to Katz's smoothing algorithm. [Zollmann, 2004] presented a consistent DOP estimator using the held-out estimation approach.

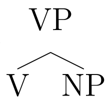
In the following sections, we present the DOP framework using its first instantiation DOP1 for illustration.

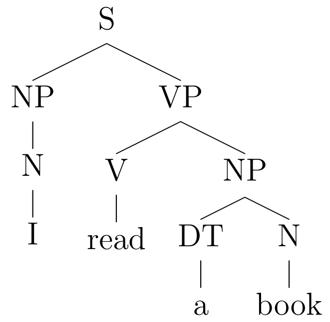
2.2.1 Fragments

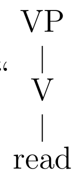
DOP captures all the syntactic analyses that appear in the treebank. Each syntactic analysis of a tree is represented by a *fragment* of the tree. A tree f is a *fragment* of tree T if

- f consists of more than one node.
- f is a connected sub-graph of T.
- except for the leaf nodes of f, each node in f has the same daughter nodes as the corresponding node in T.

For example, " 

" and " 

" are fragments of the parse tree " 

", while " 

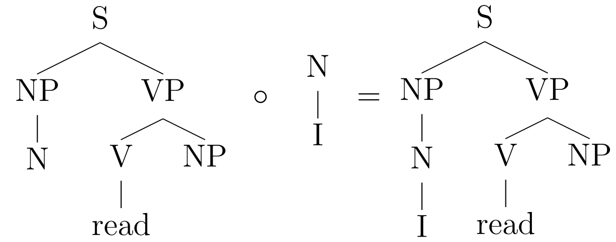
" is not a fragment of the tree.

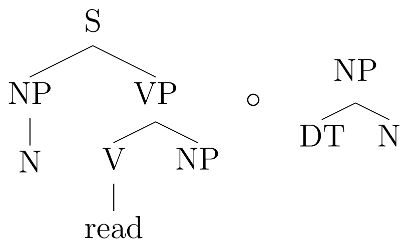
If we extract all fragments of all the trees in the treebank, we get the *fragment corpus* of the treebank. A fragment corpus is a multi-set of fragments. The fragment corpus of treebank TC is denoted as $\mathbf{Frag}_{\text{TC}}$. In DOP, often we are interested in the fragments with the same root, we denote $\mathbf{RFrag}_{\text{TC}}(f)$ as the collection of fragments in $\mathbf{Frag}_{\text{TC}}$ that have the same root as f.

2.2.2 Combination Operator

A combination operator specifies how fragments in the fragment corpus of a treebank are combined to create a new tree. DOP1 has only one combination operator, which is *leftmost substitution*.

The composition of trees t and u, denoted $t \circ u$, is defined iff the root of tree u is also the leftmost non-terminal leaf of tree t. If $t \circ u$ is defined, it yields a copy of tree t in which tree u is substituted on the leftmost non-terminal node of tree t.

Example: 

and  is undefined.

We note that the DOP1 composition operation is not associative, if t, u, v are subtrees then it is not guaranteed that $(t \circ u) \circ v = t \circ (u \circ v)$.[1]

Let $t_1, t_2, \ldots, t_k$ be subtrees, $t_1 \circ t_2 \circ \ldots \circ t_k$ is a shorthand of $(\ldots((t_1 \circ t_2) \circ \circ t_3) \circ \ldots) \circ t_k)) \ldots)$.

A sequence of subtrees $d = \langle t_1, t_2, \ldots, t_k \rangle$ is a *derivation* of a full parse tree T if $t_1 \circ t_2 \circ \ldots \circ t_k = T$

2.2.3 DOP Probability Estimation

The DOP model assigns *weights* to fragments, and the probability of a tree is calculated on the basic of weights of the tree's fragments. DOP1 uses the assumption that all fragments are independent; the weight of a fragment is its relative frequency in the treebank's fragment corpus.

Let TC be a treebank, $\mathbf{Frag}_{\text{TC}}$ be the treebank's fragment corpus, f be a fragment in the fragment corpus $\mathbf{Frag}_{\text{TC}}$. The weight of f according to the DOP1

[1] If $(t \circ u) \circ v$ and $t \circ (u \circ v)$ are both defined then $(t \circ u) \circ v = t \circ (u \circ v)$. A simple proof can verify this equation.

model is

$$\pi_{\text{DOP1}}(f) = \boldsymbol{rf}_{\mathbf{RFrag}_{\text{TC}}(f)}(f) = \frac{\boldsymbol{C}_{\mathbf{Frag}_{\text{TC}}}(f)}{\boldsymbol{C}(\mathbf{RFrag}_{\text{TC}}(f))}$$

where $\boldsymbol{C}(\mathbf{RFrag}_{\text{TC}}(f))$ is the number of occurrences of all fragments with the same root as f in $\mathbf{Frag}_{\text{TC}}$ and $\boldsymbol{C}_{\mathbf{Frag}_{\text{TC}}}(f)$ is the number of f's occurrences in $\mathbf{Frag}_{\text{TC}}$.

The fragments are considered to be independent, the probability of a derivation is the multiplication of the weights of fragments involved.

Let $d = \langle f_1, f_2, \ldots f_k\rangle$; the probability of derivation d is:

$$P_{\text{DOP1}}(d) = \pi_{\text{DOP1}}(f_1) \times \pi_{\text{DOP1}}(f_2) \times \ldots \times \pi_{\text{DOP1}}(f_k)$$

Let T be a full parse tree, each derivation of T is a combination of fragments to tree T. Thus the probability of tree T is the sum of the probabilities of its derivations.

$$P_{\text{DOP1}}(T) = \sum_{d \text{ is a derivation of } T} P_{\text{DOP1}}(d)$$

2.2.4 Stochastic Tree Substitution Grammar

The DOP model is based on the Stochastic Tree Substitution Grammar (STSG) formalism [Bod, 1995]. STSG is an extension of Probabilistic Context Free Grammar (PCFG) in which the set of rules is a set of elementary trees, which are combined with leftmost substitution composition.

A Stochastic Tree Substitution Grammar G consists of a quintuple $G = (V_N, V_T, S, R, \pi_G)$

- V_N is a finite set of non-terminal symbols.
- V_T is a finite set of terminal symbols; $V_N \cap V_T = \emptyset$.
- S is a distinguished non-terminal symbol, called start symbol: $S \in V_N$.
- R is a finite set of elementary trees whose interior nodes are labelled by non-terminal symbols in V_N and frontier nodes are labelled by terminal symbols or non-terminal symbols in $V_N \cup V_T$.
- π_G is a function that assigns *elementary probabilities* (or *weights*) to fragments in R. It yields positive real values not greater than 1: $\pi_G : R \to \mathbb{R}$ where $0 < \pi_G(f) \leq 1 \quad \forall f \in R$.

  If α is a non-terminal in V_N then $\sum_{f:root(f)=\alpha} \pi_G(f) = 1$.

An STSG *partial derivation* is a sequence of trees $\langle t_1, t_2, \ldots, t_k \rangle$ such that the result of their composition in that sequence $(t_1 \circ t_2 \circ \ldots \circ t_k)$ is a tree where non-terminal symbols can appear in the leaves.

An STSG *derivation* is a sequence of trees $\langle t_1, t_2, \ldots, t_k \rangle$ such that the result of their composition in that sequence $(t_1 \circ t_2 \circ \ldots \circ t_k)$ is a full parse tree. The probability of a derivation $d = \langle f_1, f_2, \ldots, f_k \rangle$ is the multiplication of the weights of the fragments involved.

$$P_G\left(d\right) = \pi_G(f_1) \times \pi_G(f_2) \times \ldots \times \pi_G(f_k)$$

The probability of a full parse tree T is the sum of the probabilities of its derivations.

$$P_G(T) = \sum_{d \text{ is a derivation of } T} P_G\left(d\right)$$

The DOP model uses the assumption that the language is generated by an STSG. The goal of DOP is to determine the STSG from a given treebank. The first DOP model, DOP1, assigns weights to fragments as their relative frequencies in the fragment corpus. Only the DOP estimator in [Hoogweg, 2000] proposed a new insertion operation to the STSG framework, other DOP estimators focus on the weight assignments to fragments in the fragment corpus.

2.2.5 The Shortcomings of DOP1

Bias towards big trees

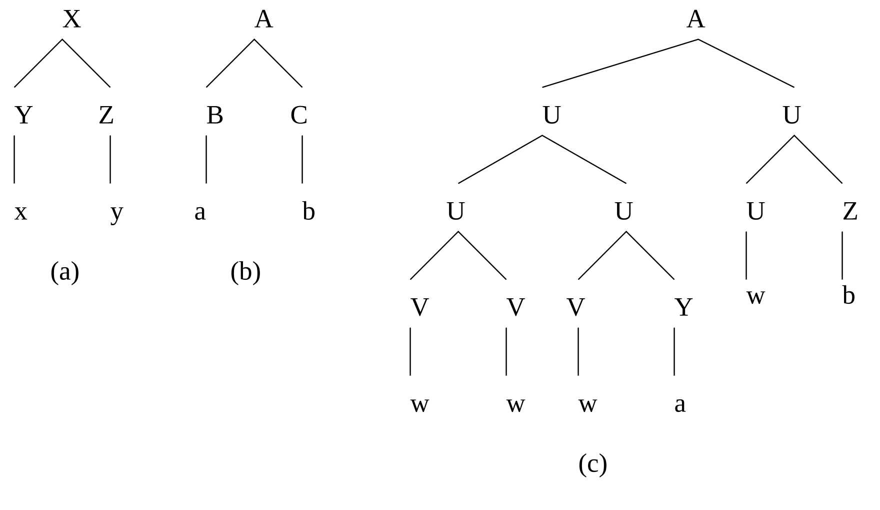


Figure 2.2: A treebank that shows DOP1's estimation bias towards big trees

[Bonnema, 2003] used the toy treebank in Figure (2.2) to illustrate DOP1's bias toward big trees problem.

If we extract all fragments of the treebank we will get 134 different fragments with root "A". So according to DOP1 estimation, all these fragments are assigned the weight 1/134. There are four different fragments with root "X" so their weights all are 1/4. It turns out that DOP1 assigns the tree $\begin{array}{cc} \multicolumn{2}{c}{X} \\ Y & Z \\ | & | \\ a & b \end{array}$ a higher probability than the tree $\begin{array}{cc} \multicolumn{2}{c}{A} \\ B & C \\ | & | \\ a & b \end{array}$. This contradicts the intuition that the latter tree which appears in the treebank and should be the preferred parse of the sentence "ab".

Let t be a tree root N and $t_1, t_2, \ldots, t_l$ are subtrees of t and t_i is the subtree headed by the i-th immediate daughter node of the root node of t. The number of t's subtrees headed by the root of t, denoted $\sigma(t)$, is

$$\sigma(t) = \prod_{i=1}^{i=l} (\sigma(t_i) + 1).$$

So the number of subtrees of a tree grows exponentially with the tree's depth. This explains the above counter-intuitive example of DOP1 estimation. In the example, tree (c) with depth four generates 130 fragments with root A, and tree (b) generates 4 fragments with root A, though tree (c) and tree (b) both appear in the corpus once. In DOP1, the big trees contribute high probability mass to fragments of the same root.

Bias & inconsistency

The data in a language can be assumed to follow a "true" probability distribution. An estimator is *biased* if the difference between the estimator's expected value [2] and the true distribution is different from zero. The estimator is *inconsistent* if the estimated distribution does not converge to the true distribution as the size of the training corpus increases.

The bias and inconsistency of DOP1 estimation were first pointed out by [Johnson, 2002]. [Zollmann, 2004] proved that an estimation is biased unless it

[2] If X is a random variable of the set Ω with probability mass function $p(x)$ then the expected value of X is $E(X) = \sum_{x \in \Omega} xp(x)$. [Krenn & Samuelsson, 1997]

assigns probability zero to all trees that do not appear in the treebank. DOP1 extracts all fragments from the treebank and generates trees outside the treebank. Hence, DOP1 is biased.

Denote the language as the probability distribution P^* over the set of full parse trees Ω. A treebank is a sequence of samples from the set Ω. Let $\textsc{tc}$ be a treebank with size n: $\textsc{tc} \in \Omega^n$..

An estimator is a function whose input is a treebank and returns a probability distribution of the language. Let *est* denote an estimator and est_n denote the estimator *est* whose domain is restricted to the treebanks with size n; $\mathcal{M}$ is the set of probability distributions over the set Ω, $est_n : \Omega^n \to \mathcal{M}$.

[Johnson, 2002] defines the difference or the *loss function* between two probability distributions, P and P^*, by the Euclidean distance metric $\mathcal{L}(\mathrm{P}, \mathrm{P}^*)$

$$\mathcal{L}(\mathrm{P}, \mathrm{P}^*) = \|\mathrm{P} - \mathrm{P}^*\|^2 := \sum_{t \in \Omega} \mathrm{P}^*(t)(est(\textsc{tc})(t) - \mathrm{P}^*(t))^2.$$

The *risk* or the *estimation error* of an estimator *est* is its expected loss $\mathrm{E}(\mathcal{L}(est(\textsc{tc}), \mathrm{P}^*))$. The estimator *est* is *inconsistent* if

$$\lim_{n \to \infty} \sum_{\textsc{tc} \in \Omega^n} \mathrm{P}^*(\textsc{tc}) \sum_{t \in \Omega} \mathrm{P}^*(t)(est(\textsc{tc})(t) - \mathrm{P}^*(t))^2 \neq 0 \tag{2.1}$$

where $\mathrm{P}^*(\textsc{tc})$ is the probability of treebank $\textsc{tc}$ size n in the language P^*. If $\textsc{tc} = (t_1, t_2, \ldots t_n)$ then $\mathrm{P}^*(\textsc{tc}) = \mathrm{P}^*(t_1) \times \mathrm{P}^*(t_2) \times \ldots \times \mathrm{P}^*(t_n)$.

To demonstrate the inconsistency of DOP1, [Johnson, 2002] used the example of a language P^*_Ω where Ω consists of two trees $t_1 = \begin{array}{cc} \multicolumn{2}{c}{\mathrm{S}} \\ \mathrm{A} & \mathrm{A} \\ | & | \\ \mathrm{a} & \mathrm{a} \end{array}$ and $t_2 = \begin{array}{c} \mathrm{S} \\ | \\ \mathrm{A} \\ | \\ \mathrm{a} \end{array}$ with probability distribution $P^*_\Omega(t_1) = P^*_\Omega(t_2) = 1/2$.

A treebank $\textsc{tc}$ of the language P^*_Ω would generate the following fragments

$$f_1 = \begin{array}{cc} \multicolumn{2}{c}{\mathrm{S}} \\ \mathrm{A} & \mathrm{A} \\ | & | \\ \mathrm{a} & \mathrm{a} \end{array}, \quad f_2 = \begin{array}{cc} \multicolumn{2}{c}{\mathrm{S}} \\ \mathrm{A} & \mathrm{A} \\ | & \\ \mathrm{a} & \end{array}, \quad f_3 = \begin{array}{cc} \multicolumn{2}{c}{\mathrm{S}} \\ \mathrm{A} & \mathrm{A} \\ & | \\ & \mathrm{a} \end{array}, \quad f_4 = \begin{array}{cc} \multicolumn{2}{c}{\mathrm{S}} \\ \mathrm{A} & \mathrm{A} \end{array},$$

$$f_5 = \begin{array}{c} \mathrm{S} \\ | \\ \mathrm{A} \\ | \\ \mathrm{a} \end{array}, \quad f_6 = \begin{array}{c} \mathrm{S} \\ | \\ \mathrm{A} \end{array}, \quad f_7 = \begin{array}{c} \mathrm{A} \\ | \\ \mathrm{a} \end{array}$$

Further calculation would show that given the treebank TC with size n including k trees t_1 and $(n-k)$ trees t_2, DOP1 would assign tree t_1 probability $2k/(n+k)$ and tree t_2 probability $(n-k)/(n+k)$. Apply the DOP1 estimation values to equation (2.1) to get the inconsistency of DOP1.

2.3 Some Variants of DOP Estimators

2.3.1 DOP Maximum Likelihood Estimator

The DOP Maximum Likelihood Estimator ($\mathrm{DOP}_{\mathrm{MLE}}$) attempts to generate the language such that probability of the input treebank is maximized.

Suppose we have the treebank TC, [Bonnema, 2003] shows that $\mathrm{DOP}_{\mathrm{MLE}}$ estimates probability of a tree in the treebank as its relative frequency in the treebank. That means

$$\mathrm{P}_{\mathrm{DOP}_{\mathrm{MLE}}}(t) = \boldsymbol{rf}_{\mathrm{TC}}(t) \qquad \forall\, t \in \mathrm{TC}$$

The probability assignment that maximizes the likelihood of TC assigns zero probability to trees outside the treebank and non-zero to trees in the treebank. $\mathrm{DOP}_{\mathrm{MLE}}$ probability assignment does not use the treebank as evidence to parse the sentences outside the treebank. $\mathrm{DOP}_{\mathrm{MLE}}$ does not allow for generalization, so it is not applicable in practice.

2.3.2 Bonnema DOP

In [Bonnema *et al.*, 1999], each full parse tree is considered as a set of all its derivations. Thus, the model views the corpus as the set of all of its derivations, each consisting of a sequence of subtree substitutions. From this different view of the corpus, the model assigns a weight to a fragment based on the fragment's participation in the set of treebank derivations. We omit here the detailed calculation of weight assignments but present the formula directly:

$$\pi_{\mathrm{BON}}(f) = \frac{\boldsymbol{C}_{\mathbf{Frag}_{\mathrm{TC}}}(f)}{\boldsymbol{C}_{\mathrm{TC}}(\mathrm{root}(f))} \times 2^{-\mathrm{N}(f)}$$

where $\mathrm{N}(f)$ is the function that returns the number of internal nodes of fragment f.

Using the $2^{-\mathrm{N}(f)}$ factor, the model reserves a high probability mass for the small fragments and a low probability mass for the big fragments, so it addresses the original DOP1 problem of bias toward big trees [Bonnema, 1999]. Also, when the trees in the language are generated by a PCFG, it was proved that Bonnema estimator is unbiased and consistent.

However, [Buratto, 2003] reported a disappointing result when the model was tested on the OVIS corpus[3]. We come back to the motivation of the model that considers all the derivations of a tree equally likely and independent. In that way, the model looses the dependencies of derivations of a tree. Also, the weight assignment to fragments does not consider the co-occurrence of fragments in a derivation that occurs in the treebank.

2.3.3 Back-off DOP

[Sima'an & Buratto, 2003] developed a different approach to parameter estimation for DOP than earlier works.

Each fragment in the fragment corpus is identified as a *complex fragment* or an *atomic fragment*. A fragment f is a complex fragment iff there exist two fragments f_1 and f_2 such that $f_1 \circ f_2 = f$; the derivation $\langle f_1, f_2 \rangle$ constitutes a *backoff* of the fragment f. A fragment is atomic iff it has no backoff. [Sima'an & Buratto, 2003] uses the backoff relation between a fragment and a pair of other fragments to form a directed acyclic graph (called backoff graph). A directed edge of the graph points from a node containing fragment f to another node containing one of its backoff pairs. One fragment might be in the derivations of length two of many other fragments, so one fragment can appear in different nodes of the graph.

The estimation procedure now operates iteratively, top-down over the backoff graph. In the first step, probabilities of fragments are assigned as their relative frequencies. In the subsequent steps, the model redistributes probability mass among DOP fragments by transferring stepwisely probability mass from complex fragments to their backoffs. The redistribution formula is similar to the Katz smoothing method [Sima'an & Buratto, 2003]. The Katz smoothing method helps a system handle the unseen events in the n-gram model. Back-off DOP, on the other hand, just adapts the Katz formula to transfer probability mass among fragments. [Zollmann, 2004] gave a review of the model and compared the backoff DOP approach and the Katz smoothing method approach.

We believe that the backoff DOP weight assignment reserves higher probability mass for short derivations than for long derivations. By applying backoff between fragments, the model tackles the assumption that derivations are equally likely and independent which is employed by previous models. Also, [Sima'an & Buratto, 2003] reported a very good empirical result when testing on the OVIS corpus.

However, the formula used by [Sima'an & Buratto, 2003] deviates from Katz's formula. This makes the model problematic (Sima'an, personal communication). When a fragment is a backoff of some other fragments, it will receive probability

[3]The reader can refer to chapter 5 for the description of OVIS corpus.

mass transferring several times. Intuitively, this situation does not affect the above properties of the model since the transferred probability mass is just a "smoothing" component. The existing work on backoff DOP does not yet provide its mathematical justification.

2.3.4 A Consistent DOP Estimator

[Zollmann, 2004] proposed the first non-trivial consistent DOP (called DOP*) estimator using held-out estimation. The training corpus is split into two parts: the extraction corpus (EC corpus) and the held-out corpus (HC corpus). The fragments of the EC corpus is extracted and assigned weights so that the likelihood of the held-out corpus is maximized. According to [Zollmann, 2004], this could be achieved by maximizing the joint probability of their *shortest derivations*. Fragments in EC are assigned weights based on their participation in the shortest derivations of the parse trees in the held-out corpus.

The model first determines p_{unkn}: an estimate for the probability that a tree will be unknown during the testing as the relative frequency of trees in held-out corpus that are not derivable using the fragments of the extraction corpus.

$$p_{\text{unkn}} = \frac{\boldsymbol{C}_{\text{HC}}\left(\{t \in \text{HC} \quad | \quad t \text{ is not EC-derivable}\}\right)}{\boldsymbol{C}(\text{HC})}$$

To each fragment f in the treebank's fragment corpus, the model assigns a weight $\pi_{\text{DOP}^*}(f)$:

$$\pi_{\text{DOP}^*}(f) = \beta(f) + \beta_{\text{smooth}}(f). \tag{2.2}$$

Readers are referred to [Zollmann, 2004] to see how the functions $\beta(f)$ and $\beta_{\text{smooth}}(f)$ are derived and their detailed formulas. Here, we are interested in the following properties of the weight assignment formula:

- $\beta(f) \neq 0$ if f participates in a shortest derivation of a tree in the held-out corpus.

  $\beta(f) = 0$ otherwise.

- $\beta_{\text{smooth}}(f) = 0$ if $p_{\text{unkn}} = 0$ and there are fragments with the same root as f participating in a shortest derivation of a tree in the held-out corpus.

  $\beta_{\text{smooth}}(f) \neq 0$ otherwise.

Thus if $p_{\text{unkn}} \neq 0$ then $\beta_{\text{smooth}}(f) \neq 0$. So from the weight formula (2.2), it is clear that when $p_{\text{unkn}} \neq 0$, every fragment f in the treebank's fragment corpus is assigned a non-zero weight. According to [Zollmann, 2004], DOP* only assign nonzero weights to a number of fragments that is linear in the number of depth-1 fragments contained in the held-out corpus. This property of DOP* is only achieved when $p_{\text{unkn}} = 0$. To explain it in another way, the efficiency of DOP* is obtained under the condition that every parse tree in the held-out corpus is derivable from fragments in the extraction corpus. In practice, this condition is often fails to be fulfilled because of the *sparse data* problem: even if the extraction corpus is quite big, the held-out corpus might contain the parse tree of a sentence with unknown words in the extraction corpus.

Moreover, the efficiency property of DOP*, when achieved, would bring out another shortcoming of the model: the coverage of DOP* is limited in comparison with DOP1.

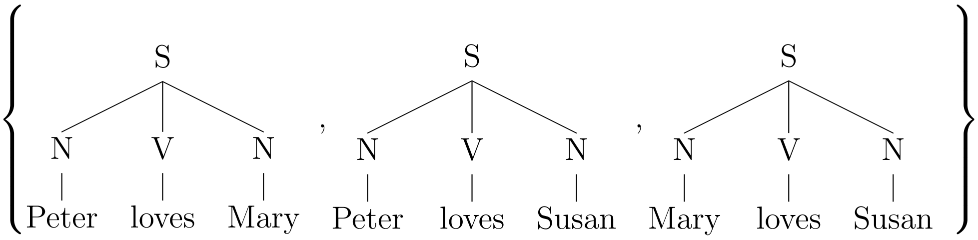


Figure 2.3: A DOP* treebank example

Suppose we are given a treebank that contains the parse trees of three sentences: "Peter loves Mary"; "Peter loves Susan"; "Mary loves Susan" as in Figure (2.3).

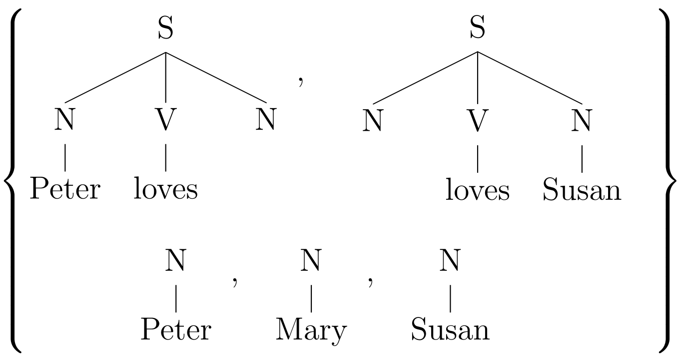


Figure 2.4: The DOP* fragments of the treebank example in Figure (2.3)

To maximize the coverage of DOP*, we successively use each tree in the treebank as the held-out corpus and get the other trees' fragments that involved in its shortest derivations. DOP* only assigns non-zero weights to fragments in Figure (2.4). Now we have the sentence "Susan loves Mary" and use the DOP* model to parse the sentence. Using the fragments in Figure (2.4), the test sentence can not be parsed. However, the sentence is easy to parse using the DOP1 model with the same treebank in Figure (2.3).

[Zollmann, 2004] proved that DOP* is consistent. This is a nice property. The DOP*'s results would improve with the treebank's size. However, held-out estimation is not a unique approach to get a consistent DOP estimator. According to Sima'an (personal communication), it is possible to devise Back-off DOP's weight assignments so that consistency is achieved.

We believe that using held-out estimation as [Zollmann, 2004], in the training process the held-out corpus probability is maximized. Also, DOP* can be always an efficient DOP estimator if the smoothing component in the weight assignment formula, $\beta_{\text{smooth}}(f)$, only assigns weights to depth-1 fragments (Zollmann, personal communication). The weakness of the model is the fact that the $\beta(f)$ component in the weight assignment formula only applies to fragments that participate in the shortest derivations of the held-out corpus. This would result in a low coverage of the model as explained in the treebank in Figure (2.3). Further calculation of DOP*'s weight assignment on the treebank in Figure (2.3) would show that the model fails to preserve the frequencies of full parse trees in the treebank.

2.4 Conclusion

We have introduced some NLP terminologies and the notations that we use in this thesis. Also, we introduced the Data-Oriented Parsing framework. The first DOP instantiation, DOP1, achieved state-of-the-art results but has some disturbing properties such as bias toward large trees and bias and inconsistency in estimation theory. Bonnema's DOP model addressed the DOP1's bias toward big trees problem. Backoff DOP gave a sensible probability estimation to DOP by structuring the set of fragments according to backoff order, tackling the independence assumption of previous work. [Zollmann, 2004] proposed the first consistent DOP estimator using held-out estimation.

DOP estimators use the assumption that the treebank is a representation of an STSG that generates the language. In all the mentioned DOP estimators, the treebank is regarded as a grammar generator. The models do not provide any properties of the estimations' results on trees in the treebank. For two trees in the treebank, it is possible that the models return a higher probability to the tree that appears with the lower frequency.

In the next chapter, we step into parsing models from the estimation theoretical point of view. We discuss the consistency property of an estimator and provide the definition of a *rank consistent*[4] estimator that preserves the ranking of frequencies in the treebank.

[4]The details of *rank consistent* are in chapter 3, section (3.1).

Chapter 3

Rank consistent estimation

In this chapter, first we discuss the concept of a consistent estimator in statistical parsing in section (3.1). A consistent estimator gives the intuition that the estimator's error tends to zero as the training corpus grows. There are several definitions of a consistent estimator, the differences between definitions are in how to evaluate the "estimator error". We show by example that the consistency property of an estimator, however, is dependent on the choice of an "estimator error" function.

In order to help us to decide what is the best estimator to use in any situation, we have to know about the properties of these estimators and choose those which have good properties. The consistency property emphases that the difference between estimator's result and the true distribution of data in the language should converge to zero as the training corpus size becomes large. However, in practice the treebank is finite and the available data is sparse. The consistency property does not predict the estimator's performance on finite treebanks. In probabilistic parsing, we would like to place a ranking on possible parses showing how likely each one is, or return the most likely parse of a sentence. So it is important for an estimator to estimate the true ranking probabilities. Section (3.2) will give a formal definition of a *rank consistent* estimator that preserves the ranking of parse trees' probabilities. We will show that the original data-oriented parsing model, DOP1, is not a *rank consistent* estimator.

Also, this chapter studies the relation of a rank consistent estimator and a consistent estimator as the treebank size grows.

3.1 Consistency definition

Let P^*_Ω be the probability distribution of the language over the set of full parse trees Ω. A training corpus TC with size n is a finite sequence of n random data from Ω. An estimator's input is a training corpus and its output is a probability distribution P for language Ω. In this thesis, we adopt the formal definition of an estimator that appears in [Prescher *et al.*, 2004].

Let est_n be an estimator of training corpora of size n and let $\mathcal{M}$ be the set of probability distributions over Ω. The estimator est_n maps a training corpus of size n to a probability distribution over language Ω as follows:

$$\text{est}_n : \Omega^n \to \mathcal{M} \qquad \text{TC} \in \Omega^n \Rightarrow \text{est}_n(\text{TC}) = P \in \mathcal{M} \ .$$

$range(\text{est}_n)$ is the set of probability distributions that estimator est_n returns. Each probability distribution P appears in $range(\text{est}_n)$ according to the probability of the training corpus TC in Ω^n that generated P.

Estimator est is a consistent estimator if as n grows to infinity, the probability distributions in $range(\text{est}_n)$ approach the true probability distribution P^* of language Ω.

An estimator error function $Err(\text{est}_n, P^*)$ maps the estimator est_n and probability distribution P^* to a non-negative value that measures the difference of set $range(\text{est}_n)$ to the true distribution P^*.

Thus estimator est is a consistent estimator if

$$\lim_{n\to\infty} Err(\text{est}_n, P^*) = 0$$

There are several ways to define a consistent estimator, and thus there are several ways to define our estimator error function Err.

One approach is to define a loss function $\mathcal{L} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$. If P and P' are two probability distributions then $\mathcal{L}(P', P)$ returns a non-negative real value that measures the loss of probability distribution P' with respect to P. The estimation error $Err(\text{est}_n, P^*)$ function takes into account the loss function value $\mathcal{L}(P, P^*)$ of all probability distributions P in $range(\text{est}_n)$.

For example: A possible definition of estimator error function is

$$Err(\text{est}_n, P^*) = \max_{\text{TC}\in\Omega^n} \mathcal{L}(est_n(\text{TC}), P^*) \ .$$

[Johnson, 2002] proposes the estimator error (or *risk function*)

$$Err(\text{est}_n, P^*) = \sum_{\text{TC}\in\Omega} \mathrm{P}_{\Omega^n}(\text{TC}) \times \mathcal{L}(est_n(\text{TC}), P^*)$$

where the loss function $\mathcal{L}(P, P^*)$ is the mean square error of P to P^* over Ω

$$\mathcal{L}(P, P^*) = \sum_{t\in\Omega} P^*(t)\left(P(t) - P^*(t)\right)^2 \ . \tag{3.1}$$

Another approach is to define estimator error as the probability mass of a set of training corpora for which the loss function value is greater than a real value ε.

$$Err(\mathrm{est}_n, P^*) = \sum_{\mathcal{L}(est_n(\textsc{tc}),P^*)>\varepsilon} P^*_{\Omega^n}(\textsc{tc})$$

In [Prescher *et al.*, 2004] $\mathcal{L}(P, P^*) := |P - P^*|$ thus the estimator error function is:

$$Err(\mathrm{est}_n, P^*) = \sum_{|est_n(\textsc{tc})-P^*|>\varepsilon} P^*_{\Omega^n}(\textsc{tc}) \; .$$

In the following example, an estimator est and two estimation error functions Err_1 and Err_2 are given. We will show that estimator est is consistent when estimator error function Err_1 is chosen and it is not consistent when Err_2 is chosen.

Example 3.1.1 We define $Err_1(est_n, P^*)$ as in [Johnson, 2002]

$$\mathcal{L}_1(P^*, \mathrm{est}_n(\textsc{tc}))(t) = (\mathrm{est}_n(\textsc{tc})(t) - P^*(t))^2$$

$$\mathcal{L}_1(P^*, \mathrm{est}_n(\textsc{tc})) = \sum_{t\in\Omega} P^*(t)(\mathrm{est}_n(\textsc{tc})(t) - P^*(t))^2$$

$$Err_1(\mathrm{est}_n, P^*) = \sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc}) \sum_{t\in\Omega} P^*(t)(est_n(\textsc{tc})(t) - P^*(t))^2$$

Estimator error function Err_2:

If we consider the loss value of probability distribution P' w.r.t probability distribution P as the total probabilities of trees for which probability distribution P' is different than probability distribution P, then the loss function $\mathcal{L}_2(P', P)$ is defined as follows:

$$\mathcal{L}_2(P^*, \mathrm{est}_n(\textsc{tc}))(t) = \begin{cases} 0 & \text{if} \quad P^*(t) = \mathrm{est}_n(\textsc{tc})(t) \\ 1 & \text{if} \quad P^*(t) \neq \mathrm{est}_n(\textsc{tc})(t) \end{cases}$$

$$\Longrightarrow \mathcal{L}_2(P^*, \mathrm{est}_n(\textsc{tc})) = \sum_{t\in\Omega} P^*(t)\mathcal{L}_2(P^*, \mathrm{est}_n(\textsc{tc}))(t)$$

$$Err_2(\mathrm{est}_n, P^*) = \sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc})\mathcal{L}_2(est_n(\textsc{tc}), P^*)$$

Let Ω be a set of two trees $\Omega = \{t_1, t_2\}$ and the language P^*_Ω with distribution: $P^*_\Omega(t_1) = P^*_\Omega(t_2) = \frac{1}{2}$.

Let $\textsc{tc}$ be a training corpus size n of Ω^n, estimator est_n is defined as follows:

$$\text{est}_n(\textsc{tc})(t_1) = \frac{1}{2} - \frac{1}{n} \qquad \text{est}_n(\textsc{tc})(t_2) = \frac{1}{2} + \frac{1}{n}\ .$$

i) Examine the consistency property of est when the estimation error function is Err_1

$$\mathcal{L}_1(est_n(\textsc{tc}), P^*)(t_1) = (est_n(\textsc{tc})(t_1) - P^*(t_1))^2 = (\frac{1}{2} - \frac{1}{n} - \frac{1}{2})^2 = \frac{1}{n^2}$$
$$\mathcal{L}_1(est_n(\textsc{tc}), P^*)(t_2) = (est_n(\textsc{tc})(t_2) - P^*(t_2))^2 = (\frac{1}{2} + \frac{1}{n} - \frac{1}{2})^2 = \frac{1}{n^2}$$

$$\mathcal{L}_1(est_n(\textsc{tc}), P^*) = P^*(t_1)\mathcal{L}_1(est_n(\textsc{tc}), P^*)(t_1) + P^*(t_2)\mathcal{L}_1(est_n(\textsc{tc}), P^*)(t_2)$$
$$\mathcal{L}_1(est_n(\textsc{tc}), P^*) = \frac{1}{2n^2} + \frac{1}{2n^2} = \frac{1}{n^2}$$

$$\begin{aligned}\Rightarrow Err_1(\text{est}_n, P^*) &= \sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc}) \underbrace{\sum_{t\in\Omega} P^*(t)\mathcal{L}_1(est_n(\textsc{tc}), P^*)(t)}_{=1/n^2} \\ &= \sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc})\frac{1}{n^2} \\ &= \frac{1}{n^2}\sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc}) = \frac{1}{n^2}\end{aligned}$$

$$\Rightarrow \lim_{n\to\infty} Err_1(\text{est}_n, P^*) = \lim_{n\to\infty}\frac{1}{n^2} = 0.$$

So est is consistent with estimation error function Err_1.

ii) Examine the consistency property of est when the estimation error function is Err_2.

Because $\forall n : \text{est}_n(\textsc{tc})(t_1) \neq P^*(t_1)$ and $\text{est}_n(\textsc{tc})(t_2) \neq P^*(t_2)$ so:

$$\forall n : \qquad \mathcal{L}_2(est_n(\textsc{tc}), P^*)(t_1) = \mathcal{L}_2(est_n(\textsc{tc}), P^*)(t_2) = 1$$

$$\begin{aligned}\Rightarrow Err_2(\text{est}_n, P^*) &= \sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc}) \sum_{t\in\Omega} P^*(t)\,\underbrace{\mathcal{L}_2(est_n(\textsc{tc}), P^*)(t)}_{=1} \\ &= \sum_{\textsc{tc}\in\Omega^n} P^*(\textsc{tc}) \sum_{t\in\Omega} P^*(t) = 1\end{aligned}$$

$$\Rightarrow \lim_{n\to\infty} Err_2(\text{est}_n, P^*) = \lim_{n\to\infty} 1 = 1 \neq 0.$$

So est is not consistent with estimation error function Err_2.

As the training corpus grows to the limit, a consistent estimator's error tends to zero. Though the consistency property of an estimator is intuitively appealing, the above discussion shows that whether an estimator is consistent or not depends on the definition of estimator error. In this thesis, we use the estimator error defined by [Johnson, 2002], unless it is explicitly stated otherwise.

The consistency property of an estimator expresses the estimator's results when the training corpus size grows to infinity. In practice, the training corpus is always a finite sequence of trees. Consistency does not give us information about the estimator when the training corpus size is finite. In the next section, we introduce the property of *rank consistency* which means that given a training corpus, the estimator's result is always consistent with the ranking frequency of trees in the training corpus.

3.2 Definition of Rank Consistent Estimator

Let s be a sentence; the parser evaluates the probability of a tree t to be the parse of s by finding $P(t|s)$. Tree t^* is the parse of s if

$$t^* = \arg\max_t P(t|s) = \arg\max_t \frac{P(t,s)}{P(s)} = \arg\max_t P(t,s)$$

Given tree t, sentence s is always determined, thus finding probability $P(t,s)$ is equivalent to finding probability of the tree, $P(t)$.

The estimator should assign probabilities to all trees in the language using a finite treebank as the language's partial evidence. The most famous criterion for choosing probabilities is Laplace's Principle of Insufficient Reason (or Indifference Principle) [Guiasu & Shenitzer, 1998]: *when one has no information to distinguish the probabilities of two events the best strategy is to consider them equally likely.* Applying Laplace's Principle, the estimator should find a probability assignment that agrees with the information in the treebank. One crucial constraint is that when two full parse trees appear in the treebank with the same frequency the estimator should always assign them the same probability. Also, when one tree appears in the treebank with a higher frequency than another tree, it should be assigned a higher probability than the other tree.

Let Ω be the language with probability distribution P^*; let $\text{est}(\textsc{tc})$ be the estimator induced by training corpus $\textsc{tc}$. The estimator $\text{est}(\textsc{tc})$ will give the correct most probable parse of a sentence or a correct priority of trees to be the parse of the sentence if for any tree t_1 and t_2 of the language Ω

$$P^*(t_1) \leq P^*(t_2) \qquad \text{implies} \qquad \text{est}(\textsc{tc})(t_1) \leq \text{est}(\textsc{tc})(t_2).$$

Let t_1 and t_2 be two trees in training corpus $\textsc{tc}$, we have $P^*(t_1) \leq P^*(t_2)$ implies $\boldsymbol{rf}_{\textsc{tc}}(t_1) \leq \boldsymbol{rf}_{\textsc{tc}}(t_2)$ or $\boldsymbol{C}_{\textsc{tc}}(t_1) \leq \boldsymbol{C}_{\textsc{tc}}(t_2)$

We define the concept of *rank consistency* that adopts the mentioned ranking preservation property.

Definition 3.2.1 An estimator est is *rank consistent* if for any training corpus TC, if tree t_1 and tree t_2 are in corpus TC with frequency $\boldsymbol{C}_{\text{TC}}(t_1)$ and $\boldsymbol{C}_{\text{TC}}(t_2)$ respectively, the estimator est induced by TC will assign tree t_1 a greater probability than t_2 if $\boldsymbol{C}_{\text{TC}}(t_1)$ is greater than $\boldsymbol{C}_{\text{TC}}(t_2)$ and vice versa. $\lhd$

Contemplation

The *rank consistency* is a necessary property of an estimation. It guarantees that the estimation preserves the trees' frequencies in the treebank. The rank consistent estimator should assign its probability in such a way that the unknown tree's probability is an adaption of the probabilities of other similar trees that appear in the treebank.

If we assume that the unknown tree is built up from fragments that appear in the treebank as in DOP framework, the estimator should not only preserve the ranking consistency of full parse trees but also preserve the ranking consistency of fragments that appear in the treebank.

3.2.1 The failure of DOP1

In DOP1, [Bod, 1995] defines the weight of a fragment t of category X as its frequency in the tree bank divided by the frequency of all subtrees with the same category X.

In this section, we show that DOP1 is not *rank consistent* where the selection criterion is most probable parse or most probable derivation.

DOP1 most probable parse is not *rank consistent*

[Johnson, 2002] proved that DOP1's most probable parse is not consistent and biased. We will show that it is not *rank consistent* either.

We take as an example the tree bank as given in Figure (3.1), where tree (a) appears two times in the corpus, tree (b) appears once and tree (c) appears nine times. Rank consistency requires that tree (a), which appears twice, should be assigned a higher parse tree probability than tree (b), which appears once in the corpus. This tree bank will generate the corpus fragments as shown in Figure (3.2). The numbers in brackets under fragments indicate their DOP1 weights.

Consider two trees of depth two $S,(A,(a,()))$ and $S,(A,(a,()),(b,()))$ of this corpus. We see the difference of the trees' frequencies and the trees' DOP1 parse

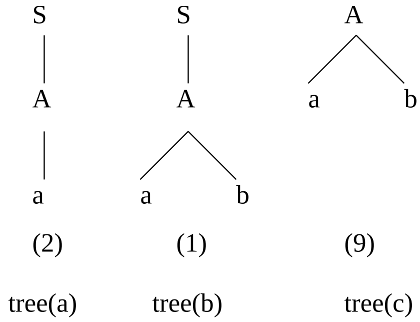


Figure 3.1: Example tree bank

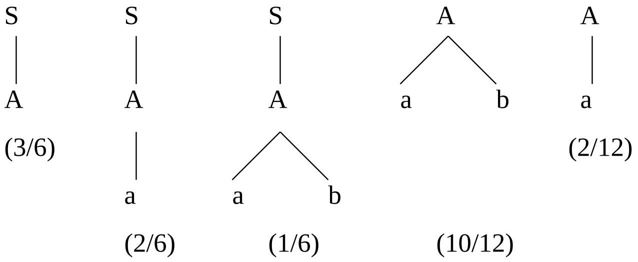


Figure 3.2: Fragments of tree bank in figure 3.1

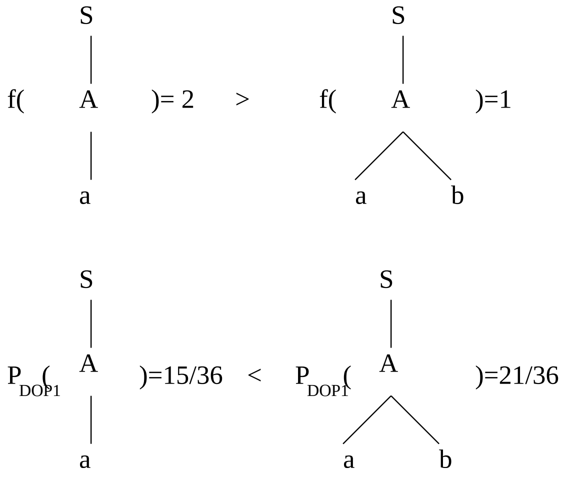


Figure 3.3: Frequencies and DOP1 parse tree probabilities of two trees

tree probabilities as depicted in Figure (3.3). DOP1 assigns tree (a) a lower parse tree probability than tree (b). Thus, when calculating the most probable parse of a tree, DOP1 does not give the result consistent with trees' frequencies. Hence, DOP1 is not *rank consistent*.

DOP1 most probable derivation is not *rank consistent*

A derivation probability is calculated as product of its fragments' weights. In DOP1 a fragment's weight is its relative frequency. If two trees appear in the corpus, these two trees are also in fragment corpus. It would be expected that DOP1's most probable derivation probabilities would be their weights in the fragment corpus. However, this is not always true. If one tree appears in the corpus with low frequency and its subtrees appear in the corpus with high frequencies, a derivation that involves more fragments may be the DOP1 's most probable derivation. Thus DOP1's most probable derivation is not consistent with the tree's low frequency in the corpus and it is not *rank consistent*.

We use the tree bank in Figure (3.1) again as an example. From the fragments' weights in Figure (3.2) we have the most probable derivation of two trees as depicted in Figure (3.4). DOP1 most probable derivation assigns tree $S,(A,(a,()),(b,())))$ higher probability than tree $S,(A,(a,()))$ though $S,(A,(a,()),(b,())))$ has lower frequency in the tree bank than $S,(A,(a,()))$. So DOP1 most probable derivation is not *rank consistent*.

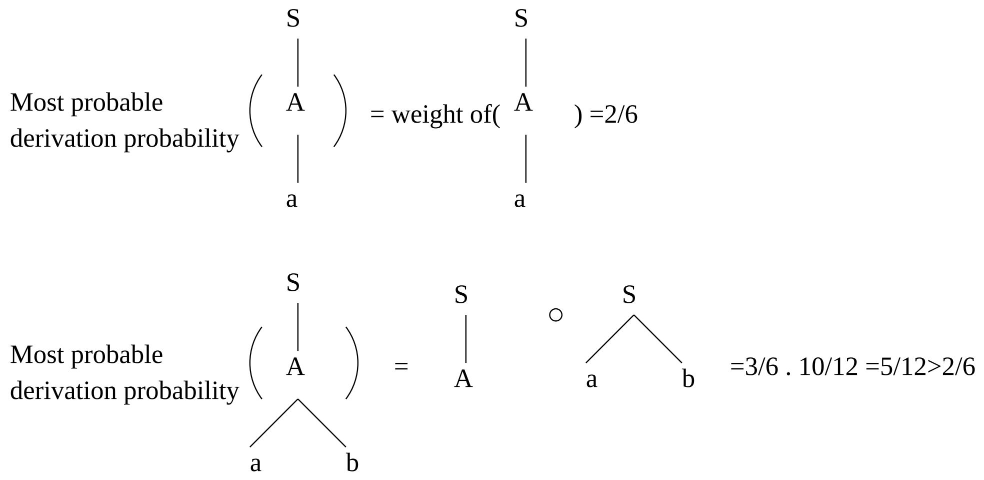


Figure 3.4: DOP1 most probable derivation of two trees in the corpus

3.2.2 A *rank consistent* estimator versus a consistent estimator

Consistency and *rank consistency* are two different properties of estimators.

A *rank consistent* estimator estimates the probabilities of trees in the corpus the same ranking as their frequencies. It uses the assumption that the ranking of trees' frequencies in the corpus is also the ranking of trees' probabilities in the language. This assumption is quite reasonable because the training corpus is the only data that the estimator observes from the language. The consistency definition considers the estimator's result from all the possible training corpora. In this section, we study if a consistent estimation preserves the ranking of tree probabilities as their relative frequencies in the corpus.

We give an example to show that when the training corpus grows to infinity, a consistent estimator is not a *rank consistent* estimator.

However, for any two trees with different probabilities in the language, the total probability of training corpora such that a consistent estimator yields an inconsistent ranking to these trees tends to zero.

In the limit a consistent estimator is not guaranteed to be *rank consistent*

We use again the estimator est in example (3.1.1).

Let $\Omega = \{t_1, t_2\}$ and $P^*_\Omega(t_1) = P^*_\Omega(t_2) = \frac{1}{2}$

TC is a training corpus sized n of Ω, estimator est_n is defined as follows:

$$\mathrm{est}_n(\textsc{tc})(t_1) = \frac{1}{2} - \frac{1}{n} \qquad \mathrm{est}_n(\textsc{tc})(t_2) = \frac{1}{2} + \frac{1}{n}.$$

Example (3.1.1) shows that estimator est is consistent when the mean-square error is used as estimator error function. However, estimator est always selects the tree t_2 as a more probable tree than tree t_1. Now we prove that est is not *rank consistent*.

Let $\boldsymbol{C}_{\textsc{tc}}(t_1)$ and $\boldsymbol{C}_{\textsc{tc}}(t_2)$ be the frequencies of t_1 and t_2 in training corpus TC. The probability of a training corpus TC size n is:

$$\begin{aligned} P^*(\textsc{tc}) &= (P^*(t_1))^{\boldsymbol{C}_{\textsc{tc}}(t_1)}(P^*(t_2))^{\boldsymbol{C}_{\textsc{tc}}(t_2)} \\ &= (\frac{1}{2})^{\boldsymbol{C}_{\textsc{tc}}(t_1)}(\frac{1}{2})^{\boldsymbol{C}_{\textsc{tc}}(t_2)} = (\frac{1}{2})^{\boldsymbol{C}_{\textsc{tc}}(t_1)+\boldsymbol{C}_{\textsc{tc}}(t_2)} = (\frac{1}{2})^n. \end{aligned}$$

Estimator est estimates probability of t_1 less than t_2. So if in training corpus TC frequency of tree t_1 is equal or greater than frequency of tree t_2, the estimator est is not consistent with their relative frequencies.

Because each training corpus TC of size n has the same probability $1/2^n$, the total probabilities of training corpora in which the frequency of tree t_1 is equal or greater than the frequency of tree t_2 is at least $1/2$ or total probabilities of training corpora size n that the estimator est estimates trees' probabilities not consistent with trees' frequencies is at least $1/2$ for all n.

Estimator est and language Ω above are examples of a language in which there are two trees with the same true distribution while estimator est, though consistent, always give the probability of one tree different than the other tree. This situation is depicted in the graph of Figure(3.5)

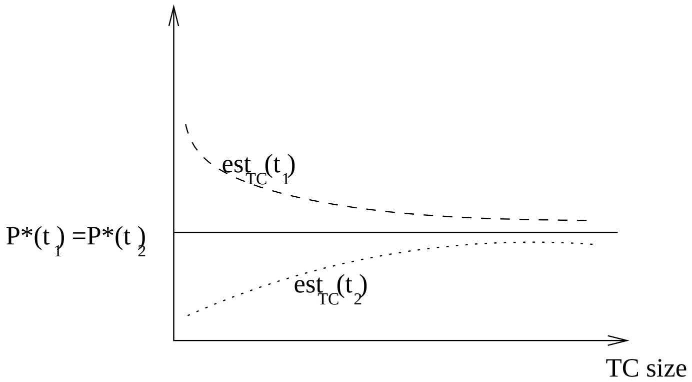


Figure 3.5: A consistent estimator est and most probable derivation of two trees in the corpus

For any tree in the language, the difference between a consistent estimator's result and tree's actual distribution approaches zero as the training corpus size grows to infinity. However, this property of a consistent estimator does not imply that in the limit the estimator estimates probabilities trees in the corpus consistent with their actual ranking or their relative frequency in the corpus.

In the previous section, we showed that a consistent estimator does not preserve the ranking of relative frequencies of all trees in the corpus. However, for any two trees t_1 and t_2 with different probabilities in the language, we will prove that the consistent estimator estimates their probabilities consistent with their relative frequencies when the corpus size grows to infinity.

The intuition above is stated formally by the following theorem (3.2.2).

Given corpus TC, a consistent estimator est estimates tree t_1 and t_2 probabilities $\mathrm{est}_n(\mathrm{TC})(t_1)$ and $\mathrm{est}_n(\mathrm{TC})(t_2)$ consistent with ranking of their relative frequencies $\boldsymbol{rf}_{\mathrm{TC}}(t_1)$ and $\boldsymbol{rf}_{\mathrm{TC}}(t_2)$ if

$$(\mathrm{est}_n(\mathrm{TC})(t_1) - \mathrm{est}_n(\mathrm{TC})(t_2))(\boldsymbol{rf}_{\mathrm{TC}}(t_1) - \boldsymbol{rf}_{\mathrm{TC}}(t2)) > 0\ .$$

Theorem 3.2.2 *Let t_1 and t_2 be two trees in language Ω with different probabilities. If est is a consistent estimator of Ω, then for all $q > 0$ there exists an integer N that*

$$\sum_{(est_n(\text{TC})(t_1)-est_n(\text{TC})(t_2))(\boldsymbol{rf}_{\text{TC}}(t_1)-\boldsymbol{rf}_{\text{TC}}(t2))<0} P^*(\text{TC}) < q \qquad \forall n > N$$

To prove theorem (3.2.2) we need the following lemma:

Lemma 3.2.3 *Let Ω be language with true distribution P^*, est be a consistent estimator. For all trees t_0 of Ω and for any real value $\varepsilon > 0, q > 0$ there exists N such that*

$$\sum_{|est_n(\text{TC})(t_0)-P^*(t_0)|>\varepsilon} P^*(\text{TC}) < q \qquad \forall n > N$$

Proof: Because est is a consistent estimator so $\forall q' > 0$, there exists N such that

$$\sum_{\text{TC}\in\Omega^n} P^*(\text{TC}) \sum_{t\in\Omega} P^*(t)(P^*(t) - est_n(\text{TC})(t))^2 < q' \qquad \forall n > N \tag{3.2}$$

Choose $q' = q.P^*(t_0).\varepsilon^2$ and replace to (3.2):

$$\sum_{\text{TC}\in\Omega^n} P^*(\text{TC}) \sum_{t\in\Omega} P^*(t)(P^*(t) - est_n(\text{TC})(t))^2 < q.P^*(t_0).\varepsilon^2 \tag{3.3}$$

Applying

$$P^*(t_0)(P^*(t_0) - est_n(\text{TC})(t_0))^2 < \sum_{t\in\Omega} P^*(t)(P^*(t) - est_n(\text{TC})(t))^2$$

to (3.3). We have

$$\sum_{\text{TC}\in\Omega^n} P^*(\text{TC})P^*(t_0)(P^*(t_0) - est_n(\text{TC})(t_0))^2 < q.P^*(t_0).\varepsilon^2 \tag{3.4}$$

Applying

$$\sum_{\text{TC}\in\Omega^n,|est_n(\text{TC})(t_0)-P^*(t_0)|>\varepsilon} P^*(\text{TC}) < \sum_{\text{TC}\in\Omega^n} P^*(\text{TC})$$

to (3.4) we have:

$$\sum_{\text{TC}\in\Omega^n,|est_n(\text{TC})(t_0)-P^*(t_0)|>\varepsilon} P^*(\text{TC})P^*(t_0)\underbrace{(P^*(t_0) - est_n(\text{TC})(t_0))^2}_{>\varepsilon^2} < q.P^*(t_0).\varepsilon^2.$$

$$ \tag{3.5} $$

$$\Rightarrow \sum_{\text{TC}\in\Omega^n, |est_n(\text{TC})(t_0)-P^*(t_0)|>\varepsilon} P^*(\text{TC})P^*(t_0).\varepsilon^2 < q.P^*(t_0).\varepsilon^2. \tag{3.6}$$

$$\Rightarrow \sum_{\text{TC}\in\Omega^n, |est_n(\text{TC})(t_0)-P^*(t_0)|>\varepsilon} P^*(\text{TC}) < q \tag{3.7}$$

QED

Note [Zollmann, 2004] proposes a definition of strong consistent estimation. According to him, an estimation is strongly consistent if the `supremum` difference of the estimation result and the true distribution in the whole language converges to zero as the training corpus grows large. Then in (theorem (3.1.1), [Zollmann, 2004]), he proved that a strong consistent estimation is also a Johnson consistent estimator. The remaining question goes in the opposite direction of his theorem: whether a consistent estimation is also a strong consistent estimation? In our lemma (3.2.3), we solve a somewhat weaker form of the question. Given any tree t_0, in the limit a consistent estimation will converge on its true probability. The lemma serves as the main step in the proof of theorem 3.2.2.

Proof of 3.2.2. We use lemma (3.2.3) to prove theorem (3.2.2) as follows:

Choose $\varepsilon = \frac{1}{2}|P^*(t_1) - P^*(t_2)|$, because $P^*(t_1) \neq P^*(t_2)$ so $\varepsilon > 0$.

Suppose $P^*(t_1) < P^*(t_2)$ and

$$|est_n(\text{TC})(t_1) - P^*(t_1)| < \varepsilon \quad \text{and} \quad |est_n(\text{TC})(t_2) - P^*(t_2)| < \varepsilon$$

$$\Rightarrow est_n(\text{TC})(t_1) < est_n(\text{TC})(t_2) \tag{3.8}$$

Similarly, if

$$|\boldsymbol{rf}_{\text{TC}}(t_1) - P^*(t_1)| < \varepsilon \quad \text{and} \quad |\boldsymbol{rf}_{\text{TC}}(t_2) - P^*(t_2)| < \varepsilon$$

$$\Rightarrow \boldsymbol{rf}_{\text{TC}}(t_1) < \boldsymbol{rf}_{\text{TC}}(t_1) \tag{3.9}$$

From (3.8) and (3.9) we have

$$(\text{est}_n(\text{TC})(t_1) - \text{est}_n(\text{TC})(t_2))(\boldsymbol{rf}_{\text{TC}}(t_1) - \boldsymbol{rf}_{\text{TC}}(t2)) > 0$$

So if $(\mathrm{est}_n(\textsc{tc})(t_1) - \mathrm{est}_n(\textsc{tc})(t_2))(\boldsymbol{rf}_{\textsc{tc}}(t_1) - \boldsymbol{rf}_{\textsc{tc}}(t2)) < 0$, then

$$\begin{aligned}
&|est_n(\textsc{tc})(t_1) - P^*(t_1)| > \varepsilon \quad \text{or} \quad |\boldsymbol{rf}_{\textsc{tc}}(t_1) - P^*(t_1)| > \varepsilon \\
\text{or} \quad &|est_n(\textsc{tc})(t_2) - P^*(t_2)| > \varepsilon \quad \text{or} \quad |\boldsymbol{rf}_{\textsc{tc}}(t_2) - P^*(t_2)| > \varepsilon \\
\text{where} \quad &\varepsilon = \frac{1}{2}|P^*(t_1) - P^*(t_2)|
\end{aligned} \tag{3.10}$$

Choose $q' = \frac{q}{4}$

est is a consistent estimator, apply lemma (3.2.3): there exists $N_{\mathrm{est}_1} > 0$, $N_{\mathrm{est}_2} > 0$ that

$$\sum_{\textsc{tc}\in\Omega^n, |est_n(\textsc{tc})(t_1) - P^*(t_1)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N_{\mathrm{est}_1}$$

and

$$\sum_{\textsc{tc}\in\Omega^n, |est_n(\textsc{tc})(t_2) - P^*(t_2)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N_{\mathrm{est}_2}$$

Relative frequency estimator ($\mathrm{DOP}_{\mathrm{MLE}}$) is also a consistent estimator, apply lemma (3.2.3): there exists $N_{\mathrm{MLE}_1} > 0$ and $N_{\mathrm{MLE}_2} > 0$ that

$$\sum_{\textsc{tc}\in\Omega^n, |rf(\textsc{tc})(t_1) - P^*(t_1)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N_{\mathrm{MLE}_1}$$

$$\sum_{\textsc{tc}\in\Omega^n, |rf(\textsc{tc})(t_2) - P^*(t_2)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N_{\mathrm{MLE}_2}$$

Choose $N = \max\{N_{\mathrm{est}_1}, N_{\mathrm{est}_2}, N_{\mathrm{MLE}_1}, N_{\mathrm{MLE}_2}\}$, we have

$$\sum_{\textsc{tc}\in\Omega^n, |est_n(\textsc{tc})(t_1) - P^*(t_1)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N \tag{3.11}$$

$$\sum_{\textsc{tc}\in\Omega^n, |\boldsymbol{rf}_{\textsc{tc}}(t_1) - P^*(t_1)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N \tag{3.12}$$

$$\sum_{\textsc{tc}\in\Omega^n, |est_n(\textsc{tc})(t_2) - P^*(t_2)| > \varepsilon} P^*(\textsc{tc}) < q' \qquad \forall n > N \tag{3.13}$$

$$\sum_{\text{TC}\in\Omega^n, |\boldsymbol{rf}_{\text{TC}}(t_2)-P^*(t_2)|>\varepsilon} P^*(\text{TC}) < q' \qquad \forall n > N \tag{3.14}$$

Sum over (3.11), (3.12), (3.13) and (3.14):

$$\begin{aligned}
&\sum_{\text{TC}\in\Omega^n, |est_n(\text{TC})(t_1)-P^*(t_1)|>\varepsilon} P^*(\text{TC}) + \sum_{\text{TC}\in\Omega^n, |\boldsymbol{rf}_{\text{TC}}(t_1)-P^*(t_1)|>\varepsilon} P^*(\text{TC}) + \\
&\sum_{\text{TC}\in\Omega^n, |est_n(\text{TC})(t_2)-P^*(t_2)|>\varepsilon} P^*(\text{TC}) + \sum_{\text{TC}\in\Omega^n, |\boldsymbol{rf}_{\text{TC}}(t_2)-P^*(t_2)|>\varepsilon} P^*(\text{TC}) \quad < \\
&\qquad q' + q' + q' + q' \quad = q \quad \forall n > N
\end{aligned}$$

$$\Rightarrow \sum_{\substack{\text{TC}\in\Omega^n, |est_n(\text{TC})(t_1)-P^*(t_1)| >\varepsilon \\ \text{or} \ |\boldsymbol{rf}_{\text{TC}}(t_1)-P^*(t_1)| >\varepsilon \\ \text{or} \ |est_n(\text{TC})(t_2)-P^*(t_2)| >\varepsilon \\ \text{or} \ |\boldsymbol{rf}_{\text{TC}}(t_2)-P^*(t_2)| >\varepsilon}} P^*(\text{TC}) < q \qquad \forall n > N \tag{3.15}$$

Apply result in (3.10) to (3.15) we have

$$\sum_{(\text{est}_n(\text{TC})(t_1)-\text{est}_n(\text{TC})(t_2))(\boldsymbol{rf}_{\text{TC}}(t_1)-\boldsymbol{rf}_{\text{TC}}(t2))<0} P^*(\text{TC}) < q \qquad \forall n > N$$

QED

3.3 Conclusion

We have discussed the consistency property of probability estimation in Natural Language Processing. A consistent estimator gives the intuition of an estimator assigning true probabilities for trees as their true probability in the language when the training corpus grows to infinity. We show by example that the consistency property of an estimator depends on the definition of error function.

The training corpus is the only evidence of the language for an estimator. If one tree appears with higher frequency than the other tree in a training corpus, intuitively the former tree has higher probability in the language. We defined this estimation's property as *rank consistent* and proved that the commonly used DOP1 is not *rank consistent*.

We showed that a consistent estimation does not estimate trees' probabilities consistent with the ranking of trees' frequencies in the whole tree bank but for any two trees with different actual probabilities in the language, in the limit a consistent estimator estimates their probabilities consistent with their frequencies in the tree bank.

In the following chapter, we present a nontrivial *rank consistent* DOP estimation.

Chapter 4

The new DOP estimation - DOPα

4.1 Overview

For estimation, a treebank serves as the only evidence of the language, thus the distribution in the treebank is assumed to reflect the true distribution. The purpose of estimation is to approximate this true distribution of the language from the treebank. In the previous chapters, we discuss the rank consistency property that the ranking of parse tree probabilities of trees in the treebank consistent with their frequencies. In this chapter, we will present a *rank consistent* DOP estimator.

The DOP estimations assume that trees of the language are generated by an STSG that is encoded in the treebank. Thus in a *rank consistent* DOP estimation, the treebank should be regarded not only as a stochastic generating system but also as a sample of the stochastic process.

A naive solution for the DOP model, DOP_{MLE}, assigns probabilities to trees in the corpus as their relative frequencies and zero probability for the trees outside the corpus. However, the treebank is finite and just contains a portion of the language. The DOP_{MLE} is extremely *overfitting* and against the DOP spirit that a new sentence is parsed by combining the syntactic analyses that already appear in the treebank.

To keep the DOP spirit and avoid the overfitting, the treebank should be viewed not only as a sequence of trees but also as all the fragments that can be extracted as DOP prescribes. This is represented by the fragment corpus of the treebank. Thus the estimator should generate not only the full parse trees in the treebank but also generate all fragments in the fragment corpus. One constraint we think crucial is that the estimator should preserve the ranking of relative frequencies of fragments in the fragment corpus. In this way, the estimator also preserves the ranking of frequencies of trees in the treebank, because trees in the treebank are also fragments of the fragment corpus. The estimator that preserves the ranking of the fragments is also *rank consistent*.

Now we discuss the roles of extracted fragments and their weights in the DOP estimation process. If we retrieve only depth-one fragments from the treebank, the resulting PCFG model also generates all the trees and subtrees that DOP generates. However, the PCFG model does not account for the co-occurrence of rules. In DOP, all the fragments of the treebank are extracted. This makes DOP able to capture all the long distance dependencies of the constituents, related lexical items and structures that appear in the treebank.

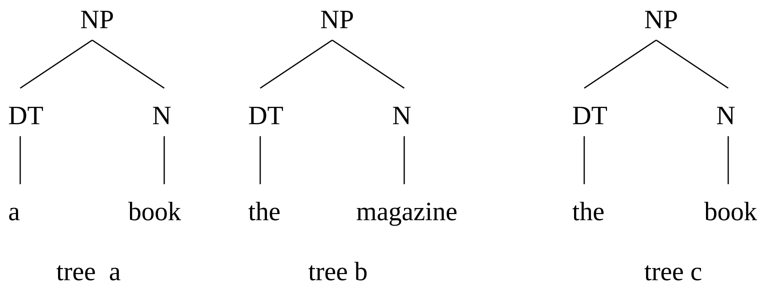


Figure 4.1: Tree (a) and (b) are fragments of the corpus that contains two sentences "I read a book" and "He bought the magazine" while tree (c) is not a fragment of the corpus. All three trees can be derived from smaller fragments of the corpus.

Let us consider the toy treebank in Figure (2.1) that contains the parse trees of two sentences "I read a book" and "He bought the magazine". Tree (a) and tree (b) in Figure (4.1) are fragments of the treebank's fragment corpus with relative frequencies 0.1 and 0.1 respectively while tree (c) is not a fragment of the treebank. However, all trees (a), (b) and (c) can be generated from their smaller subtrees in the fragment corpus. In DOP, fragments (a), (b) are included in the STSG rules to account for the co-occurrence of their sub-structures in the corpus.

Thus the estimation should also assign weights to fragments in such a way that the joint dependency of the fragment's subtrees are recovered. The dependency of a fragment's sub-structures is captured by the fragment's relative frequency in the fragment corpus that DOP extracts from the treebank.

Suppose the STSG $G = (V_N, V_T, S, R, \pi)$ generates the language. The STSG grammar treats fragments as if they were disjoint. The dependencies of sub-structures in a tree are recovered by summing up over all the derivations of the tree. Let T be a full parse tree of grammar G. A derivation of T represents a possible combination of fragments in G. Also, a derivation of T is one possible way of generating T from grammar G. The parse tree probability of T, $\mathrm{P}_G(T)$, is the sum of all the derivations' probabilities of T.

The STSG model G not only generates full parse trees, it also generates *partial trees* whose leaves are non-terminals or trees whose roots are not the start symbol of the grammar. However, the existing STSG definitions of derivation and parse tree probability are only applicable to full parse trees. As we will explain in section (4.2), the STSG partial derivation concept fails to capture all the possible combinations of fragments to a tree. To support the new view of the corpus as its collection of fragments, the STSG derivation definition is generalized so that it is applicable to an arbitrary tree. We call it *derivation**. A *derivation** of tree t, according to grammar G is a possible combination of fragments in G to tree t. The formal definition of *derivation** is given in section (4.2). We define the parse probability of t, $\mathrm{P}_G(t)$, is the sum of all *derivations** probabilities. It specifies how likely t is generated from grammar G and also captures the dependencies of sub-structures in t. For a full parse tree, the new notions of *derivation** and probability coincide with the original definition of derivation and full parse tree probability.

From the above discussion, we see the co-occurrences of subtrees from the treebank's point of view and from the STSG's point of view. From the treebank's point of view, the co-occurrences of a tree's sub-structures is captured by the tree's relative frequency in the treebank's fragment corpus. From the STSG point of view, the co-occurrences of a tree's sub-structures are captured by the parse probability of the tree. So how should the estimator assign weights to fragments such that their probabilities reflect their relative frequencies in the fragment corpus and also the model is *rank consistent*?

One answer would be an imaginary estimator γ that assigns weights to fragments such that $\mathrm{P}_\gamma(f) = \boldsymbol{rf}_{\mathbf{RFrag}}(f)$ for all fragments f in the fragment corpus $\mathbf{Frag}_{\mathrm{TC}}$.[1] However, this condition is too strict: There are corpora of a language such that this condition can not be fulfilled. For example, if we have the treebank in Figure (4.2) and its fragment corpus is in Figure (4.3), we cannot assign weights to its fragments that fulfill the condition $\mathrm{P}_\gamma(f) = \boldsymbol{rf}_{\mathbf{RFrag}}(f)$.

To fulfill the condition $\mathrm{P}_\gamma(f) = \boldsymbol{rf}_{\mathbf{RFrag}}(f)$, the weight of $\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A} \\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}$ is calculated

[1] The model γ is different from DOP1. DOP1 assigns weights to fragments as their relative frequencies. Thus in DOP1, the relative frequency of a fragment just takes part as the probability of one derivation of the fragment whereas the probability of the fragment is the sum of all the derivations probabilities. The model γ assigns weights to fragments such that sum of all the derivations probabilities of a fragment is equal to its relative frequency

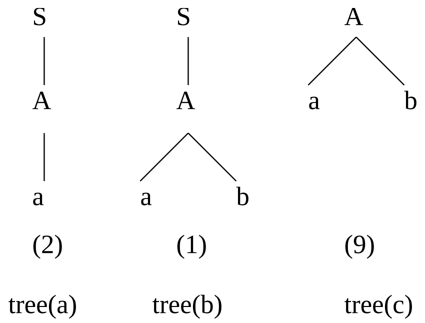


Figure 4.2: A treebank example

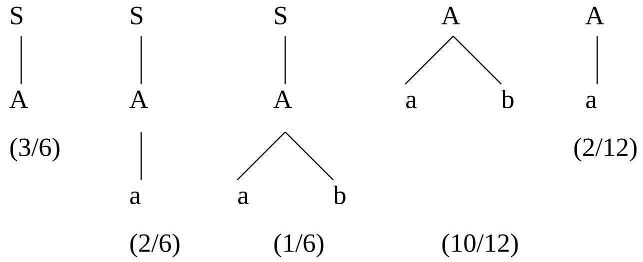


Figure 4.3: Fragments of treebank in figure 4.2, the numbers in brackets are the fragments' relative frequencies.

from its parse tree probability as follows:

$$
\mathrm{P}_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) = \pi_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\end{array}\right) \times \pi_{\gamma}\left(\begin{array}{c}\mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) + \pi_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) \tag{4.1}
$$

Because

$$
\pi_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\end{array}\right) = \mathrm{P}_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\end{array}\right) = \boldsymbol{rf}_{\mathbf{RFrag}}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\end{array}\right)
$$

$$
\pi_{\gamma}\left(\begin{array}{c}\mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) = \mathrm{P}_{\gamma}\left(\begin{array}{c}\mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) = \boldsymbol{rf}_{\mathbf{RFrag}}\left(\begin{array}{c}\mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right).
$$

Apply the above equations to (4.1), we have

$$
\Rightarrow \boldsymbol{rf}_{\mathbf{RFrag}}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) = \boldsymbol{rf}_{\mathbf{RFrag}}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\end{array}\right) \times \boldsymbol{rf}_{\mathbf{RFrag}}\left(\begin{array}{c}\mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) + \pi_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right)
$$

$$
\Rightarrow \frac{2}{6} = \frac{3}{6} \times \frac{10}{12} + \pi_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right)
$$

$$
\Rightarrow \pi_{\gamma}\left(\begin{array}{c}\mathrm{S}\\ | \\ \mathrm{A}\\ \widehat{\mathrm{a}\quad\mathrm{b}}\end{array}\right) = \frac{2}{6} - \frac{5}{12} < 0.
$$

So it is impossible to assign weights to fragments to fulfill the constraint $\mathrm{P}_{\gamma}(f) = \boldsymbol{rf}_{\mathbf{RFrag}}(f)$ for all fragments f.

We cannot build a model for which $\mathrm{P}_{\gamma}(f) = \boldsymbol{rf}_{\mathbf{RFrag}}(f)$ for all fragments f. However, we want to build a model such that probabilities of fragments capture their relative frequencies and the probabilities of fragments preserve the ranking of their relative frequencies. The most appropriate answer is to assign weights to fragments so that the parse probabilities of the fragments are *proportional* to their relative frequencies in the fragment corpus. We call this model DOPα.

That means we will determine a real value α such that $P_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f)$ for all fragments f in $\mathbf{Frag}_{\text{TC}}$. We call α the *proportionality factor*.

In the following subsection, we show that assigning weights to fragments such that their probabilities are proportional to their relative frequencies makes DOPα *rank consistent*. The definition of *derivation** is in section (4.2). The estimation procedure of DOPα is in section (4.3).

4.1.1 DOPα is *rank consistent*

In DOPα, we treat the corpus not only as a multi-set of parse trees but also as a multi-set of subtrees.

Let T be a full parse tree in the treebank, T is also a fragment in the treebank's fragment corpus.

$$P_{\text{DOP}\alpha}(T) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(T) = \alpha \times \frac{\boldsymbol{C}_{\mathbf{Frag}_{\text{TC}}}(T)}{\boldsymbol{C}(\mathbf{RFrag}_{\text{TC}}(T))} = \alpha \times \frac{\boldsymbol{C}_{\text{TC}}(T)}{\boldsymbol{C}(\mathbf{RFrag}_{\text{TC}}(T))} \quad (4.2)$$

Let T_1 and T_2 be two trees in the treebank TC and $\boldsymbol{C}_{\text{TC}}(T_1) \leq \boldsymbol{C}_{\text{TC}}(T_2)$. These two trees have the root as the start symbol of the model. Trees T_1 and T_2 are also fragments in the fragment corpus of the same root. From equation (4.2) we have $P_{\text{DOP}\alpha}(T_1) \leq P_{\text{DOP}\alpha}(T_2)$. So DOP$\alpha$ is *rank consistent*.

4.2 Definition of *derivation**

When calculating the probability of one full parse tree, we calculate probabilities of all the derivations of the parse tree, that means all the possible combinations of the fragments to the tree. So if T is a subtree of the tree, we have to consider all the possible combinations of the fragments to the tree T. The set of all the possible combinations of the fragments to an arbitrary tree T is not captured by the existing STSG *partial derivation* or *derivation* definition.

In DOP1, a *partial derivation* $\langle t_1, t_2, \ldots t_k \rangle$ is a finite sequence of leftmost nonterminal substitutions. The result of a partial derivation is a tree that might have nonterminals appearing on the leaves. The definition of a tree's *partial derivation* restricts the number of possible combinations of subtrees to the tree and is not suitable to our goal.

For example take tree (T) in Figure (4.4). Tree T has a nonterminal node NP in its leaves. Let $\langle t_1, t_2, \ldots, t_k \rangle$ be one partial derivation of T, then t_1 has the node NP as its leftmost non-terminal node. If the length of the derivation is

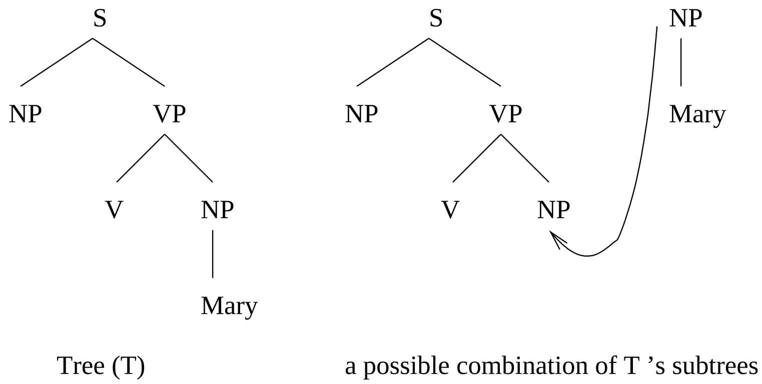
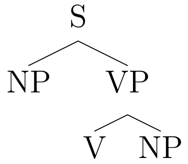


Figure 4.4: tree(T) and a possible combination of its subtrees

greater than or equal to two, tree t_2 will be substituted on this NP node, but the node NP is a leaf node of T so it is impossible for the length of the derivation to be greater than or equal two. Tree (T) has only one leftmost partial derivation i.e. tree (T) itself. However, tree (T) is the result of the combination of two trees  and  which is depicted in the right side of Figure (4.4).

We define *derivation** of an arbitrary tree T so that the set of *derivation** of tree T covers all possible combinations of fragments to tree T. When T is a parse tree whose leaves are terminal symbols, the notion *derivation** of T is equivalent to the STSG derivation of T and vice versa. As for STSG's, probability of tree T is the sum of the probabilities of T's *derivations**.

Definition

Definition 4.2.1 A tree t is called a *root subtree* of tree T if t is a fragment of T and the root of t corresponds to the root of T $\lhd$

Example An example of a tree's root subtree is in Figure (4.5)

Definition 4.2.2 If t is a root subtree of T then a non-terminal leaf node of t is called a *T-substitutable* node if it corresponds to a non-leaf node in T. $\lhd$

Example In Figure (4.6), tree t is a root subtree of T. In tree t, NP (the non-terminal with italic font in dotted circle) is the only *T-substitutable* node

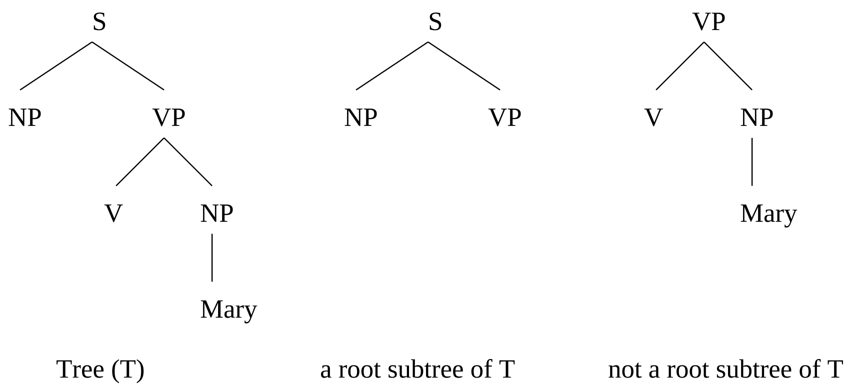


Figure 4.5: An example of a root subtree

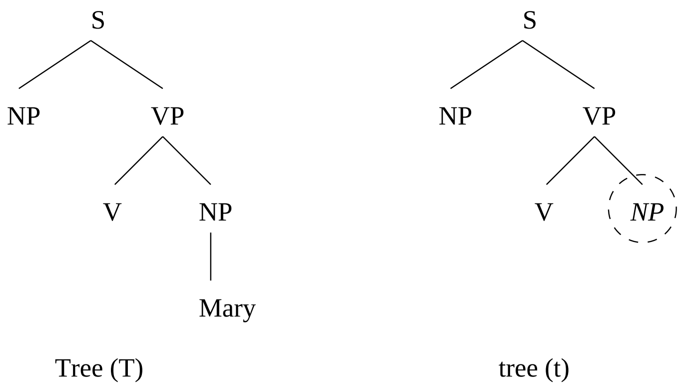


Figure 4.6: An example of a *T-substitutable* node. The "NP" node in dotted circle is tree t's only *T-substitutable* node.

Note When T is a full parse , and t is a root subtree of T, the *T-substitutable* nodes of t are also its leftmost non-terminal leaves.

Definition 4.2.3 If t_1 is a root subtree of tree T and the leftmost *T-substitutable* node of t_1 is the same as the root of some tree t_2, then the *T-composition* of t_1 and t_2, $t_1 \circ_T t_2$, is a copy of tree t_1 in which t_2 has been substituted into the leftmost *T-substitutable* node. $\lhd$

Example An example of *T-composition* is in Figure (4.7)

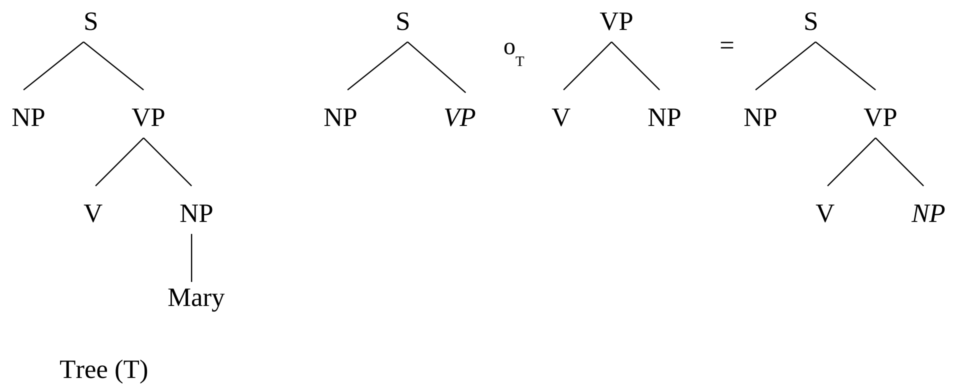


Figure 4.7: An example of *T-composition.*

Note If T is a tree whose leaves are terminal symbols and t is a root subtree of T then the leftmost *T-substitutable* nodes of T are also the leftmost non-terminal leaf of t. So the *T-composition* of t_1 and t_2, $t_1 \circ_T t_2$, is the same as the DOP composition of t_1 and t_2, $t_1 \circ t_2$

Definition 4.2.4 A tuple of trees $< t_1, t_2, \ldots t_k >$ is a *derivation** of tree T iff the *T-composition* of $t_1, t_2, \ldots t_k$ yields tree T, i.e., $t_1 \circ_T t_2 \circ_T \ldots \circ_T t_k = T$ $\lhd$

Example An example of *derivation** is in Figure (4.8)

Note When T is a tree whose leaves are terminal symbols, the *T-composition* operator gives the same result as the composition operator for STSG's so each *derivation** of tree T is also a STSG's derivation of T and the set of *derivations** and STSG's derivations are the same.

Definition 4.2.5 If $\langle t_1, t_2, \ldots t_k \rangle$ is a *derivation** of tree T, the probability of this *derivation** is $\pi_{\text{DOP}\alpha}(t_1) \times \pi_{\text{DOP}\alpha}(t_2) \times \ldots \times \pi_{\text{DOP}\alpha}(t_k)$ and probability of tree T is the sum of the probabilities of its *derivations**. $\lhd$

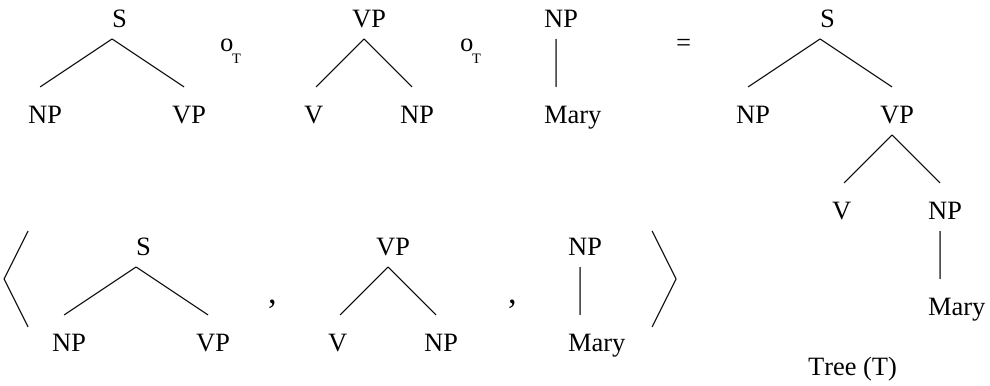


Figure 4.8: An example of a *derivation** of T

4.3 DOPα Estimation Procedure

We repeat again the condition of DOPα. For all fragments f:

$$\alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) = P_{\text{DOP}\alpha}(f) \tag{4.3}$$

$$P_{\text{DOP}\alpha}(f) =^{\text{def}} \sum_{\text{d is a } derivation^* \text{of } f} P_{\text{DOP}\alpha}(d) \tag{4.4}$$

Combining equation (4.3) and equation (4.4)we have

$$\alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) = \sum_{\substack{\text{d is a } derivation^* \text{of } f \\ \text{length of } d \text{ is } 1}} P_{\text{DOP}\alpha}(d) + \sum_{\substack{\text{d is a } derivation^* \text{of } f \\ \text{length of } d \text{ is greater than } 1}} P_{\text{DOP}\alpha}(d)$$

$$= \sum_{d=<f>} P_{\text{DOP}\alpha}(d) + \sum_{d=<f_1,f_2,\ldots,f_n>,\ n>1} P_{\text{DOP}\alpha}(d)$$

$$= \pi_{\text{DOP}\alpha}(f) + \sum_{d=<f_1,f_2,\ldots,f_n>,\ n>1} P_{\text{DOP}\alpha}(d)$$

$$\Rightarrow \pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - \sum_{d=<f_1,f_2,\ldots,f_n>,\ n>1} P_{\text{DOP}\alpha}(d)$$

Denote $P^-_{\text{DOP}\alpha}(f) = \sum_{d=<f_1,f_2,\ldots,f_n>,\ n>1} P_{\text{DOP}\alpha}(d)$.

We derive the formula for DOPα weight in Figure (4.9)[2].

[2]DOPα weight assignment formula to a fragment is similar to Back-off estimation approach in the way that it is the result of the relative frequency of the fragment after discounting the probabilities of the fragments' derivations probabilities. However, DOPα considers all

$$
\pi_{\text{DOP}\alpha}(f) = \begin{cases} \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) & \text{if depth of } f \text{ is one} \\ \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - P^{-}_{\text{DOP}\alpha}(f) & \text{if depth of } f \text{ is greater than one} \end{cases}
$$

$$
\text{where } P^{-}_{\text{DOP}\alpha}(f) = \sum_{d=<f_1,f_2,\ldots,f_n>,\ n>1} P_{\text{DOP}\alpha}(d)
$$

Figure 4.9: DOPα weight assignment formula

Equation in figure (4.9) gives a hint of how to obtain the DOPα weight for fragment f. If we have determined the proportionality factor α then $\alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f)$ is determined. We have to calculate $P^{-}_{\text{DOP}\alpha}(f) = \sum_{d=<f_1,f_2,\ldots,f_n>,\ n>1} P_{\text{DOP}\alpha}(d)$.

A recursive estimator that is tractable

Now we develop a recursive way for estimating $\pi_{\text{DOP}\alpha}(f)$.

Let $d =< f_1, f_2, \ldots, f_n >$ be a *derivation** of f where the length n of d is greater than one. So each fragment f_i $(i = 1, \ldots n)$ is a subtree of f with a number of internal nodes less than f.

Define function #intnodes : $\mathbf{Frag}_{\text{TC}} \rightarrow \mathbb{N}$; #intnodes$(f)$ returns the number of internal nodes of f. If f is a subtree depth 1 then #intnodes$(f) = 1$.

Denote maxintnodes$(\textsc{tc}) = \max_{f \in \mathbf{Frag}_{\text{TC}}}$ #intnodes(f) as the maximum number of internal nodes of a fragment in the fragment corpus $\mathbf{Frag}_{\text{TC}}$.

We use the fragment corpus in Figure (4.3) to illustrate function #intnodes and maxintnodes value

$$
\#\text{intnodes}\left(\begin{array}{c} \text{S} \\ | \\ \text{A} \\ \widehat{\text{a}\quad\text{b}} \end{array}\right) = 2, \qquad \#\text{intnodes}\left(\begin{array}{c} \text{A} \\ \widehat{\text{a}\quad\text{b}} \end{array}\right) = 1 \qquad \text{maxintnodes}(\textsc{tc}) = 2
$$

Now we partition the fragment corpus $\mathbf{Frag}_{\text{TC}}$ according to fragments with the same number of internal nodes.

the derivations of the fragments whereas Back-off DOP, in [Sima'an & Buratto, 2003], only considers the derivations of length two of a fragment. Also, DOPα recursively assigns weights to small fragments first then to big fragments whereas Back-off DOP in contrast, assigns weights to big fragments first then to small fragments. In Back-off DOP, in some situations, a later modification of weight assignment in small fragments could result in an unexpected modification of the parse tree probabilities of other big fragments.

Denote $\mathbf{Frag}^n_{\text{TC}}$ as the set of fragments that have exactly n internal nodes in fragment corpus $\mathbf{Frag}_{\text{TC}}$.

$$\mathbf{Frag}_{\text{TC}} = \bigcup_{n=1}^{\text{maxintnodes(TC)}} \mathbf{Frag}^n_{\text{TC}} \qquad \text{and} \qquad \forall n \neq m \qquad \mathbf{Frag}^n_{\text{TC}} \cap \mathbf{Frag}^m_{\text{TC}} = \emptyset$$

The set $\{\mathbf{Frag}^n_{\text{TC}}| \quad \text{where } n = 1, \ldots, \text{maxintnodes(TC)}\}$ forms a partition of the fragment corpus $\mathbf{Frag}_{\text{TC}}$. The DOP$\alpha$ weights of fragments in $\mathbf{Frag}_{\text{TC}}$ are calculated incrementally base on the number of internal nodes n.

If f is a fragment with one internal node, $f \in \mathbf{Frag}^1_{\text{TC}}$, from the equation in Figure(4.9), the DOPα weight of f is $\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f)$.

Suppose we have calculated DOPα weights of fragments that have at least $(n-1)$ internal nodes. Fragments in $\mathbf{Frag}^n_{\text{TC}}$ are calculated as follows.

We repeat here the DOPα weight of fragment f in the formula of Figure (4.9):

$$\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - P^-_{\text{DOP}\alpha}(f)$$

where $P^-_{\text{DOP}\alpha}(f) = \sum_{\text{length of } d \text{ is greater than } 2} P_{\text{DOP}\alpha}(d)$

To get the DOPα weight of fragment f, we have to calculate $P^-_{\text{DOP}\alpha}(f)$. Each derivation d whose length is greater than two only involves fragments that have a number of internal nodes less than f. So derivation d only involves fragments that have already been assigned DOPα weights in previous steps.

We create a temporary DOP STSG grammar $G_n = (V_N, V_T, S, R, \Pi)$ that includes DOPα fragments and weights that we have calculated in the previous steps. That means the set of non-terminal symbols V_N, the set of terminal symbols V_T and the start symbol S of G_n are as for DOP1, the set of rules R only consists of fragments of at most $n-1$ internal nodes, $R = \bigcup_{i=1}^{n-1} \mathbf{Frag}^i_{\text{TC}}$. The weights of fragments in R of G_n are the already determined DOPα weights, $\pi_{G_n}(f) = \pi_{\text{DOP}\alpha}(f) \quad \forall f \in R$.

Each derivation d of DOPα that involves only fragments with less than n internal nodes is also a derivation of G_n with the same probability and vice versa. So for each fragment f of DOPα that have exactly n internal nodes $\mathrm{P}_{G_n}(f) = P^-_{\text{DOP}\alpha}(f)$.

Thus $\pi_{\text{DOP}\alpha}(f)$ can now be calculated for all fragments $f \in \mathbf{Frag}^n_{\text{TC}}$ by applying as follows: $\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - \mathrm{P}_{G_n}(f)$.

The estimator procedure for DOPα is summarized in Figure (4.10)

4.4 Determining α

In the previous section, when DOPα estimates fragment weights, we assume that the proportionality factor α is given. This section will specify how this

1. Choose α
2. Calculate maxintnodes(TC)
3. Partition $\mathbf{Frag}_{\text{TC}}$: $\mathbf{Frag}_{\text{TC}} = \mathbf{Frag}^{1}_{\text{TC}} \cup \mathbf{Frag}^{2}_{\text{TC}} \cup \ldots \cup \mathbf{Frag}^{\text{maxintnodes(TC)}}_{\text{TC}}$
4. For all $f \in \mathbf{Frag}^{1}_{\text{TC}}$ assigns $\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f)$
5. For $n = 2, \ldots,$ maxintnodes(TC)
   - Create Grammar $G_n = (V_N, V_T, S, R, \Pi)$ where $R = \bigcup_{i=1}^{n-1} \mathbf{Frag}^{i}_{\text{TC}}$ and $\pi_G(f) = \pi_{\text{DOP}\alpha}(f) \quad \forall f \in R.$
   - Calculate $P_{G_n}(f)$ for all $f \in \mathbf{Frag}^{n}_{\text{TC}}$.
   - Assigns $\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - P_{G_n}(f)$ for all $f \in \mathbf{Frag}^{n}_{\text{TC}}$.

Figure 4.10: DOPα estimation procedure

proportionality factor α is selected.

The DOPα weight of fragment f in the fragment corpus is $\pi_{\text{DOP}\alpha}(f) = \boldsymbol{rf}_{\mathbf{RFrag}}(f) - P^{-}_{\text{DOP}\alpha}(f)$. To satisfy the condition $\pi_{\text{DOP}\alpha}(f) > 0$ for all fragments f, during the DOPα incremental estimator, the following inequality always holds $\alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) > P^{-}_{\text{DOP}\alpha}(f)$ for all $f \in \mathbf{Frag}_{\text{TC}}$.

In the beginning of this chapter, we showed by example that when $\alpha = 1$, there exist treebanks for which the proportional condition cannot be fulfilled. The question here is if we have a treebank of sample data from the language, does a real value α always exist so that DOPα can assign weights to fragments to satisfy the condition $P_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f)$ for all fragment f?

The answer to the above question is "Yes". We will show in theorem (4.4.1) that for any treebank, there is at least a range of values that DOPα can select proportionality factor α from so that the probabilities of fragments are proportional with factor α to their relative frequencies in the fragment corpus of the same root.

Theorem 4.4.1 *Let* TC *be a treebank, there exists $\alpha_0 > 0$ such that any α, $0 < \alpha \leq \alpha_0$, can be used as a valid proportionality factor of* DOPα *model.*

To prove the theorem, we need the following lemma.

Lemma 4.4.2 *For any tree T, let $\{\langle t_{11}, t_{12}\rangle, \langle t_{21}, t_{22}\rangle, \ldots, \langle t_{l1}, t_{l2}\rangle\}$ be the set of* *derivations* length two of T. The following inequality always hold:*

$$P^{-}_{\text{DOP}\alpha}(T) < \sum_{i=1}^{l} P_{\text{DOP}\alpha}(t_{i1}) \times P_{\text{DOP}\alpha}(t_{i2})$$

Proof:

$$P^{-}_{\text{DOP}\alpha}(T) = \sum_{\substack{\text{d is a } derivation^{*} \text{of } f \\ \text{length of } d \text{ is greater than 1}}} P_{\text{DOP}\alpha}(d) \tag{4.5}$$

Let $\mathcal{D}_2(T)$ be the set of *derivations** of T and the lengths of the derivations are greater than one . Equation (4.5) is rewritten as:

$$P^{-}_{\text{DOP}\alpha}(T) = \sum_{d \in \mathcal{D}_2(T)} P_{\text{DOP}\alpha}(d)$$

We partition the set $\mathcal{D}_2(T)$ so that the *derivations** that have the same last fragments belonging to the same set. $\mathcal{D}_{2,f}(T) = \{\langle f_1, f_2, \ldots, f_k\rangle | k \geq 2, f_k = f\}$ be the set of *derivations** in $\mathcal{D}_2(T)$ such that the last fragment in the derivation is f. Choose tree t such that $t \circ_T f = T$, so $\langle t, f\rangle$ is also a *derivation** length two of tree T. Also, if $\langle f_1, f_2, \ldots, f_k\rangle$, $k \geq 2$ is a *derivation**in $\mathcal{D}_{2,f}(T)$, we have $t = f_1 \circ_T f_2 \circ_T \ldots \circ_T f_{k-1}$.

Now we write the formula for the sum probabilities of *derivations** in $\mathcal{D}_{2,f}(T)$

$$\sum_{\langle f_1,\ldots,f_{k-1},f\rangle \, \in \, \mathcal{D}_{2,f}(T)} P_{\text{DOP}\alpha}(\langle f_1, \ldots, f_{k-1}, f\rangle) =$$

$$\begin{aligned} &= \sum_{\langle f_1,\ldots,f_{k-1},f\rangle \in \mathcal{D}_{2,f}(T)} \pi_{\text{DOP}\alpha}(f_1) \times \ldots \times \pi_{\text{DOP}\alpha}(f_{k-1}) \times \pi_{\text{DOP}\alpha}(f) \\ &= \underbrace{\left(\sum_{\langle f_1,\ldots,f_{k-1}\rangle \text{ is a } derivation^{*} \text{of } t} \pi_{\text{DOP}\alpha}(f_1) \times \ldots \times \pi_{\text{DOP}\alpha}(f_{k-1}) \right)}_{P_{\text{DOP}\alpha}(t)} \times \pi_{\text{DOP}\alpha}(f) \end{aligned} \tag{4.6}$$

$$= P_{\text{DOP}\alpha}(t) \times \pi_{\text{DOP}\alpha}(f) \leq P_{\text{DOP}\alpha}(t) \times P_{\text{DOP}\alpha}(f).$$

From equation (4.6): $\sum_{d \in \mathcal{D}_{2,f}(T)} P_{\text{DOP}\alpha}(d) \leq P_{\text{DOP}\alpha}(t) \times P_{\text{DOP}\alpha}(f)$.

Apply to equation (4.5)

$$
\begin{aligned}
P^{-}_{\text{DOP}\alpha}(T) &= \sum_{\substack{\text{d is a } derivation^{*} \text{of } f \\ \text{length of } d \text{ is greater than } 1}} P_{\text{DOP}\alpha}(d) \\
&= \sum_{f} \sum_{d \in \mathcal{D}_{2,f}(T)} P_{\text{DOP}\alpha}(d) \\
&\leq \sum_{\langle t, f\rangle \text{ is a } derivation^{*} \text{of } T} P_{\text{DOP}\alpha}(t) \times P_{\text{DOP}\alpha}(f) \\
&\leq \sum_{i=1}^{l} P_{\text{DOP}\alpha}(t_{i1}) \times P_{\text{DOP}\alpha}(t_{i2}).
\end{aligned}
$$

QED

Using the result in lemma (4.4.2) to prove theorem (4.4.1) as follows:

Proof of theorem (4.4.1). A real value α is a valid proportionality factor of the DOPα model only if the condition $P^{-}_{\text{DOP}\alpha}(f) \leq \alpha \times \boldsymbol{rf}_{\textbf{RFrag}}(f)$ for all fragments f in the fragment corpus when the DOPα estimates the weights of fragments incrementally.

Denote $\text{Der}_2(f) = \{\langle f_1, f_2\rangle, \text{ where } \langle f_1, f_2\rangle \text{ is a } derivation^{*} \text{ length 2 of } f\}$.

Apply lemma (4.4.2):

$$
\begin{aligned}
P^{-}_{\text{DOP}\alpha}(f) &\leq \sum_{\langle f_1, f_2\rangle \in \text{Der}_2(f)} P_{\text{DOP}\alpha}(f_1) \times P_{\text{DOP}\alpha}(f_2) \\
&= \sum_{\langle f_1, f_2\rangle \in \text{Der}_2(f)} \alpha \times \boldsymbol{rf}_{\textbf{RFrag}}(f_1) \times \alpha \times \boldsymbol{rf}_{\textbf{RFrag}}(f_2) \\
&= \alpha^2 \times \sum_{\langle f_1, f_2\rangle \in \text{Der}_2(f)} \boldsymbol{rf}_{\textbf{RFrag}}(f_1) \times \boldsymbol{rf}_{\textbf{RFrag}}(f_2)
\end{aligned}
\tag{4.7}
$$

Now choose α such that

$$
\alpha^2 \times \sum_{\langle f_1, f_2\rangle \in \text{Der}_2(f)} \boldsymbol{rf}_{\textbf{RFrag}}(f_1) \times \boldsymbol{rf}_{\textbf{RFrag}}(f_2) \leq \alpha \times \boldsymbol{rf}_{\textbf{RFrag}}(f)
$$

or

$$
\alpha \leq \frac{\boldsymbol{rf}_{\textbf{RFrag}}(f)}{\sum_{\langle f_1, f_2\rangle \in \text{Der}_2(f)} \boldsymbol{rf}_{\textbf{RFrag}}(f_1) \times \boldsymbol{rf}_{\textbf{RFrag}}(f_2)}.
$$

We will have $P^{-}_{\text{DOP}\alpha}(f) \leq \alpha \times \boldsymbol{rf}_{\textbf{RFrag}}(f)$

Choose $\alpha_0 = \min_{f \in \mathbf{Frag}_{\text{TC}}} \dfrac{\boldsymbol{rf}_{\mathbf{RFrag}}(f)}{\sum_{\langle f_1, f_2 \rangle \in \text{Der}_2(f)} \boldsymbol{rf}_{\mathbf{RFrag}}(f_1) \times \boldsymbol{rf}_{\mathbf{RFrag}}(f_2)}$

For each $\alpha \leq \alpha_0$, we have

$$P^{-}_{\text{DOP}\alpha}(f) \leq \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) \quad \forall f \in \mathbf{Frag}_{\text{TC}}$$

QED

The aim for DOPα is to assign weights to fragments to satisfy the proportionality condition. In practice, the selected proportionality factor will influence the estimator's result, we will discuss this in the section on empirical experiment. In this theoretical frame work of DOPα, the α chosen is any positive value such that the condition $\alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) = P_{\text{DOP}\alpha}(f)$ is satisfied for all fragments $f \in \mathbf{Frag}_{\text{TC}}$. Theorem (4.4.1) showed that for any treebank there exists a α_0 such that any $\alpha \leq \alpha_0$ can be a valid proportionality factor of the model. The theorem did not claim that if $\alpha > \alpha_0$, it is not a valid proportional factor. This gives a hint how to select proportionality factor as follows.

1. Start $\alpha = 1$
2. Partition $\mathbf{Frag}_{\text{TC}}$: $\mathbf{Frag}_{\text{TC}} = \mathbf{Frag}^1_{\text{TC}} \cup \mathbf{Frag}^2_{\text{TC}} \cup \ldots \cup \mathbf{Frag}^{\text{maxintnodes(TC)}}_{\text{TC}}$
3. For all $f \in \mathbf{Frag}^1_{\text{TC}}$ assigns $\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f)$
4. For $n = 2, \ldots, \text{maxintnodes(TC)}$
   - Create Grammar $G_n = (V_N, V_T, S, R, \Pi)$ where
     $R = \bigcup_{i=1}^{n-1} \mathbf{Frag}^i_{\text{TC}}$ and $\pi_G(f) = \pi_{\text{DOP}\alpha}(f) \quad \forall f \in R$.
   - For $f \in \mathbf{Frag}^i_{\text{TC}}$
     Calculate $P_{G_n}(f)$
     If $\alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) < P_{G_n}(f)$ go to step 6
     Else assign $\pi_{\text{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - P_{G_n}(f)$
5. Return α value.
6. Reduce α. Go back to step 3.

Figure 4.11: Determining α procedure

At the beginning, we assign α to some starting value, then check the validity of α. If it is not valid, we reduce the selected α. At some point, we will get to a value α that is a valid proportional factor of the model. A value α is a valid one if $P^{-}_{\mathrm{DOP}\alpha}(f) < \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) \quad \forall f \in \mathbf{Frag}_{\mathrm{TC}}$. The selection α algorithm is summarized in the Figure (4.11).

4.5 DOPα and Language Models

Natural language processing has applications ranging from speech recognition, text summarization to machine translation. In these applications, often probabilities of strings are computed to avoid ambiguity. The usual approach is to define a *language model* which assigns a probability distribution over all the parse trees in the language. The probability of a string is the sum of probabilities of all its possible parse trees. The language model thus also assigns a probability distribution over all strings of the language.

$$\sum_{s \text{ is a string of the language}} P(s) = 1$$

More formally, in an application, we are given an input I and need to output a string O^* to be the message conveyed or encoded by the input I. The probability $P(O|I)$ is computed by Bayes' rule as follows:

$$P(O|I) = \frac{P(O) \times P(I|O)}{P(I)}$$

Since the input I is fixed, it is the same for all the output string O so $P(I)$ can be ignored; $P(I|O)$ is determined by the input evidence I; $P(O)$ is determined by the language model.

$$P(O) = \sum_{T \text{ is a parse tree of } O} P(T, O)$$

The NLP application uses a statistical parsing model $\mathcal{M}$ to determine the possible parse trees T of string O and $P_{\mathcal{M}}(T, O)$ or $P_{\mathcal{M}}(T)$ as an approximation of its true distribution in the language.

In DOP1, the fragments are assumed to be independent and the weight of a fragment is considered as the probability to substitute the fragment into the node of the root label. The weights of fragments of the same root sum up to one, thus the probabilities of all full parse trees also sum up to one. In our estimation, DOPα, fragments are not considered to be independent. The weight of a fragment is assigned to account for the co-occurrences of its subtrees that the fragment's subtrees fail to capture.

$$\pi_{\mathrm{DOP}\alpha}(f) = \alpha \times \boldsymbol{rf}_{\mathbf{RFrag}}(f) - P^{-}_{\mathrm{DOP}\alpha}(f)$$

where $P^{-}_{\text{DOP}\alpha}(f) = \sum_{\text{length of } d \text{ is greater than } 2} P_{\text{DOP}\alpha}(d)$

DOPα is more flexible for weight assignment in that the sum of fragments' weights of the same root is not equal to one but to a positive number less than one. DOPα does not return the approximation of the probability of a tree in the language but a proportion of the probability of the tree. By normalization, we will obtain approximations of the true probabilities of trees in the language.

The statistical parsing is used as a sub-module of the other applications as explained before. The proportionality to the true probability property of DOPα does not affect the adaptability of the model. As we will explain in the following, the model can return the $P_{\text{DOP}\alpha}(T)$ directly as the other parsing models without affect to the final result of the applications.

Let a be the normalization factor.

$$\begin{aligned}
O^* &= \arg\max_O P(O) \times P(I|O) \\
&= \arg\max_O \sum_{T \text{ is a parse tree of } O} P(T) \times P(I|O) \\
&= \arg\max_O \sum_{T \text{ is a parse tree of } O} (a \times P_{\text{DOP}\alpha}(T)) \times P(I|O) \\
&= \arg\max_O \sum_{T \text{ is a parse tree of } O} P_{\text{DOP}\alpha}(T) \times P(I|O)
\end{aligned}$$

In the above equation, the normalization factor a does not appear. Even if the probabilities of all full parse trees do not sum up to one, the DOPα model can be used with other applications.

4.6 Summary

In this chapter, we presented an estimator that satisfies the rank consistency property. The treebank is viewed as a multi-set of fragments as well as multi-set of full parse trees. We also gave an extension of the definition of the traditional derivation and probability to an arbitrary tree. The probability of fragments are proportional to their relative frequencies in the treebank.

Chapter 5

Empirical Results

In this chapter, we provide the empirical results of DOPα and compare them with DOP1's results when using the same training and testing data. All the experiments were trained and tested on the OVIS [1] corpus.

5.1 OVIS treebank

OVIS treebank is a speech-based Dutch language corpus which was extracted from telephone conversations about the railway timetable. Each sentence in the corpus is syntactically and semantically annotated.

The parse tree of sentence "ik wil naar den haag centraal station" ("I want to go to den haag central station") depicted in Figure (5.1) is an annotated sentence from the OVIS. The OVIS corpus contains 10,049 parse trees, 548 non-terminal symbols and 965 terminal symbols. The average number of words per sentence is 3.47.

The graph in Figure (5.2) shows the percentage of sentences with respect to sentences' word lengths in the OVIS. Most of the sentences consist of less than 6 words and about 30% of the sentences are one-word sentences. Since one-word sentences are easy to parse, the high statistics results of a test set might be explained by the high percentage of one-word sentences. To obtain a more meaningful result, we remove all one-word trees from the test set and report the results for sentences that contain more than one word.

[1] OVIS stands for Openbaar Vervoer Informatie Systeem (Public Transport System)

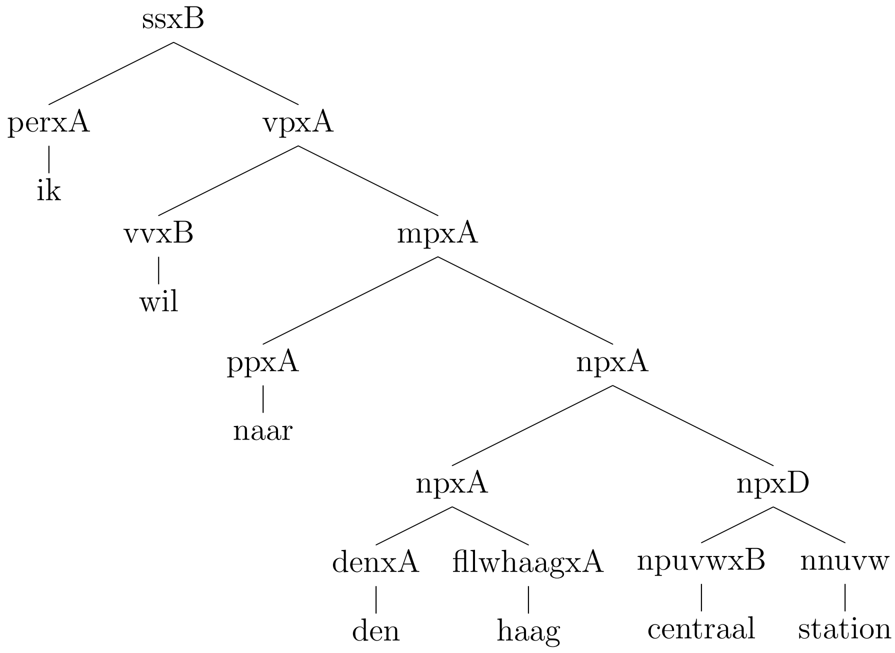


Figure 5.1: An example of a parse tree in the OVIS corpus

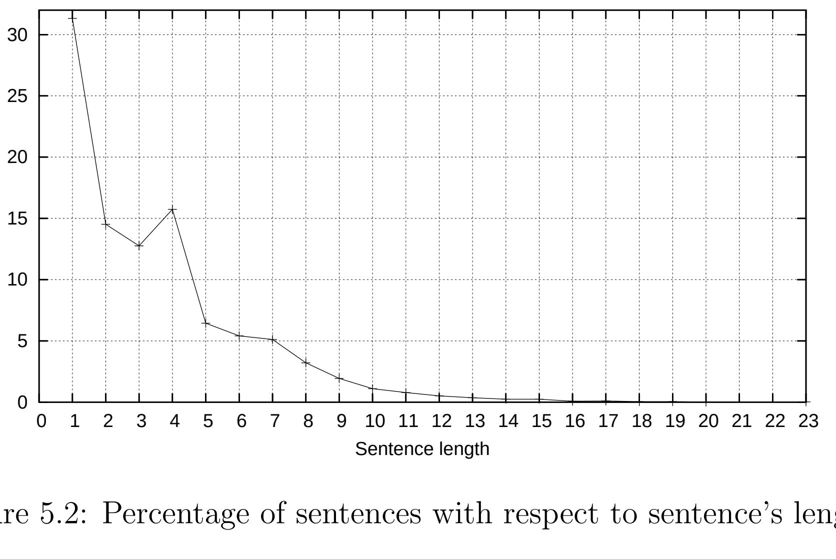


Figure 5.2: Percentage of sentences with respect to sentence's length

5.2 Testing

The OVIS corpus is randomly split: 90% of the corpus is reserved for the training process and the remaining 10% of the corpus is for the testing. To avoid that the results are due to the random training/testing splitting, the experiments were carried on five different such random splits. While testing for DOP1 and DOPα depth 5, there were approximately 1,500,000 fragments extracted. The large number of fragments causes overflow errors during the testing of DOP1 and DOPα experiments of two splits. Unless stated otherwise, the reported results are the average of the other three splits' results.

We used Khalil Sima'an's DOPDIS parser for both DOP1 and DOP$\alpha\dot{}$The DOPDIS is available at http://staff.science.uva.nl/˜simaan/dopdis.

5.2.1 Metrics

We evaluate the parsers by their exact match, bracketing recall and label precision percentages. Exact match is the strictest criterion that count one for sentence that the parser gets the parse tree completely right and zero for otherwise. The exact match evaluation is the most sensible one that most of the parsing models try to maximize. Bracketing Recall and Bracketing Precision measures on pieces of trees how likely the parser's results and the expected correct trees are.

Exact match

$$\text{Exact match } \% = \frac{\text{Number of correct parses}}{\text{Number of sentences in the test set}} \times 100$$

Bracketing Recall

Bracketing Recall measures the ratio of the correct brackets and the total number of brackets in the correct parse trees.

$$\text{Bracketing Recall } \% = \frac{\text{Number of correct bracketing}}{\text{Number of bracketing in the correct parse trees}} \times 100$$

Bracketing Precision

Bracketing Precision measures the ratio of the correct brackets and total number of brackets returned by the parser.

$$\text{Bracketing Precision } \% = \frac{\text{Number of correct bracketing}}{\text{Number of bracketing in the parser's parse trees}} \times 100$$

5.2.2 The DOPα results of a random proportional factor

We test the model as suggested in chapter 4. The proportional value α start from 1 and then α is reduced if it is not a valid proportional factor. In our implementation, the value α is reduced by dividing to 2. By that we get a random valid proportional factor. We present the results of the experiments which the depths upper bound of fragments are from one to five. The following table lists the proportional factor value α found for each depth.

Depth	α	
Depth 1	1	
Depth 2	0.00390625	(2^{-8})
Depth 3	0.0009765625	(2^{-10})
Depth 4	0.0009765625	(2^{-10})
Depth 5	0.0009765625	(2^{-10})

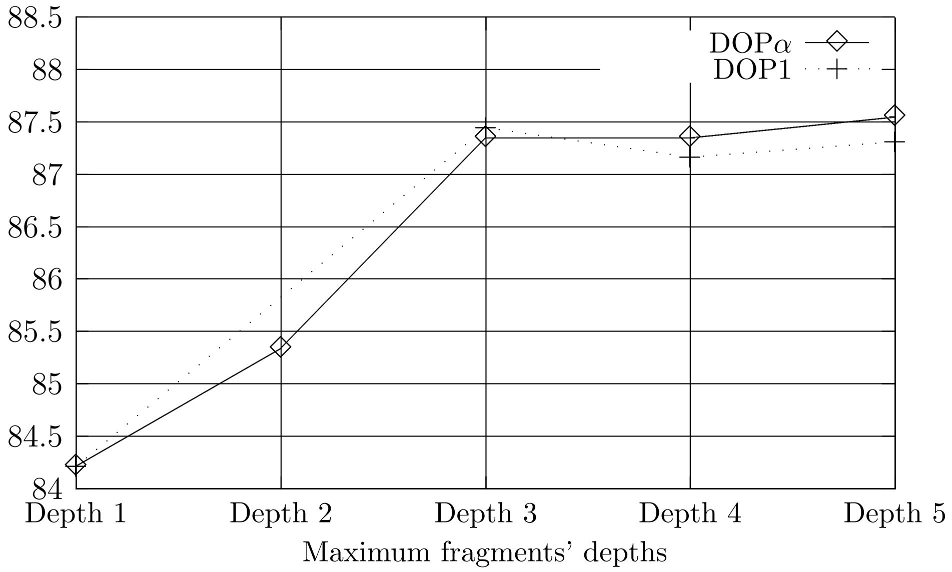


Figure 5.3: Exact match percentage as a function of the depth upper bound (a random α selection)

Figure (5.3), (5.4) and (5.5) show the exact match, bracketing precision, bracketing recall percentage when the model used the first found valid proportional factor values. The DOP1's results increase and get peak when the maximum depth of the extracted fragments is three, the experiments' results start to decrease at the depth four due to the fact that the DOP1's weight assignment causes the model bias toward big trees. DOPα's results, on the other hand, steadily increase from

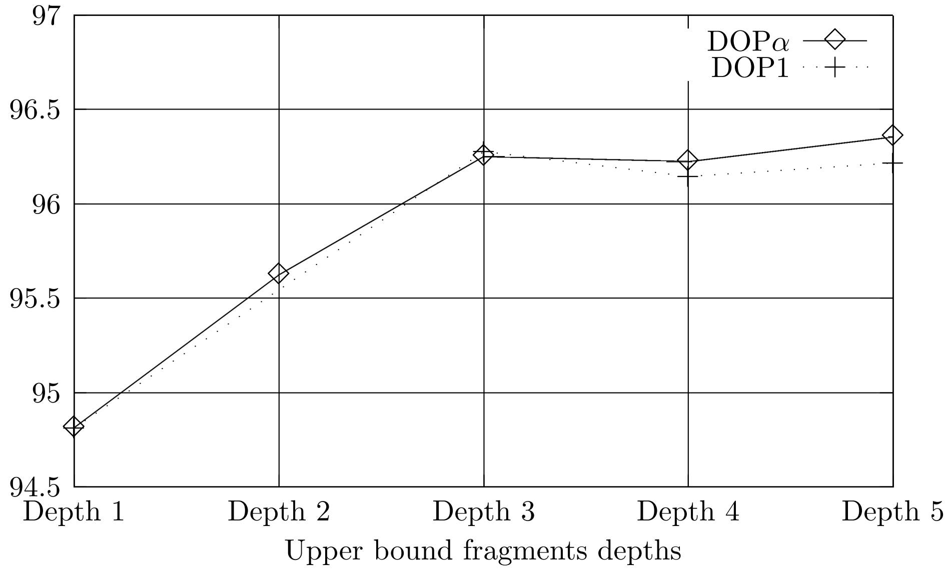


Figure 5.4: Bracketing Precision percentage as a function of the depth upper bound (a random α selection)

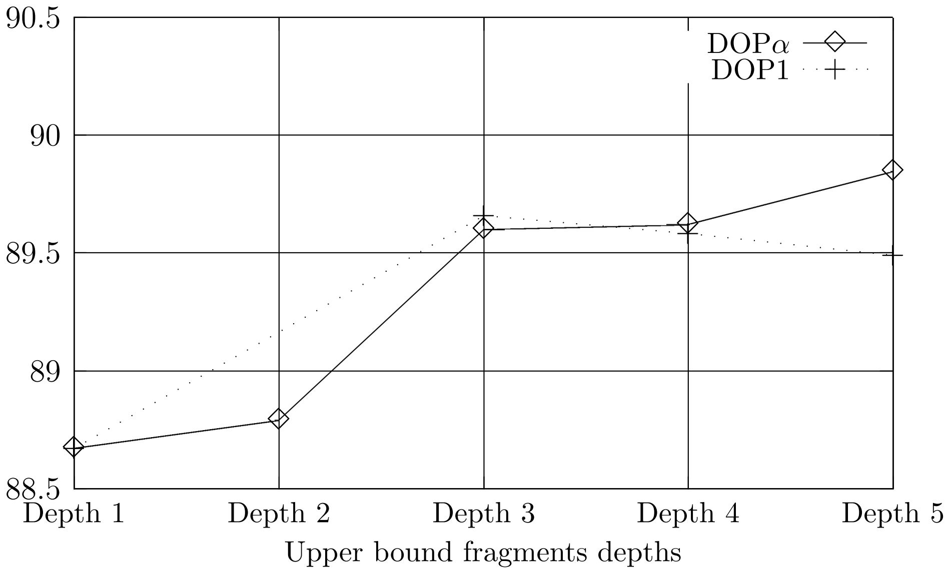


Figure 5.5: Bracketing Recall percentage as a function of the depth upper bound (a random α selection)

depth one to depth five and start to outperform DOP1's result at the depth four in all three evaluations. This is in line with our theoretical motivation presented earlier in this thesis. In DOPα, the weights of the big fragments are assigned just to capture the interdependencies of their sub-structures. Thus, the more fragments are extracted, the more contexts and dependencies of the language are captured giving the model better performance.

5.2.3 Proportionality Factor selection

To test whether the DOPα results above are due to some random properties of proportionality factor α, we select ten different valid values of α from $0.0001, 0.0002, \ldots, 0.0009, 0.001$ and tests the model for each value in the first splitting.

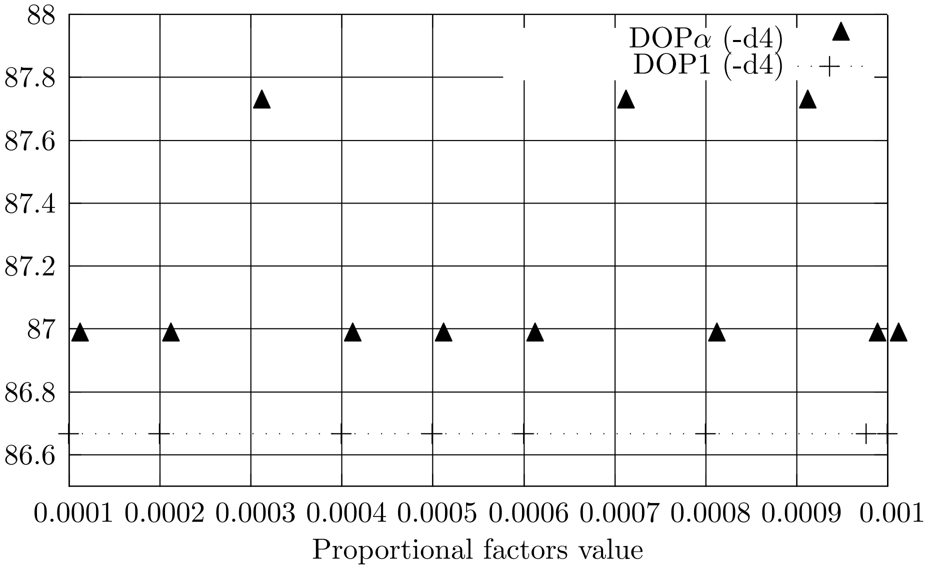


Figure 5.6: Exact match percentage as a function of the proportionality factors

Figure (5.6) shows the exact match results of DOPα when testing with different proportional factors. There are two different values of exact match percentages. In our tested splitting of training and testing, the results get high value when the proportional factors are $0.0003, 0.0007, 0.0009$ and get low value when tested with other proportional factors. But even when the proportionality factors cause DOPα the worst performance, DOPα's results still outperform DOP1. We suggest that the optimal proportionality factor α can be estimated in the training process by the cross-validation method. The training corpus is split into different sets, using each set for held-out testing. The selected α starts as a first random valid one for the first held-out testing and is then refined after each test.

5.2.4 Learning curve

In order to know the DOPα's results with different training corpus sizes, we successively test the model using the training corpus from 20%, 30%, ..., to 100% data of the original training corpus. Figure (5.7) shows the exact match percentages as the training corpus size increases.

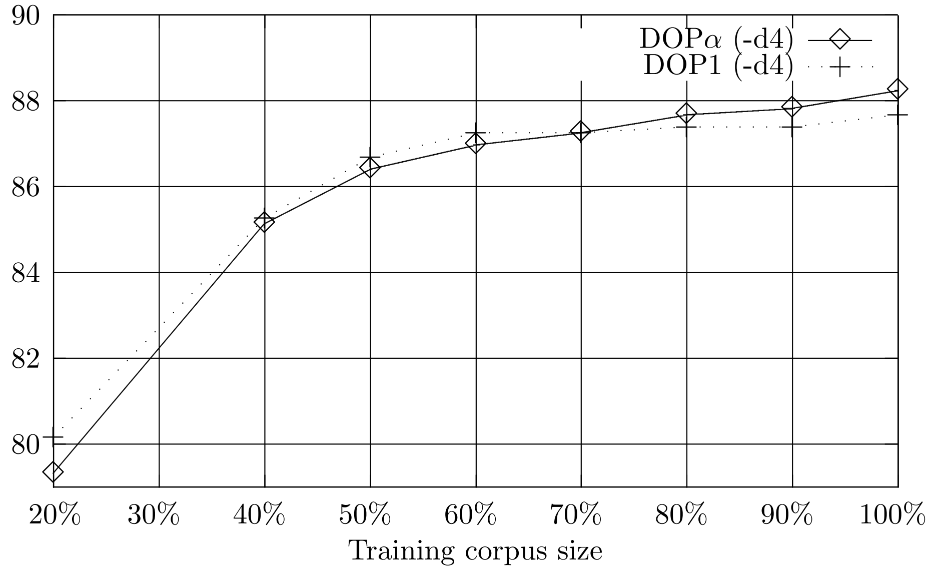


Figure 5.7: Exact match percentage as a function of the training corpus size

DOP1 exact match results steadily increase as the training corpus size increases from 20% to 60%. However, there is no significant difference of DOP1 exact match percentages when the training corpus size increases from 60% to 100%. Moreover, when the training corpus size increases from 60% to 70% or from 80% to 90%, the DOP1 exact match results stay the same. DOPα results on the other hand steadily increase with the training corpus size and start to outperform the DOP1 results when the training corpus size is 70% the original one. This suggests that DOPα might have outperformed DOP1 even clearer if more data had been available.

5.3 Summary

We have provided the empirical results of DOPα and compared the results with the DOP1 model. The DOPα results start to outperform DOP1 when the maximum depth of fragments is at least four. Also, the proportional factor α plays an important role in the estimation result. When using the experiment in which the maximum depth of fragments is four, the DOPα results outperform the DOP1 results even with the proportional factor selection that gives the worst performance.

We also saw that the DOPα performance increases as the training corpus size increases.

Chapter 6

Conclusion

The goal of this thesis has been to show a non-trivial DOP estimator for which the treebank is regarded as a sample of the distribution in the language. To avoid the overfitting problem, the treebank is considered as a multiset of fragments. In the new estimator, the weights of fragments are assigned so that the parse tree probabilities of fragments are proportional to their relative frequencies in the treebank's fragment corpus. Statistical parsing belongs to the set of problems in which only partial information is available.

Laplace 's "Principle of Insufficient Reason" stated that in the absence of information regarding a set of alternative solutions, one should assign equal probabilities to all possibilities [Laplace, 1829]. The rank consistency property is defined as an adaption of Laplace 's principle in the statistical parsing area: if one tree appears in the treebank with higher frequencies than the other tree, the estimator also assigns the former tree higher probability than the latter tree. The new estimator, DOPα, was proved to be rank consistent.

The theoretical properties of the model are validated by the empirical results. We test the model on the OVIS corpus. In our experiments, the new model results improve upon the original DOP1 results.

Also, we study the relationship of consistency property and rank consistency property in estimation theory. In estimation theory, consistency is a desirable property. Is our estimation consistent? The DOPα model estimates probability of a tree as a proportional of its relative frequency in the treebank. Because the probabilities of trees sum up to one, when the treebank size grows to infinity, intuitively, the DOPα will estimate the probability of a tree as its relative frequency in the treebank. The DOP$_{\mathrm{MLE}}$ was proved to be consistent, thus DOPα is also consistent. Further work of DOPα could provide the formal justification of the consistency of the model.

We can only conclude that our model potentially provides better results in the area of Data-Oriented Parsing. In our existing work, the proportional factor value is just selected to be a valid one. However, the experiment in section (5.2.3) shows that the proportional factor value has influence on the final testing results. This brings up a deeper question. How the proportional factor value is selected in the training process such that the model gets the optimal results? In section (5.2.3), we suggest one possible solution. The proportional factor value maybe selected using held-out estimation. The first selected proportional value is a random valid one. The value is then refined after each test on the held-out corpus. Possible future work could be an algorithm to get the optimal result of the selected proportional factor value.

Bibliography

[Black *et al*, 1993] E.Black, R.Garside and G.Leech. 1993. *Statistically-Driven Computer Grammars of English.* The IBM/Lancaster Approach, Ropodi: Amsterdam-Atlanta.

[Bod, 1992] Bod, Rens. 1992. *Data Oriented Parsing (DOP).* Proceedings COLING'92, Nantes, France.

[Bod, 1993] Bod, Rens. 1993. *Using an Annotated Corpus as a Stochastic Grammar.* Proceedings EACL'93, Utrecht, The Netherlands.

[Bod, 1995] Bod, Rens. 1995. *Enriching Linguistics with Statistics: Performance Models of Natural Language.* ILLC Dissertation Series 1995-14, University of Amsterdam. (ftp://ftp.fwi.uva.nl/pub/theory/illc/dissertations/DS-95-14.text.ps.gz)

[Bod & Scha, 1996] Bod, Rens, and Remko Scha. 1996. *Data-Oriented Language Processing: An Overview.* Research report nr. LP-96-13, ILLC Research reports, University of Amsterdam.

[Bod, 1998] Bod, Rens. 1998. *Beyond Grammar: An Experience-Based Theory of Language.* Stanford, CA: CSLI Publications.

[Bod, 2000] Bod, Rens. 2000. Parsing with the Shortest Derivation. *Proceedings COLING-2000.* Saarbruecken, Germany. Available at `staff.science.uva.nl/~rens`.

[Bonnema *et al.*, 1999] Bonnema, Remko, Paul Buying, and Remko Scha. 1999. A New Probability Model for Data-Oriented Parsing. *Proceedings of the Amsterdam Colloquium 1999.* Amsterdam. Available at `citeseer.nj.nec.com/bonnema99new.html`.

[Bonnema, 2003] Bonnema, Remko. 2003. Probability Models for DOP. In: Bod, R., Scha, R., and Sima'an, K. *Data Oriented Parsing.* CSLI Publications, Stanford University. Stanford, California, USA.

[Booth & Thompson, 1973] Booth, Taylor and Thompson. 1973. *Applying Probability Measure to Abstract Languages*. IEEE Transaction on Computers C-22(5): 442-450.

[Brill & Mooney, 1997] Eric Brill, Raymond J. Mooney. 1997. *An overview of empirical natural language processing - Natural Language Processing*. AI Magazine, Winter, 1997.

[Buratto, 2003] Burrato, L. 2002. *Backoff as Parameter Estimation for DOP models*. Master of Logics series (MoL-2002-07). ILLC Scientific Publications, Institute for Logic, Language and Computation (ILLC), Amsterdam. The Netherlands.

[Goodman, 1998] Goodman, Joshua. 1998. *Parsing Inside-Out*. Ph.D. thesis, Harvard University, Massachusetts. Available at `http://citeseer.nj.nec.com/article/goodman98parsing.html`

[Guiasu & Shenitzer, 1998] S. Guiasu and A. Shenitzer. 1998. *The principle of maximum entropy*. The Mathematical Intelligencer, 7(1), 1985. (An overview paper)

[Hoogweg, 2000] Hoogweg, L. 2000. *Extending DOP with Insertion*. Master of Logics series. ILLC Scientific Publications, Institute for Logic, Language and Computation (ILLC), Amsterdam. The Netherlands.

[Johnson, 2002] Johnson, Mark. 2002. *The DOP Estimation Method Is Biased and Inconsistent*. Computational Linguistics 28(1), pages 71-76. Available at `cog.brown.edu/~mj/Publications.htm`.

[Laplace, 1829] Laplace, P.S. 1829. *Essai philosophique sur les Probabilités.*. H.Remy. Fifth edition.

[Manning & Schütze, 1999] Manning, Christopher, and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts.

[Krenn & Samuelsson, 1997] Krenn, Brigitte, and Christer Samuelsson. 1997. *The Linguist's Guide to Statistics—Don't Panic*. `citeseer.nj.nec.com/krenn97linguists.html`.

[Prescher, 2003] Prescher, Detlef. 2003. *A Tutorial on the Expectation-Maximization Algorithm Including Maximum-Likelihood Estimation and EM Training of Probabilistic Context-Free Grammars*. Presented at ESSLLI-03, Vienna, Austria. Available at `http://staff.science.uva.nl/~prescher/papers/`.

[Prescher *et al.*, 2004] Prescher, Detlef, Remko Scha, Khalil Sima'an, and Andreas Zollmann. 2004. On the Statistical Consistency of DOP Estimators. To be published elsewhere. Available at `http://staff.science.uva.nl/~azollman/publications.html`.

[Scha, 1990] Scha, Remko. 1990. Taaltheorie en Taaltechnologie; Competence en Performance. In: de Kort, Q. A. M., and Leerdam, G. L. J., (eds.), *Computertoepassingen in de Neerlandistiek*, Almere: Landelijke Vereniging van Neerlandici (LVVN-jaarboek). English translation as: Language Theory and Language Technology; Competence and Performance; `http://iaaa.nl/rs/LeerdamE.html`

[Shao, Jun, 1999] Shao, Jun. *Mathematical Statistics*. Springer Verlag, NewYork.

[Sima'an, 1999] Sima'an, Khalil. *Learning Efficient Disambiguation*. PhD dissertation (University of Utrecht). ILLC dissertation series 1999-02, University of Amsterdam. Amsterdam.

[Sima'an, 2000] Sima'an, Khalil. *Tree-gram Parsing: Lexical Dependencies and Structural Relations*. Proceedings ACL'2000, HongKong, China.

[Sima'an & Buratto, 2003] Sima'an, Khalil, and Luciano Buratto. *Backoff Parameter Estimation for the DOP Model*. Proceedings of the European Conference on Machine Learning (ECML'03). Dubrovnik, Croatia. Available at `staff.science.uva.nl/~simaan`

[Resnik, 1992] Philip Resnik. *Probabilistic tree-adjoining grammar as a framework for statistical natural language processing*. Proceedings of the 14th conference on Computational linguistics - Volume 2. Nantes, France.

[Zollmann, 2004] Zollmann, A. 2004. *A Consistent and Efficient DOP estimation*. Master of Logics series. ILLC Scientific Publications, Institute for Logic, Language and Computation (ILLC), Amsterdam. The Netherlands.

Index