A New DOP Model for Phrase-structure Parsing of Persian Sentences

Zahra Sarabi, Morteza Analouie
(1) Iran University of Science and Technology
(2) Iran University of Science and Technology

Z_sarabi@comp.iust.ac.ir, analoui@iust.ac.ir

ABSTRACT

In this paper we employ a most recent approach to Data Oriented Parsing (DOP), which has named Double-Dop, for Persian sentences. Like other DOP models, Double-Dop parser utilizes syntactic fragments of arbitrary size from a treebank to analyse new sentences, but it extracts a restricted yet representative subset of fragments. It uses only those which are encountered at least twice. The accuracy of Double-DOP is well within the range of state-of-the-art parsers currently used in other NLP-tasks, while offering the additional benefits of a simple generative probability model and an explicit representation of grammatical constructions.
Heretofore there isn't any standard parser for Persian language and this work try to employ Double-Dop Method for parsing Persian sentences.

KEYWORDS: Data Oriented Parsing, Persian Language, Tree Substitution Grammar, Parsing, DOUBLE DOP.

Proceedings of the 10th Workshop on Asian Language Resources, pages 45–54,
COLING 2012, Mumbai, December 2012.

1 Introduction

The Data-Oriented Parsing (DOP) framework, is a famous and wide-coverage parsing method which was first proposed by Scha in 1990(Scha 1990) and formalized by Rens Bod (Bod 1992). Its underlying assumption is that human perception of language based on previous language experiences rather than abstract grammar rules. In the most prominent DOP variants, certain subtrees (called fragments) of variable size, are extracted from the parse trees of the treebank during the training process. These fragments are assigned weights between 0 and 1. Fragments can be recombined to assign parse trees to new sentences. The first implementation of DOP, DOP1 (Bod 1992), and its developed versions(e.g.(Bod 2003)) aimed at extracting all subtrees of all trees in the treebank. The total number of constructions, however, is prohibitively large for non-trivial treebanks: it grows exponentially with the length of the sentences, yielding the astronomically large number of approximately 10^{48} for section 2-21 of the Penn WSJ corpus. Later DOP models have used the Goodman transformation(Goodman 1996; Goodman 2003)to obtain a compact representation of all fragments in the treebank (Bod 2003; Bansal and Klein 2010). The transformation was defined for some versions of DOP to an equivalent PCFG-based model, with the number of rules extracted from each parse tree being linear in the size of the trees. Bod has argued for the Goodman transform as the solution to the computational challenges of DOP (e.g.,(Bod 2003)); it is important to realize, however, that the resulting grammars are still very large: WSJ sections 2-21 yield about7.8×10^6 rules in the basic version of Goodman's transform. Moreover, the transformed grammars differ from untransformed DOP grammars in that larger fragments are no longer explicitly represented. This way, an attractive feature of DOP, viz. the explicit representation of the 'productive units' of language, is lost.
In this paper we use a novel DOP model(Double-DOP) in which we extract a restricted yet representative subset of fragments: those recurring at least twice in the treebank(Sangati and Zuidema 2011). The accuracy of Double-DOP is well within the range of state-of-the-art parsers currently used in other NLP-tasks, while offering the additional benefits of a simple generative probability model and an explicit representation of grammatical constructions. This model reduces the number of extracted fragments from the astronomical 10^{48} to around 10^6.
The rest of the paper is structured as follows. In section 2 we describe Formal Specification of DOP model in general and Double-DOP model in detail, which we will use for parsing. In section 3 we illustrate the Implementation phase and the difficulties of Persian sentences parsing which we are encountered and finally we come to conclusion.

2. Data Oriented Parsing

2.1 Formal Specification of DOP

A DOP grammar can be described as a collection T of fragments. Figure 1 shows an example of four fragments that are extracted from the training parse tree depicted in figure 2, belonging to the PTB[1] training corpus. Fragments are defined in such a way that

[1] Persian Treebank (Per TreeBank) :http://hpsg.fu-berlin.de/~ghayoomi/PTB.html
See also(Ghayoomi 2012)

every node is either a non-terminal leaf (with no more children), or has the exact same children as in the original tree. Since Persian is a right-to-left language, the trees in Figures 1and 2 should be read right-to-left.

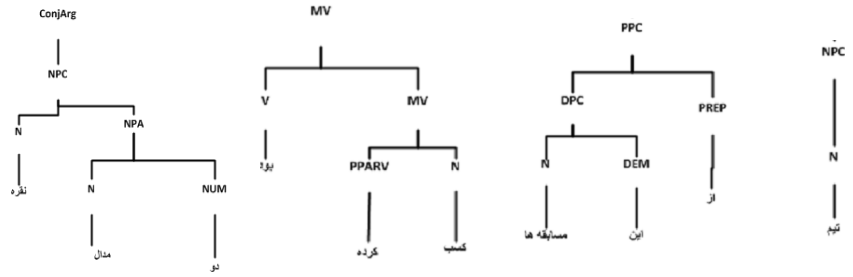

FIGURE 1- Example of elementary trees of depth 4, 3, 3, and 2.

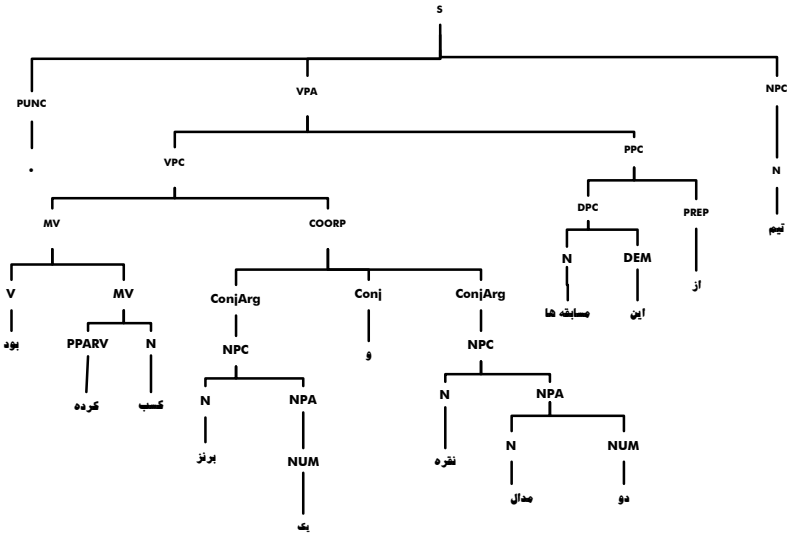

FIGURE 2- Parse tree of the sentence "تیم از این مسابقه‌ها دو مدال نقره و یک برنز کسب کرده بود-".

DOP parses new input by combining treebank subtrees by means of a leftmost node-substitution operation, indicated as º. Two subtrees t and u can be combined by means

of the substitution operation, t°u, if the label on the leftmost nonterminal leaf node of t is identical to the label on the root node of u. The result of this operation is a unified fragment which corresponds to subtree t with the leftmost nonterminal leaf replaced with the entire fragment u. The substitution operation can be applied iteratively since ° is left associative: t ° u ° z = (t ° u) ° z (Bod 1998).
The probability of a parse tree is computed from the occurrence frequencies of the subtrees in the treebank. That is, the probability of a subtree t is taken as the number of occurrences of t in the training set, | t |, divided by the total number of occurrences of all subtrees t' with the same root label as t. Let r(t) return the root label of t:

$$P(t) = \frac{|t|}{\sum_{t':r(t')=r(t)} |t'|} \qquad (1)$$

The probability of a derivation $t_1 \circ \ldots \circ t_n$ is computed by the product of the probabilities of its subtrees t_i:

$$P(t1 \circ \ldots \circ tn) = \prod_i P(t_i) \qquad (2)$$

A same parse tree can be generated by a large number of different derivations, which involve different fragments from the corpus. The probability of a parse tree is the probability that it is produced by any of its derivations. These derivations have their own probability of being generated. Therefore, the probability of a parse tree T is the sum of the probabilities of its distinct derivations D:

$$P(T) = \sum_{D \text{ derives } T} P(D) \qquad (3)$$

A disadvantage of this model is that an extremely large number of subtrees (and derivations) must be taken into account. This leads to exponentially many trees, and thus both exponential time and space requirements. On top of this, the typical optimization criterion, most probable parse, is NP-complete to solve for, leading to an exponentially hard problem of exponential size(Sima'an 1999).The solution has typically been various approximations, such as sampling from the set of all trees, to reduce the size of the grammar, or sampling from the set of all parses – Monte Carlo approximations – to reduce the difficulty of the search. One alternate solution is PCFG-reduction of DOP that generates the same trees with the same probabilities (Goodman 2003).Goodman was able to define a way to convert the DOP grammar in a novel CFG, of which the size increases linearly in the size of the training data. Bod shows that these PCFG-reductions result in a 60 times speedup in processing time w.r.t. DOP1(Bod 2003). However In this case the grammatical constructions are no longer explicitly represented and substantial engineering effort is needed to optimally tune the models and make them efficient.

2.2 Formal Specification of Double-DOP model

The most recent solution to computational challenges of DOP is Double-Dop model which propose a more principled-based approach for explicitly extracting a relatively small but still representative set of fragments from a treebank, i.e., those which are encountered at least twice in the treebank, for which there is evidence about their reusability(Sangati and Zuidema 2011). More precisely the model extracts only the largest shared fragments for all pairs of trees in the treebank. The most important technical contributions of Double-Dop method is: (i) a way to restrict the set of

fragments to only those that occur multiple times in the train set, (ii)a transform-backtransform approach that allows using off-the-shelf PCFG parsing techniques.
The first step to build a DOP model is to define the set of elementary fragments in the model. Although extracting recurring fragments is not trivial, but Sangati in(Sangati, Zuidema et al. 2010)proposed a dynamic programming algorithm. The algorithm iterates over every pair of trees in the treebank and looks for common maximal fragments. All subtrees of these extracted fragments necessarily also occur at least twice, but they are only explicitly represented in our extracted set if they happen to form a largest shared fragment from another pair of trees. Figure 3 shows an example of a pair of trees <α,β>, being compared. All the non-terminal nodes of the two trees are indexed following a depth-first ordering. The algorithm builds a chart M with one column for every indexed non-terminal node α_i in α, and one row for every indexed non-terminal node β_j in β. Each cell M<i , j>, identifies a set of indices corresponding to the largest fragment in common between the two trees starting from α_i andβ_j . This set is empty if α_i and β_j differ in their labels, or they do not have the same list of child nodes. Otherwise (if both the labels and the lists of children match) the set is computed recursively as follows:

$$M < i,j > = \{\alpha_i\} \cup \left(\cup_{c=\{1,2,\ldots,|ch(\alpha)|\}} M < ch(\alpha_i, c), ch(\beta_j, c) >\right) \qquad (4)$$

Where ch(α) returns the indices of α's children, and ch(α, c) the index of its c^{th} child.

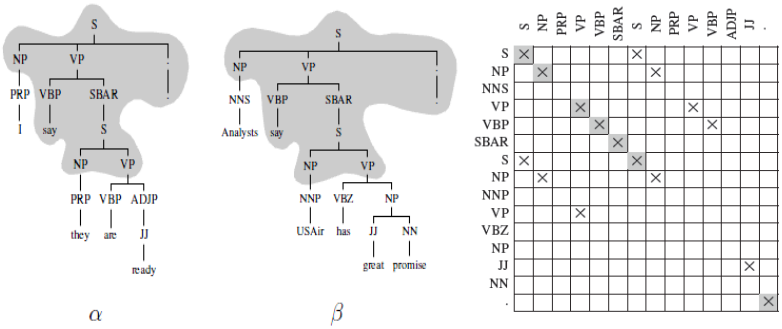

FIGURE 3: Left: example of two trees sharing a single maximum fragment, highlighted in the two trees. Right: the chart used in the algorithm to extract all maximum fragments shared between the two trees(Sangati and Zuidema 2011)

The number of recurring fragments in this grammar, extracted from the training sections of the Penn WSJ treebank, is around 1 million, and thus is significantly lower than previous work extracting explicit fragments (e.g.,(Bod 2001) used more than 5 million fragments up to depth 14).

2.2.1 Parsing with Double-DOP

It is possible to define a simple transform of probabilistic fragment grammar, such that off-the shelf parsers can be used. In order to perform the PTSG/PCFG conversion, every fragment in the grammar must be mapped to a CFG rule which will keep the same probability as the original fragment. The corresponding rule will have as the left hand side the root of the fragment and as the right hand side its yield, i.e., a sequence of terminals and nonterminals (substitution sites)(Sangati and Zuidema 2011).

2.2.2 Inducing probability distributions

Relative Frequency Estimate (RFE): The simplest way to assign probabilities to fragments is to make them proportional to their counts in the training set.

$$P_{RFE}(f) = \frac{count(f)}{\sum_{f' \in F_{root(f)}} count(f')} \qquad (5)$$

3 Implementation

In this section we want to illustrate the implementation phase in which we employ Double-DOP model to Persian language. For this purpose we have some challenges and difficulties in processing Persian language. In continue we first analyse these challenges and after that explain the detail of our implementation.

3.1 challenges of Persian language processing

Persian language is the formal language of Iran and some neighbourhood countries like Afghanistan and Tajikistan and more than one hundred millions of people speak with this language. Furthermore many written resources like online pages, news, books and translated books exist for this language. So preparing tools and processing resources, which is used in linguistics applications, should take into consideration.

Nowadays the importance of availability or development of annotated data becomes crucial to feed linguistic investigation and also to use data driven approaches in human language technologies. Some languages like English and German are given a great amount of consideration which results to various types of data sources; while some other languages like Persian are less developed in terms of availability of annotated data.
A necessary condition for testing a DOP model is the availability of annotated language corpora. Therefore one of the most important challenges in parsing Persian sentences with data oriented approaches is the lack of linguistic annotated resources like a standard treebank. Until very recently, there wasn't any standard public available treebank for Persian language, but fortunately some efforts being performed by Ghayoomi in Freie Universität Berlin, Germany, who is developing the Persian treebank and make it publicly available as the only readily available corpora consisted of syntactically labeled phrase-structure trees[1](Ghayoomi 2012). This Persian treebank

[1]The developed Persian treebank is accessible from this link:
http://hpsg.fu-berlin.de/_ghayoomi/PTB.html

(PTB) currently contains 1012sentences with the total size of 27731 word tokens.Of course in order to employ DOP model, we would need a larger treebank and it seems that the results of our work would become poor relative to the same methodology applied to English treebanks, But that does not detract from the value and efficiency of this approach.

3.2 Experimental setup

In order to build and test our Double-DOP model we employ the Persian treebank-PTB(Ghayoomi 2012).

Preparing the treebank: We start with some pre-processing of the treebank, following standard parsing methods. We named this step preparing the treebank and performed following tasks in this step in order: first we divide the treebank into two sections of train and test and assign about 1000 sentences for train and 200 sentences for test. Next we have removed all empty nodes, functional tags, semantic tags, traces and also punctuations from the treebank. After that we apply binarization procedure to training pars trees of treebank. Binarization is particularly important for generative models like DOP and PCFGs, and was essential for the success of the model, where all the children of an internal node are produced at once. Binarization, in fact, provides a way to generalize flat rules, by splitting it in multiple generation steps. Double-DOP model creators claim that, on an unbinarized treebank, the model performed rather poorly because of the abundance of flat rules. However, our current model uses a strict left binarization as in(Matsuzaki, Miyao et al. 2005).

Executing Fragment seeker algorithm (Sangati, Zuidema et al. 2010):

In this step we explicitly extract a subset of fragments from the training treebank. As explained in section 2, we extract only those fragments that occur two or more times in the treebank. Details of this algorithm illustrated in (Sangati, 2010) and they implemented software for extracting recurring fragments named Fragment Seeker[1]. Although this software was available and free but it didn't work well for Persian language and therefore we implemented the algorithm again for Persian language. Running this algorithm is the most time-consuming step (around 160 CPU hours).Parse trees in the training corpus are not necessarily covered entirely by recurring fragments; to ensure better coverage, we also extract all PCFG-productions not included in the set of recurring fragments.

Parsing

We convert our PTSG into a PCFG (section 2.2.1) and use Bitpar[2] parser (Schmid 2004)to parse the 200 sentences in the test set. Bitpar is a parser that implements the CYK algorithm. It can parse sentences starting from a set of CFG rules. So it's not specific to a certain language.

[1]The implemented software for extracting recurring fragments (Fragment Seeker) is available at http://staff.science.uva.nl/~fsangati/

[2] http://www.ims.uni-stuttgart.de/tcl/SOFTWARE/BitPar.html

Results

In this work in order to clarify to what extent Double DOP parsing improves parsing results, we first train a PCFG parser for Persian Treebank as a baseline and achieve only 36% in F-score. We have also compared our best Double-DOP base model with some previous DOP models like DOP-Goodman and DOP $_{h=2}$.
Table 1 shows a summary of the parsing results of our system on Double-DOP model , which achieves 59% in labeled F-score. Further investigations suggest that the majority of parsing errors are due to crossing brackets and wrongly labeled constituents are in fact a minor source of error.

Parsing Model	Result
PCFG(H=1 , P=1)	33%
Stanford PCFG Parser (H=1 P=1)	39%
DOP1	48%
DOP Goodman	50%
DOP $_{h=2}$	49%
Double-DOP	59%

Table 1- Summary of the Parsing evaluation results

Conclusion

We have presented a simplified DOP formalism, named Double-DOP for learning the constituency structure of Persian sentences. Double-DOP is a most recent DOP approach for parsing, which uses all constructions recurring at least twice in a treebank. Other DOP models have many shortcomings so that DOP parsers are almost never used in other NLP tasks. The most important reasons for this are probably the computational inefficiency of many instances of the approach, the lack of downloadable software and the difficulties with replicating some of the key results. Fortunately Double-DOP model untie these difficulties by: the efficient algorithm for identifying the recurrent fragments in a treebank runs in polynomial time. The transformation to PCFGs that the model defines allows us to use a standard PCFG parser, while retaining the benefit of explicitly representing larger fragments.
The results of our work are poor relative to the same methodology applied to English treebanks. One of the main reasons is certainly the smaller size of the training corpus used in the current shared task. As in other types of exemplar-based learning techniques, DOP models require a large amount of data in order to achieve high accuracy.
We try to improve our results in further investigations.

References

Bansal, M. and D. Klein (2010). "Simple, accurate parsing with an all-fragments grammar." Proceedings of the 48th Annual Meeting of the ACL: pages 1098–1107.

Bod, R. (1992). "A computational model of language performance: Data oriented parsing." Proceedings COLING'92 (Nantes, France): pp: 855–859.

Bod, R. (1998). "Beyond Grammar: An Experience-Based Theory of Language." Stanford, CSLI Publications.

Bod, R. (2001). "What is the Minimal Set of Subtrees that Achieves Maximal Parse Accuracy?" Proceedings ACL'2001, Toulouse, France.

Bod, R. (2003). "An efficient implementation of a new DOP model." Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics , Volume 1(EACL'03): PP:19–26.

Ghayoomi, M. (2012). "Bootstrapping the Development of an HPSG-based Treebank for Persian." Linguistic Issues in Language Technology: LiLT.

Goodman, J. (1996). "Efficient algorithms for parsing the DOP model." In Proceedings of the Conference on Empirical Methods in Natural Language Processing,: pages 143–152.

Goodman, J. (2003). "Efficient parsing of DOP with PCFG-reductions." In Bod et al. (2003).

Matsuzaki, T., Y. Miyao, et al. (2005). "Probabilistic CFG with latent annotations." Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pages 75–82, Morristown, NJ, USA.

Sangati, F. and W. Zuidema (2011). "A Recurring Fragment Model for Accurate Parsing: Double-DOP." In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing,: pages 84–95.

Sangati, F., W. Zuidema, et al. (2010). "Efficiently Extract Recurring Tree Fragments from Large Treebanks." Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10),.

Scha, R. (1990). " Taaltheorie en taaltechnologie; competence en performance." Q. A. M. de Kort and G. L. J. Leerdam, editors, Computertoepassingen in de Neerlandistiek, LVVNjaarboek: 7-22.

Schmid, H. (2004). "Efficient parsing of highly ambiguous context-free grammars with bit vectors." Proceedings of COLING 2004, pp. 162{168. Geneva, Switzerland.

Sima'an, K. (1999). "Learning Efficient Disambiguation." PhD thesis, University of Amsterdam, The Netherlands.