Doubling DOP*

A comparison of Double-DOP and DOP*

Benno Kruit
10576223

Sara Veldhoen
10545298

Abstract

This paper investigates two existing estimators in the Data Oriented Parsing approach to natural language syntax. We assess the theoretical and practical differences between these estimators by comparing the grammars they derive.

1 Introduction

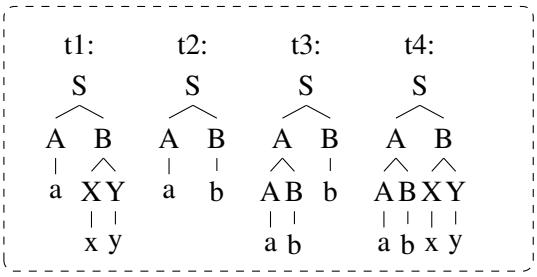


Figure 1: A toy treebank

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.

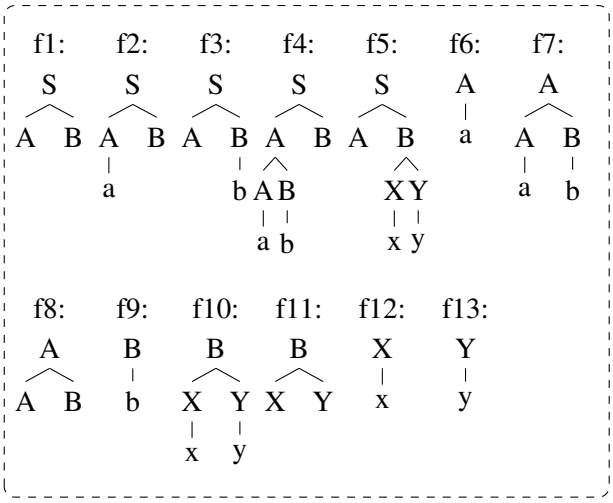


Figure 2: Extracted fragments

Fragments can be combined in a *derivation* to build syntactic structures. A step in a derivation is a composition, denoted by the symbol ∘. For

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

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 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.

1.2 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 that is consistent.

1.3 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. This appears to be far from trivial.

1.4 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. Theoretically, they differ in that DOP*, unlike Double-DOP, has been proven to be consistent (Zollmann and Sima'an, 2005). 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 $\text{EST}(X)$ an estimator trained on a sample. If $\mathcal{M}$ is the set of probability distributions 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 **??** 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 *one vs. rest* 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 in a 1 vs the rest manner 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.

	Maximal overlap	weight	Shortest deriv.[2]	weight
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 methods in a one vs. the rest manner

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 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 one vs. the rest estimation for the same estimator.

Furthermore, we can compare the grammars by having them parse a test set and determine their

[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

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 of all, we find the maximally overlapping fragments of all trees in the corpus, which corresponds to the Double-DOP 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 of the trees in HC, which corresponds to DOP*. Additionally, we find the maximally overlapping fragments and estimate their weights from the EC. We'll call this *Split Double-DOP*.

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.

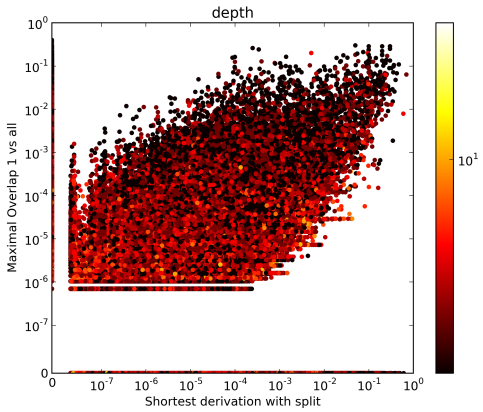
Estimation and parsing were done with the `disco-dop` framework **??**.

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.

Questions Comparing DOP* and Double-DOP: Which estimator gives more weight to large fragments? Is this related to consistency?

Comparing Double-DOP and Split Double-DOP: What influence does a split have on performance and size of a grammar? How do the fragment weight distributions differ?

Comparing Split Double-DOP and DOP*: Assuming that a larger weight in DOP* corresponds to 'usefulness' in parsing, what determines whether a fragment is useful?



5 Results

5.1 Parsing performance

Table2 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 ommitted from this table.

Both Maximal Overlap grammars perform much better than the Shortest Derivation one, in spite of the latter being consistent. One possible explanation, is the smoothing we conducted. Reall 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.

p_{unkn} was found to be 1.41×10^{-3}

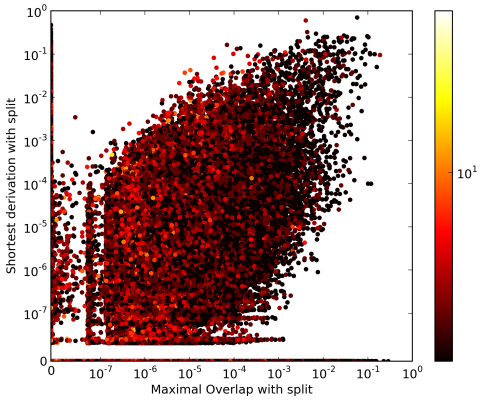
5.2 Pairwise comparison of the grammars

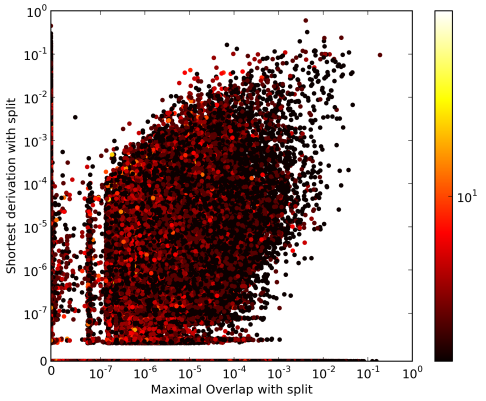
In each plot, two grammars are compared to each other. The fragments are presented in a scatter plot, with the weights assigned by the two grammars along the axes. The weights are best visualized on a logarithmic scale. However, it is also informative to see those fragments with value zero. Therefore,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

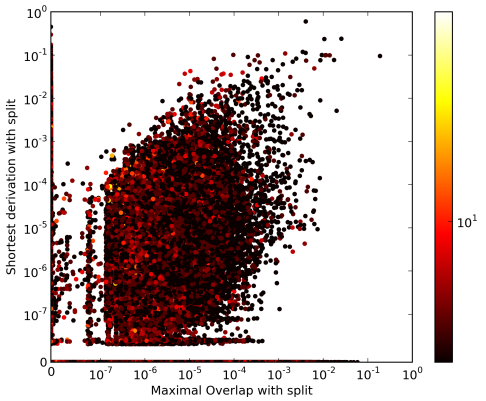
Comparing 'split' to 'one vs. rest' Figure 5.2 illustrates the effect of the split estimation as compared to the one vs. rest.

	Maximal Overlap 1 vs. Rest	Maximal Overlap Split	Shortest Derivation Split
labeled recall	90,97	90.14	84.90
labeled precision	90.25	90.10	84.39
labeled f-measure	90.61	90.12	84.64
exact match	53.27	49.53	39.25
leaf-ancestor	95.09	94.70	92.87

Table 2: Results for 321 sentences of length<16







6 Conclusion

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.