Unsupervised Dependency Parsing using Reducibility and Fertility features*

David Mareček and Zdeněk Žabokrtský
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, Prague
{marecek,zabokrtsky}@ufal.mff.cuni.cz

Abstract

This paper describes a system for unsupervised dependency parsing based on Gibbs sampling algorithm. The novel approach introduces a fertility model and reducibility model, which assumes that dependent words can be removed from a sentence without violating its syntactic correctness.

1 Introduction

One of the traditional linguistic criteria for recognizing dependency relations (including their head-dependent orientation) is that stepwise deletion of dependent elements within a sentence preserves its syntactic correctness (Lopatková et al., 2005; Kübler et al., 2009; Gerdes and Kahane, 2011).[1] If a word can be removed from a sentence without damaging it, then it is likely to be dependent on some other (still present) word.

Our approach allows to utilize information from very large corpora. While the computationally demanding sampling procedure can be applied only on limited data, the unrepeated precomputation of statistics for reducibility estimates can easily exploit much larger data.

2 Precomputing PoS tag reducibility scores

We call a word (or a sequence of words) in a sentence *reducible*, if the sentence after removing the word remains grammatically correct. Although we cannot automatically recognize grammaticality of such newly created sentence, we can search for it in a large corpus. If we find it, we assume the word was reducible in the original sentence. It is obvious that the number of such reducible sequences of words found in a corpus is relatively low. However, it is sufficient for determining reducibility scores at least for individual types of words (part-of-speech tags).[2]

Assume a PoS n-gram $g = [t_1, \ldots, t_n]$. We go through the corpus and search for all its occurrences. For each such occurrence, we remove the respective words from the current sentence and check in the corpus whether the rest of the sentence occurs at least once elsewhere in the corpus.[3] If so, then such occurrence of PoS n-gram is reducible, otherwise it is not. We denote the number of such reducible occurrences of PoS n-gram g by $r(g)$. The number of all its occurrences is $c(g)$. The relative reducibility

* This research was supported by the grants GA201/09/H057 (Res Informatica), MSM0021620838, GAUK 116310, and by the European Commission's 7th Framework Program (FP7) under grant agreement n° 247762 (FAUST).

[1] Of course, all the above works had to respond to the notorious fact that there are many language phenomena precluding the ideal (word by word) sentence reducibility (e.g. in the case of prepositional groups, or in the case of subjects in English finite clauses). However, we borrow only the very core of the reducibility idea.

[2] Although we search for reducible sequences of word forms in the corpus, we compute reducibility scores for sequences of part-of-speech tags. This requires to have the corpus morphologically disambiguated.

[3] We do not take into account sentences with less then 10 words, because they could be nominal (without any verb) and might influence the reducibility scores of verbs.

NAACL-HLT Workshop on the Induction of Linguistic Structure, pages 84–89,
Montréal, Canada, June 3-8, 2012.

unigrams	R	bigrams	R	trigrams	R
VB	0.04	VBN IN	0.00	IN DT JJ	0.00
TO	0.07	IN DT	0.02	JJ NN IN	0.00
IN	0.11	NN IN	0.04	NN IN NNP	0.00
VBD	0.12	NNS IN	0.05	VBN IN DT	0.00
CC	0.13	JJ NNS	0.07	JJ NN .	0.00
VBZ	0.16	NN .	0.08	DT JJ NN	0.04
NN	0.22	DT NNP	0.09	DT NNP NNP	0.05
VBN	0.24	DT NN	0.09	NNS IN DT	0.14
.	0.32	NN ,	0.11	NNP NNP .	0.15
NNS	0.38	DT JJ	0.13	NN IN DT	0.23
DT	0.43	JJ NN	0.14	NNP NNP ,	0.46
NNP	0.78	NNP .	0.15	IN DT NNP	0.55
JJ	0.84	NN NN	0.22	DT NN IN	0.59
RB	2.07	IN NN	0.67	NNP NNP NNP	0.64
,	3.77	NNP NNP	0.76	IN DT NN	0.80
CD	55.6	IN NNP	1.81	IN NNP NNP	4.27

Table 1: Reducibility scores of the most frequent PoS tag English n-grams.

$R(g)$ of a PoS n-gram g is then computed as

$$R(g) = \frac{1}{N}\frac{r(g)+\sigma_1}{c(g)+\sigma_2}, \quad (1)$$

where the normalization constant N, which expresses relative reducibility over all the PoS n-grams (denoted by G), causes the scores are concentrated around the value 1.

$$N = \frac{\sum_{g\in G}(r(g)+\sigma_1)}{\sum_{g\in G}(c(g)+\sigma_2)} \quad (2)$$

Smoothing constants σ_1 and σ_2, which prevent reducibility scores from being equal to zero, are set as follows: [4]

$$\sigma_1 = \frac{\sum_{g\in G} r(g)}{\sum_{g\in G} c(g)}, \qquad \sigma_2 = 1 \quad (3)$$

Table 1 shows reducibility scores of the most frequent English PoS n-grams. If we consider only unigrams, we can see that the scores for verbs are often among the lowest. Verbs are followed by prepositions and nouns, and the scores for adjectives and adverbs are very high for all three examined languages. That is desired, because the reducible unigrams will more likely become leaves in dependency trees.

3 Dependency Models

We introduce a new generative model that is different from the widely used Dependency Model with Valence (DMV).[5] Our generative model introduces *fertility* of a node. For a given head, we first generate the number of its left and right children (*fertility model*) and then we fill these positions by generating its individual dependents (*edge model*). If a zero fertility is generated, the head becomes a leaf.

Besides the fertility model and the edge model, we use two more models (*subtree model* and *distance model*), which force the generated trees to have more desired shape.

3.1 Fertility model

We express a fertility of a node by a pair of numbers: the number of its left dependents and the number of its right dependents.[6] Fertility is conditioned by part-of-speech tag of the node and it is computed following the Chinese restaurant process.[7] The formula for computing probability of fertility f_i of a word on the position i in the corpus is as follows:

$$P_f(f_i|t_i) = \frac{c^{-i}(\text{“}t_i, f_i\text{”}) + \alpha P_0(f_i)}{c^{-i}(\text{“}t_i\text{”}) + \alpha}, \quad (4)$$

where t_i is part-of-speech tag of the word on the position i, $c^{-i}(\text{“}t_i, f_i\text{”})$ stands for the count of words with PoS tag t_i and fertility f_i in the history, and P_0 is a prior probability for the given fertility which depends on the total number of node dependents denoted by $|f_i|$ (the sum of numbers of left and right dependents):

$$P_0(f_i) = \frac{1}{2^{|f_i|+1}} \quad (5)$$

This prior probability has a nice property: for a given number of nodes, the product of fertility probabilities over all the nodes is equal for all possible dependency trees. This ensures balance of this model during inference.

[4] This setting causes that even if a given PoS n-gram is not reducible anywhere in the corpus, its reducibility score is $1/(c(g)+1)$.

[5] In DMV (Klein and Manning, 2004) and in the extended model EVG (Headden III et al., 2009), there is a *STOP* sign indicating that no more dependents in a given direction will be generated. Given a certain head, all its dependents in left direction are generated first, then the *STOP* sign in that direction, then all its right dependents and then *STOP* in the other direction. This process continues recursively for all generated dependents.

[6] For example, fertility “1-3” means that the node has one left and three right dependents, fertility “0-0” indicates that it is a leaf.

[7] If a specific fertility has been frequent for a given PoS tag in the past, it is more likely to be generated again.

Besides the basic fertility model, we introduce also an extended fertility model, which uses frequency of a given word form for generating number of children. We assume that the most frequent words are mostly function words (e.g. determiners, prepositions, auxiliary verbs, conjunctions). Such words tend to have a stable number of children, for example (i) some function words are exclusively leaves, (ii) prepositions have just one child, and (iii) attachment of auxiliary verbs depends on the annotation style, but number of their children is also not very variable. The higher the frequency of a word form, the higher probability mass is concentrated on one specific number of children and the lower Dirichlet hyperparameter α in Equation 4 is needed. The extended fertility is described by equation

$$P'_f(f_i|t_i, w_i) = \frac{c^{-i}(\text{“}t_i, f_i\text{”}) + \frac{\alpha_e}{F(w_i)} P_0(f_i)}{c^{-i}(\text{“}t_i\text{”}) + \frac{\alpha_e}{F(w_i)}}, \quad (6)$$

where $F(w_i)$ is a frequency of the word w_i, which is computed as a number of words w_i in our corpus divided by number of all words.

3.2 Edge model

After the fertility (number of left and right dependents) is generated, the individual slots are filled using the *edge model*. A part-of-speech tag of each dependent is conditioned by part-of-speech tag of the head and the edge direction (position of the dependent related to the head).[8]

Similarly as for the fertility model, we employ Chinese restaurant process to assign probabilities of individual dependent.

$$P_e(t_j|t_i, d_j) = \frac{c^{-i}(\text{“}t_i, t_j, d_j\text{”}) + \beta}{c^{-i}(\text{“}t_i, d_j\text{”}) + \beta|T|}, \quad (7)$$

where t_i and t_j are the part-of-speech tags of the head and the generated dependent respectively; d_j is a direction of edge between the words i and j, which can have two values: *left* and *right*. $c^{-i}(\text{“}t_i, t_j, d_j\text{”})$ stands for the count of edges $t_i \leftarrow t_j$ with the direction d_j in the history, $|T|$ is a number of unique tags in the corpus and β is a Dirichlet hyperparameter.

[8]For the edge model purposes, the PoS tag of the technical root is set to '<root>' and it is in the zero-th position in the sentence, so the head word of the sentence is always its right dependent.

3.3 Distance model

Distance model is an auxiliary model that prevents the resulting trees from being too flat.[9] This simple model says that shorter edges are more probable than longer ones. We define probability of a distance between a word and its parent as its inverse value,[10] which is then normalized by the normalization constant ϵ_d.

$$P_d(i, j) = \frac{1}{\epsilon_d} \left(\frac{1}{|i - j|} \right)^{\gamma} \quad (8)$$

The hyperparameter γ determines the weight of this model.

3.4 Subtree model

The subtree model uses the reducibility measure. It plays an important role since it forces the reducible words to be leaves and reducible n-grams to be subtrees. Words with low reducibility are forced towards the root of the tree. We define $desc(i)$ as a sequence of tags $[t_l, \ldots, t_r]$ that corresponds to all the descendants of the word w_i including w_i, i.e. the whole subtree of w_i. The probability of such subtree is proportional to its reducibility $R(desc(i))$. The hyperparameter δ determines the weight of the model; ϵ_s is a normalization constant.

$$P_s(i) = \frac{1}{\epsilon_s} R(desc(i))^{\delta} \quad (9)$$

3.5 Probability of the whole treebank

We want to maximize the probability of the whole generated treebank, which is computed as follows:

$$P_{treebank} = \prod_{i=1}^{n} (P'_f(f_i|t_i, w_i) \quad (10)$$

$$P_e(t_i|t_{\pi(i)}, d_i) \quad (11)$$

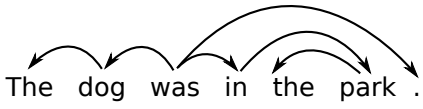
$$P_d(i, \pi(i)) \quad (12)$$

$$P_s(i)), \quad (13)$$

where $\pi(i)$ denotes the parent of the word on the position i. We multiply the probabilities of fertility, edge, distance from parent, and subtree over all

[9]Ideally, the distance model would not be needed, but experiments showed that it helps to infer better trees.

[10]Distance between any word and the technical root of the dependency tree was set to 10. Since each technical root has only one dependent, this value does not affect the model.



(((The) dog) was (in ((the) park)) (.))

Figure 1: Arrow and bracketing notation of a projective dependency tree.

words (nodes) in the corpus. The extended fertility model P'_f can be substituted by its basic variant P_f.

4 Sampling algorithm

For stochastic searching for the most probable dependency trees, we employ Gibbs sampling, a standard Markov Chain Monte Carlo technique (Gilks et al., 1996). In each iteration, we loop over all words in the corpus in a random order and change the dependencies in their neighborhood (a small change described in Section 4.2). In the end, "average" trees based on the whole sampling are built.

4.1 Initialization

Before the sampling starts, we initialize the projective trees randomly in a way that for each sentence, we choose randomly one word as the head and attach all other words to it.[11]

4.2 Small Change Operator

We use the bracketing notation for illustrating the small change operator. Each projective dependency tree consisting of n words can be expressed by n pairs of brackets. Each bracket pair belongs to one node and delimits its descendants from the rest of the sentence. Furthermore, each bracketed segment contains just one word that is not embedded deeper; this node is the segment head. An example of this notation is in Figure 1.

The small change is then very simple. We remove one pair of brackets and add another, so that the conditions defined above are not violated. An example of such change is in Figure 2.

From the perspective of dependency structures, the small change can be described as follows:

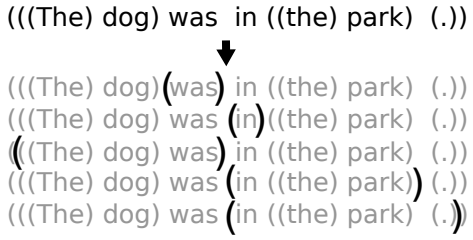


Figure 2: An example of small change in a projective tree. The bracket (in the park) is removed and there are five possibilities how to replace it.

1. Pick a random non-root word w (the word *in* in our example) and find its parent p (the word *was*).

2. Find all other children of w and p (the words *dog*, *park*, and .) and denote this set by C.

3. Choose the new head out of w and p. Mark the new head as g and the second candidate as d. Attach d to g.

4. Select a neighborhood D adjacent to the word d as a continuous subset of C and attach all words from D to d. D may be also empty.

5. Attach the remaining words from C that were not in D to the new head g.

4.3 Building "average" trees

The "burn-in" period is set to 10 iterations. After this period, we begin to count how many times an edge occurs at a particular location in the corpus. This counts are collected over the whole corpus with the collection-rate 0.01.[12]

When the sampling is finished, the final dependency trees are built using such edges that belonged to the most frequent ones during the sampling. We employ the maximum spanning tree (MST) algorithm (Chu and Liu, 1965) to find them.[13] Tree projectivity is not guaranteed by the MST algorithm.

5 Experiments

We evaluated our parser on 10 treebanks included in the WILS shared-task data. Similarly to some previous papers on unsupervised parsing (Gillenwater et al., 2011; Spitkovsky et al., 2011), the tuning experiments were performed on English only. We used

[11] More elaborated methods for generating random trees converges to similar results. Therefore we conclude that the choice of the initialization mechanism is not so important here.

[12] After each small change is made, the edges from the whole corpus are collected with a probability 0.01.

[13] The weights of edges needed in MST algorithm correspond to the number of times they were present during the sampling.

language	tokens (mil.)	language	tokens (mil.)
Arabic	19.7	English	85.0
Basque	14.1	Portuguese	31.7
Czech	20.3	Slovenian	13.7
Danish	15.9	Swedish	19.2
Dutch	27.1		

Table 2: Wikipedia texts statistics

English development data for checking functionality of the individual models and for optimizing hyperparameter values. The best configuration of the parser achieved on English was then used for parsing all other languages. This simulates the situation in which we have only one treebank (English) on which we can tune our parser and we want to parse other languages for which we have no manually annotated treebanks.

5.1 Data

For each experiment, we need two kinds of data: a smaller treebank, which is used for sampling and for evaluation, and a large corpus, from which we compute n-gram reducibility scores. The induction of dependency trees and evaluation is done only on WILS testing data.

For obtaining reducibility scores, we used Wikipedia articles downloaded by Majliš and Žabokrtský (2012). Their statistics across languages are showed in Table 2. To make them useful, the necessary preprocessing steps must have been done. The texts were first automatically segmented and tokenized[14] and then they were part-of-speech tagged by TnT tagger (Brants, 2000), which was trained on the respective WILS training data. The quality of such tagging is not very high, since we do not use any lexicons[15] or pretrained models. However, it is sufficient for obtaining good reducibility scores.

5.2 Setting the hyperparameters

The applicability of individual models and their parameters were tested on English development data and full part-of-speech tags. The four hyperparameters α (fertility model), β (edge model), γ (distance model), and δ (subtree model), were set by a grid search algorithm, which found the following optimal values:

$$\alpha_e = 0.01, \quad \beta = 1, \quad \gamma = 1.5, \quad \delta = 1$$

5.3 Results

The best setting from the experiments on English is now used for evaluating our parser across all WILS treebanks. Besides using the standard part-of-speech tags (POS), the experiments were done also on coarse tags (CPOS) and universal tags (UPOS). From results presented in Table 3, it is obvious that our systems outperforms all the given baselines considering the average scores across all the testing languages. However, it is important to note that we used an additional source of information, namely large unannotated corpora for computing reducibility scores, while the baseline system (Gillenwater et al., 2011) use only the WILS datasets.

6 Conclusions and Future Work

We have described a novel unsupervised dependency parsing system employing new features, such as reducibility or fertility. The reducibility scores are extracted from a large corpus, and the computationally demanding inference is then done on smaller data.

In future work, we would like to estimate the hyperparameters automatically, for example by the Metropolis-Hastings technique (Gilks et al., 1996). Furthermore, we would like to add lexicalized models and automatically induced word classes instead of the PoS tags designed by humans. Finally, we would like to move towards deeper syntactic structures, where the tree would be formed only by content words and the function words would be treated in a different way.

References

Thorsten Brants. 2000. TnT - A Statistical Part-of-Speech Tagger. *Proceedings of the sixth conference on Applied natural language processing*, page 8.

Y. J. Chu and T. H. Liu. 1965. On the Shortest Arborescence of a Directed Graph. *Science Sinica*, 14:1396–1400.

[14]The segmentation to sentences and tokenization was performed using the TectoMT framework (Popel and Žabokrtský, 2010).

[15]Using lexicons or another pretrained models for tagging means using other sources of human annotated data, which is not allowed if we want to compare our results with others.

	baselines			Gillenwater et al. (2011)			our approach		
Treebank	random	left br.	right br.	CPOS	POS	UPOS	CPOS	POS	UPOS
Arabic PADT	37.0	5.2	**58.7**	14.9	14.5	23.4	12.7	57.2	52.0
Basque 3LB	9.8	32.4	22.5	**48.9**	41.8	30.0	21.0	25.5	22.3
Czech PDT	10.8	23.9	29.1	25.6	31.6	27.5	**49.1**	42.9	44.1
Danish CDT	24.7	13.3	47.4	31.8	27.2	26.3	48.4	41.4	**49.7**
Dutch Alpino	17.1	27.9	24.6	23.3	21.2	23.5	28.3	**44.2**	29.9
English Childes	12.8	34.8	22.2	48.8	**50.4**	36.2	54.2	44.2	49.3
English PTB	13.2	31.4	20.4	30.9	34.4	25.9	41.0	**50.3**	37.5
Portuguese Floresta	33.1	25.9	31.1	26.0	31.1	25.5	**50.2**	49.5	29.3
Slovene JOS	9.5	31.1	10.8	36.6	**53.3**	36.6	30.4	40.8	26.7
Swedish Talbanken	23.9	25.8	28.0	27.2	27.2	27.1	48.4	50.6	**52.6**
Average:	19.2	25.2	29.5	31.4	33.3	28.2	38.4	**44.7**	39.3

Table 3: Directed attachment scores on the WILS testing data (all sentences). Punctuation was excluded. Comparison with simple baselines and the given baseline system (Gillenwater et al., 2011).

Kim Gerdes and Sylvain Kahane. 2011. Defining dependencies (and constituents). In *Proceedings of Dependency Linguistics 2011*, Barcelona.

W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. 1996. *Markov chain Monte Carlo in practice*. Interdisciplinary statistics. Chapman & Hall.

Jennifer Gillenwater, Kuzman Ganchev, João Graça, Fernando Pereira, and Ben Taskar. 2011. Posterior Sparsity in Unsupervised Dependency Parsing. *The Journal of Machine Learning Research*, 12:455–490, February.

William P. Headden III, Mark Johnson, and David McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 101–109, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sandra Kübler, Ryan T. McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Markéta Lopatková, Martin Plátek, and Vladislav Kuboň. 2005. Modeling syntax of free word-order languages: Dependency analysis by reduction. In Václav Matoušek, Pavel Mautner, and Tomáš Pavelka, editors, *Lecture Notes in Artificial Intelligence, Proceedings of the 8th International Conference, TSD 2005*, volume 3658 of *Lecture Notes in Computer Science*, pages 140–147, Berlin / Heidelberg. Springer.

Martin Majliš and Zdeněk Žabokrtský. 2012. Language richness of the web. In *Proceedings of LREC2012*, Istanbul, Turkey, May. ELRA, European Language Resources Association. In print.

Martin Popel and Zdeněk Žabokrtský. 2010. TectoMT: modular NLP framework. In *Proceedings of the 7th international conference on Advances in natural language processing*, IceTAL'10, pages 293–304, Berlin, Heidelberg. Springer-Verlag.

Valentin I. Spitkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011. Lateen EM: Unsupervised training with multiple objectives, applied to dependency grammar induction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.