A comparison of Double-DOP and DOP*

Benno Kruit
10576223
benno.kruit@student.uva.nl

Sara Veldhoen
10545298
sara.veldhoen@student.uva.nl

Abstract

This paper investigates two existing estimators in the Data Oriented Parsing approach to natural language syntax: Double-DOP and DOP*. We assess the theoretical and practical differences between these estimators by comparing the grammars they derive and their performance on part of the WSJ treebank. Performing estimation with a split corpus increased the weight of large fragments, and the consistent estimator did not perform better than the non-consistent estimators. We visualize the correspondance between the grammar distributions and their theoretical properties.
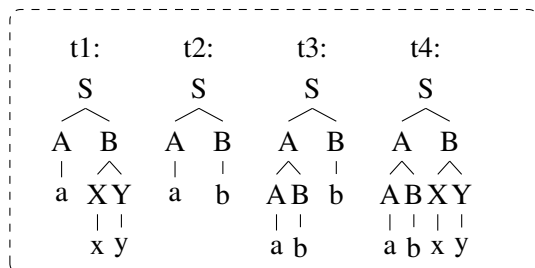
1 Introduction

Figure 1: A toy treebank

A common approach to natural language syntax, is to view the structure of sentences as constituent trees. An artificial example of a treebank is given in figure 1. Constituent trees can be described by a *Context Free Grammars* (CFGs), such that all trees are built up from rules that each describe the production (children nodes) of a single node (parent) in the tree. When building an empirical model of observed parse trees, these rules are extended with probabilities to form a *probabilistic CFG* (PCFG). This gives the trees that are 'generated' by these rules their own probability, which makes it a statistical model of a distribution over natural language syntax.

A CFG models each production as an independent event, but natural language probably has more complex interdependencies. To this end, grammars can be enriched (e.g. by Markovisation, that include information about grandparent or sibling nodes.

1.1 DOP

Data-Oriented Parsing (DOP), as first introduced in (Scha, 1990), takes a different approach. It models the language with a *Probabilistic Tree Substitution Grammar* (PTSG). The trees in the treebank are taken apart, which results in *fragments* of arbitrary depth[1]. A fragment is a connected subgraph of a tree such that it corresponds to context-free productions in that tree, i.e. each node must have either have children with the same labels as in the original tree, or no children at all. This is illustrated in figure 2. Note that not all fragments from the treebank in figure 1 are displayed. We see that each level-one fragment corresponds to a CFG rule. The *symbolic grammar* refers to the set of fragments (that receive a non-zero weight) in a grammar.

Fragments can be combined in a *derivation* to build syntactic structures. A step in a derivation is a composition, denoted by the symbol $\circ$. For instance, tree t_1 can be derived as $t_1 = f_2 \circ f_{10}$. We follow the convention to only allow left-most derivations. This means that the left-most non-terminal node in a fragment f_1 must correspond to the root node of f_2 in order to derive $f_3 = f_1 \circ f_2$

A weight must be assigned to each fragment. This can be done by counting how often it occurs in the treebank compared to others with the same root, yielding the *relative frequency estimate*. The probability of a derivation is the product of the

---

[1]Fragments are sometimes referred to as 'subtrees' in literature

Figure 2: Extracted fragments

probability of the fragments it uses. Note that a single tree can be the result of different derivations. Therefore probability of a tree is the sum of the probabilities of all its derivations.

Theoretical issues It has been shown that DOP (in its original formulation) is biased and inconsistent (Johnson, 2002), which are both assumed to be bad properties of an estimator in general. As we will see, bias is not necessarily a bad thing. In fact, Zollman proves in (Zollmann and Sima'an, 2005) that any non-overfitting estimator is biased. Furthermore, he shows that it is possible to define a DOP-estimator, DOP*, that is provably consistent.

Practical issues In its original formulation, DOP takes the trees apart in all possible ways. The number of fragments is exponential in the length of the sentences, thus the size of the symbolic grammar would be far too huge to be computationally feasible. Different approaches have been taken to reduce the symbolic grammar, e.g. by sampling or by applying a smart algorithm. Sangati and Zuidema have described a very efficient DOP variation, Double-DOP, that also produces excellent results (Sangati and Zuidema, 2011).

1.2 Our contribution

The main question of this work concerns the correspondence between theoretical properties and practical performance. Is it possible to gain theoretically attractive properties without performance losses? Can we reduce overfitting in an already well-performing approach by introducing

the right bias? How do consistent grammars and well-performing grammars differ in their fragment weight distribution?

We trained three grammars related to the different approaches on part of the WSJ treebank. The theoretical cornerstone of the DOP* approach, extracting fragments from shortest derivations, lead to smaller grammars. The measure that should prevent overfitting, estimations on random splits of the corpus, increased the weight of large fragments. Neither approach improved on the performance of the Double-DOP approach. However, the comparision of grammars leads to profound insight into the internal distribution of the model. This elucidates the correspondance between the fragment weights and theoretical properties.

1.3 Outlook

Section 2 elaborates the notions of consistency and bias and their relation to overfitting. In section 3, we outline two approaches that tackle the reduction of the symbolic grammars: Double-DOP and DOP*. This report focuses on a comparison of these approaches. Section 4 offers a detailed comparison as well as a description of the experiments we conduct. We investigate the differences between the grammars produced by Double-DOP DOP*. The algorithms can be decomposed into two parts. We also analyze the impact of the partial choices by mutually using these parts.

In section 5, we present our findings and provide an analysis.

2 Statistics: Consistency and Bias

Linguistic studies of syntax mostly concern *competence* models, which describe the structures that appear in a language. In contrast, a *performance* model of language is an estimate of the probability of observing a parse tree in language use. It treats language as a statistical distribution over syntactic structures.

Let Ω be the set of all possible parse trees. The distribution P_Ω then describes the language, where $P_\Omega(t)$ is the probability of observing a tree $t \in \Omega$. Using a sample of parse trees from the language, an *estimator* EST builds a statistical model. A parser then uses that statistical model to predict the correct parse tree of sentences. A sample $X \in \Omega^n$ from the language is called a *corpus* or *treebank* of size n, which makes $\mathrm{EST}(X)$ an estimator trained on a sample. If $\mathcal{M}$ is the set of probability distri-

butions over Ω, then $P_\Omega \in \mathcal{M}$ and $\text{EST}(X) \in \mathcal{M}$.

In theory, an estimator should make exactly the right estimations of probabilities if it's given an infinite amount of data. That is to say, it should *converge* to the true distribution. If an estimator converges in the limit, that estimator is *consistent*. However, given a finite amount of data, the estimator will probably not generate the correct distribution. The distance between the true distribution P^* and an estimate P is called the *loss* of that estimate. The loss can be defined in different ways, but the most popular is the *mean squared difference*:

$$\mathcal{L}(P, P^*) = \sum_{t \in \Omega} P^*(t)(P^*(t) - P(t))^2$$

From a true distribution, it's possible to calculate the expected loss of an estimator trained on a treebank of a certain size. This is the *risk* or *error* of that estimator given a sample size and a distribution. When the sample size approaches the limit, the error of an estimator should diminish. With these definitions, it is possible to define estimator consistency when sampling $X \in \Omega^n$ from P_Ω:

$$\lim_{n \to \infty} \mathbf{E}[\mathcal{L}(\text{EST}(X), P_\Omega)] = 0$$

In its original formulation, DOP was defined using a *relative frequency estimate*, by counting how often it occurs in the treebank compared to others with the same root. However, it has been shown (Johnson, 2002) that in this case, the RF estimator is inconsistent.

Another property of an estimator is its *bias*, which is defined as the difference between the true probability and the expected estimate. It has been proven that any unbiased DOP estimator will overfit a treebank by assigning zero-probabilites to trees outside the corpus. To prevent overfitting, it is therefore necessary to introduce a bias that assigns a non-zero probability to unseen trees. By maximizing the probability of a corpus different from the one from which the fragments are extracted, we will see that it is possible to minimize overfitting.

3 Existing Models: Double-DOP and DOP*

In this section, we outline two approaches to constrain the extraction of fragments: Double-DOP

and DOP*. Furthermore, we discuss the similarities and dissimilarities for these two approaches.

3.1 Double-DOP

In the following, we discuss Double-DOP as it was presented in (Sangati and Zuidema, 2011). In this model, no unique fragments are extracted from the dataset: if a construction occurs in one tree only, it is probably not representative for the language. This is carried out by a dynamic programming algorithm using tree-kernels. It iterates over pairs of trees in the treebank, looking for fragments they have in common. In fact, only the largest shared fragment is stored.

The symbolic grammar that is the output of this algorithm is not guaranteed to derive each tree in the training corpus. Therefore all fragments of depth one, constituting the set of PCFG-productions, are also added.

The emphasis of Double-DOP is on the extraction method for determining the symbolic grammar. However, it was also implemented with different estimators. The estimation is done in a second pass over the treebank, gathering frequency counts for the fragments in the grammar. We will use the relative frequency estimate, which was empirically found to perform best (Sangati and Zuidema, 2011).

3.2 DOP*

In DOP* (Zollmann and Sima'an, 2005), a rather different approach is taken called held-out estimation. The treebank is split in two parts, the *extraction corpus* (EC) and a *held-out corpus* (HC). An initial set of fragments is extracted from the EC, containing all the fragments from its trees. The weights are then determined so as to to maximize the likelihood of HC, under the assumption that this is equivalent to maximizing the joint probability of the *shortest derivations* of the trees in HC.

The weight of a fragment is its relative frequency of occurring in a shortest derivation, and all fragments that do not occur in such a derivation are removed from the symbolic grammar. In fact, a tree could have several shortest derivations. The probability mass is divided over the fragments taking parts in the different derivations in that case. Furthermore, some trees in HC may not be derivable at all, which indicates that the grammar does not have complete coverage: a sentence in the test set might also be underivable.

To maximize coverage of the grammar, DOP* comes with a smoothing method. The relative frequency of underivable trees in HC is denoted by p_{unkn}. This value is discounted from all the fragments in the grammar, and distributed over all the depth one fragments in the entire treebank ($HC \cup EC$).

Consistency and bias DOP* was introduced as the first consistent (non-trivial) DOP-estimator. Zollmann provides a consistency proof in (Zollmann and Sima'an, 2005). On the other hand, DOP* is biased, but Zollmann shows how bias actually arises from generalization: no non-overfitting DOP estimator could be unbiased. Bias is therefore not problematic but a desirable property of an estimator.

The consistency of DOP* is fundamentally tied to the extraction of fragments in the shortest derivations. As the treebank size increases, the expected loss of the estimate will diminish, as described above. One of the goals of this project is to describe what influence this has on the distribution of weight over the fragments in the grammar.

In (Zuidema, 2006) it is argued that there is a problem with the consistency proof given for DOP*, as well as the non-consistency proof for other DOP-estimators by (Johnson, 2002). Zuidema points out that these proofs use a frequency-distribution test, whereas for DOP a weight-distribution test would be more appropriate.

4 Comparison

DOP* and Double-DOP differ both in the set of fragments they extract and their estimation of the weights. To investigate the exact differences, we will view both steps separately.

Extraction Double-DOP uses a tree kernel approach to find the maximal overlapping fragments of pairs of trees, which are added to the symbolic grammar. We will call this the *maximal-overlap* method. DOP* iteratively finds the shortest derivation of one tree given all the fragments of a set of trees, hereafter the *shortest-derivation* method.

It is easy to see that the *shortest-derivation* extraction in itself does not depend on the corpus split: we can also find the shortest possible derivation using fragments from all the other trees. Likewise Double-DOP could be implemented using a split, comparing all trees in the HC to all trees in

the EC. We will refer to these methods as *full* and *split* estimation.

Estimation Both approaches use the relative frequencies of the fragments for the weights:

$$p(f) = \frac{count(f)}{\sum_{f' \in F_{root}(f)} count(f')} \quad (1)$$

However, in the Double-DOP case these values refer to exact counts of the fragments in the treebank, whereas in DOP* they refer to occurrence of fragments in shortest derivations.

Double-DOP determines the weights of the fragments in the symbolic grammar in a separate run over the treebank, to obtain exact counts. We use the relative frequency estimate to assign weights to the fragments. DOP* on the other hand counts the occurrence in shortest derivations of the fragments, and normalizes relative to counts of fragments with the same root.

To maximize coverage of the grammar, both Double-DOP and DOP* apply smoothing that slightly alters the weights. Next to extracting maximally overlapping fragments, Double-DOP extracts all CFG rules and estimates their weight. DOP* is smoothed by calculating the weight of the unparsed sentences, and distributes this over the CFG rules.

Example This example clarifies how the grammars that result from Double-DOP and DOP* can actually differ. Recall our toy treebank from figure 1 and the fragments in figure 2. Applying the maximal overlap extraction and shortest derivation extraction with full estimation to this treebank, yields the weights in table 1.

Note the remarkable differences in the weight distributions. For example, f_1 gets a weight of 0.5 in the maximal overlap approach, and zero in the shortest derivation case. Of course, the sparsity of the data contributes much to these extreme variations. However, the observed differences encourage us to investigate these two approaches into more depth.

4.1 Experiments

We compare the maximal overlap and shortest derivation extraction by using either a split or the whole set of trees for both estimators. We will plot

---

[2]For this dataset, two shortest derivations exist for each tree. We refer to them with the following variables: 1a = f5, f6; 1b = f2, f10; 2a = f2, f8; 2b = f3, f6; 3a = f4, f8; 3b = f3, f7; 4a = f5, f7; 4b = f4, f9

	Maximal weight overlap		Shortest weight deriv.[2]	
f1	(t1,t3),(t2,t4)	4/12	-	0
f2	(t1,t2)	2/12	1b, 2a	1/4
f3	(t2,t3)	2/12	2b, 3b	1/4
f4	(t3,t4)	2/12	3a, 4b	1/4
f5	(t1,t4)	2/12	1a, 4a	1/4
f6	(t1,t3),(t1,t4), (t2,t3),(t2,t4)	4/6	1a, 2b	1/2
f7	-	0	3b, 4a	1/2
f8	CFG rule	2/6	-	0
f9	(t2,t3),(t2,t4), (t3,t4)	4/6	2a, 3a	1/2
f10	-	0	1b, 4b	1/2
f11	CFG rule	2/6	-	0
f12	CFG rule	2/2	-	0
f13	CFG rule	2/2	-	0

Table 1: The weights assignment according to both extraction methods in a full estimation manner

the fragments according to the weights assigned by the estimators, such that the differences can stand out. In the same way, we compare the split and full estimation for the same estimator.

Furthermore, we can compare the grammars by having them parse a test set and determine their performance, e.g. the F1-score for correctly predicted parses.

Data We use the *Wall Street Journal* (WSJ) section of the Penn Treebank for our experiments. DOP* has only been applied to the Dutch OVIS corpus in (Zollmann and Sima'an, 2005), which contains relatively small and (therefore) easy sentences. Therefore we are curious about its performance on the WSJ.

The corpus was preprocessed by removing functions and binarizing the trees by Markovization (h=1, v=1).

Algorithm First, we find the maximally overlapping fragments of all trees in the corpus, which corresponds to Double-DOP, and we call the *Full Maximum Overlap* approach. Then, we randomly split the corpus ten times in two equally-sized parts, called EC and HC. For each split, build a DOP-reduction grammar from EC and use it to find the shortest derivations (*Split Shortest Derivations*) of the trees in HC, which corresponds to DOP*. Additionally, we find the maximally over-

lapping fragments and estimate their weights from the EC. We'll call this *Split Maximum Overlap*.

The results for the different splits were interpolated and the resulting grammars were smoothed as described above. For finding the weight of unparsable sentences in the DOP* estimation, we needed to find the trees that had not been derived in any split. This was done by comparing each set of underived trees with the HC sets of every split.

Estimation and parsing were done with the framework [3].

Parsing The input for the parser consisted of sentences of word and POS tag pairs. The parser matches fragments to the whole pair when the word is known, but only uses the POS tag when the word is unknown.
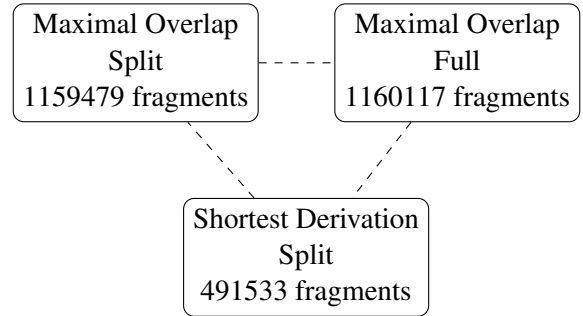
5 Results and analysis

Figure 3: The grammars and their size

Figure 3 summarizes the three grammars we investigate. We have not been able to provide an efficient implementation of shortest derivation extraction with full estimation. The split estimation was done by interpolating the estimates produced by ten random (equal) splits. All grammars have been smoothed with PCFG rules to maximize coverage. In the maximal overlap approach, this is done internally as defined in Double-DOP. In the shortest derivation apporach, we redistribute a proportion p_{unkn} of the probability mass over the PCFG grammar. p_{unkn} was found to be 1.41×10^{-3} for our dataset.

5.1 Parsing performance

Table 2 shows the parsing performance for the three grammars we constructed. Note that the POS-tags were passed to the parser in all cases, so tagging accuracy was 100% and is omitted from

	Maximal Overlap Full	Maximal Overlap Split	Shortest Derivation Split
labeled recall	86.17	85.11	79.20
labeled precision	86.05	85.50	79.32
labeled f-measure	86.11	85.31	79.26
exact match	28.32	25.87	16.52

Table 2: Results for 1229 sentences of length≤ 40

this table. Note that we did only one run on a pre-defined train/ test split.

Both Maximal Overlap grammars perform much better than the Shortest Derivation one, in spite of the latter being consistent. This might well be related to the size of the grammars: the Shortest Derivation grammar has less than half the number of fragments the other two grammars have.

A second explanation might be the smoothing we conducted. Recall that the coverage of the Maximal Overlap approach is catered in a rather natural way, by extending the symbolic grammar with all PCFG rules from the treebank and treating them like the other fragments in the estimation. In the case of Shortest Derivation however, the coverage was a bit more artificial. p_{unkn} was computed over all folds and used to redistribute weights over a classical PCFG constructed from the entire treebank.

Furthermore, we observe that the performance of Maximal Overlap with Full estimation performs slightly better than the Split estimation. The latter was expected to prevent overfitting, but apparently was not helpful in this case. As the weights of these two grammars do not diverge much, it would be recommended to do several runs (with different splits for train and test set). This was however not feasible for us, given the limited amount of time for this project.

5.2 Pairwise comparison of the grammars

In each plot, two grammars are compared to each other. The fragments are presented in a scatter plot, with the weights assigned by the two grammars along the axes. The weights are best visualized on a logarithmic scale. However, it is also informative to see those fragments with value zero. Therefore, the first interval ($[0, 10^{-6}]$) is linear, while the rest of the plot is logarithmic. The difference between grammars is represented by the distance of the points to the *identity line* $x = y$. The color corresponds to some feature, e.g. the depth of the fragment. The color mapping is also logarithmic.

Depth of the fragments In figure 4, the color of the points in the scatter plot refer to the depth of the fragments. Depth is a common measure of fragment size. Johnson shows in (2002) that the original DOP1 had a bias towards larger fragments.

Plot 4c illustrates the effect of the split estimation as compared to full estimation. We see a remarkable, almost linear separation of fragments with larger depth (lighter color) below the identity line, and fragments with smaller depth above it. This indicates that the split estimation tends to reduce the bias towards larger fragments.

In plots 4a and 4b, the shortest derivation extraction is compared to both maximal overlap grammars. Although the correlation between weight assignment and depth is less evident, the area above the identity line gets fragments with a lighter color. This reveal that shortest derivation extraction gives a higher weight to large fragments than maximal overlap.

Width of the fragments The width of the fragments is another measure of their size. It is defined as the number of substitution sites plus the number of terminals. As an example, in figure 5a both extraction methods are compared with split estimation. The dark peak on the right corresponds to the smoothing: PCFG fragments have low width by definition. In figure 5b only fragments of depth 2 and deeper are shown, and the dark peak is gone.

The way the fragments with certain width are distributed does not differ much from the depth, of course width and depth are strongly correlated themselves. We have also looked at the number of substitution sites and the number of terminals of isolation, but this did not reveal much more than the total width.
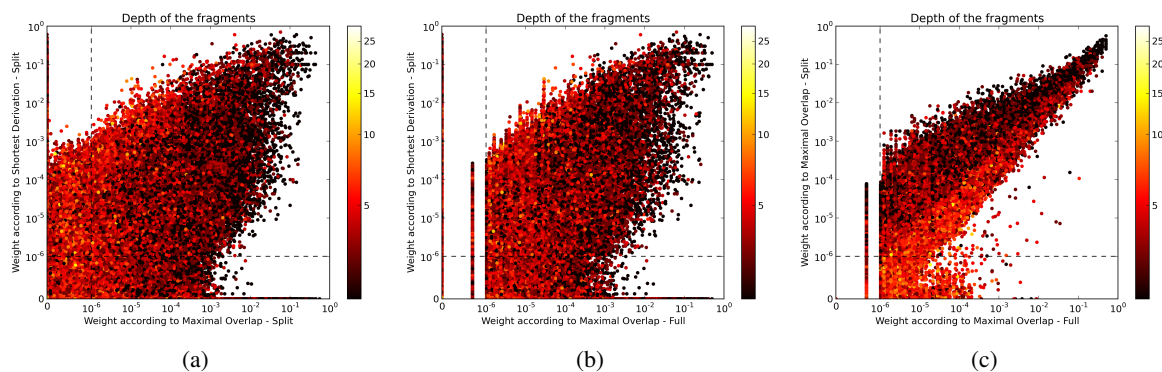
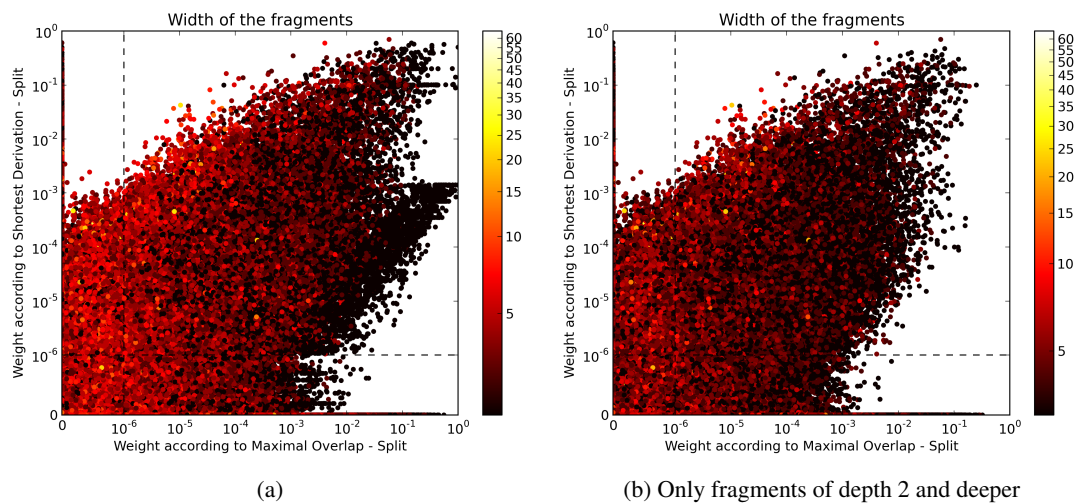Figure 4: Comparing three grammars by depth of the fragments

Figure 5: Comparing shortest derivation to maximal overlap extraction by width of the fragments

6 Conclusion

We have investigated two existing estimators in the Data Oriented Parsing approach to natural language syntax: Double-DOP and DOP*. Our main question concerned the correspondence between theoretical properties and practical performance.

We compared the theoretically attractive shortest-derivation fragment extraction from DOP* with the practical Double-DOP approach. Is proved impossible to match its performance. Because the consistent grammar performed poorer, it seems that consistency might be a less interesting property in practice.

We introduced the corpus splitting from DOP* into the Double-DOP approach. Estimating on a random split did not give us the expected overfitting avoidance. The split does seem to make the grammar less biased towards large fragments.

Concerns While all grammars were trained on the same data set, the grammar from the shortest derivation with split was significantly smaller than the others. This could in itself explain the lower performance of this grammar, instead of the estimation method, because larger tree substitution grammars simply describe more data. It is therefore impossible to be completely certain in blaming the lower performance on the shortest-derivation approach.

Future work Our analysis concerned only two existing DOP estimators: Double-DOP and DOP*. Others have been proposed with one or more of the same desirable properties or comparable performance, of which the comparitive analysis is future work.

Additionally, the fragment weight comparision in this paper is exploratory and rough. It would be very interesting to perform clustering on features such as depth, number of substitution sites and terminals to build a model of the difference in weight distribution.

Finally, it is an important question whether estimator consistency for DOP is a valuable property at all. Behaviour of an estimator in the limit, on finite languages and on artificial constructions is not guaranteed to lead to improved performance on real treebanks.

References

[Johnson2002] Mark Johnson. 2002. The dop estimation method is biased and inconsistent. *Computational Linguistics*, 28(1):71–76.

[Sangati and Zuidema2011] Federico Sangati and Willem Zuidema. 2011. Accurate parsing with compact tree-substitution grammars: Double-dop. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 84–95. Association for Computational Linguistics.

[Scha1990] Remko Scha. 1990. Language theory and language technology; competence and performance. *Computertoepassingen in de Neerlandistiek, Almere: Landelijke Vereniging van Neerlandici (LVVN-jaarboek)*, 11:7–22. Original title: Taaltheorie en taaltechnologie; competence en performance.

[Zollmann and Sima'an2005] Andreas Zollmann and Khalil Sima'an. 2005. A consistent and efficient estimator for data-oriented parsing. *Journal of Automata, Languages and Combinatorics*, 10(2/3):367–388.

[Zuidema2006] Willem Zuidema. 2006. Theoretical evaluation of estimation methods for data-oriented parsing. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, pages 183–186. Association for Computational Linguistics.