Dealing with Function Words in Unsupervised Dependency Parsing

David Mareček and Zdeněk Žabokrtský

Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
{marecek,zabokrtsky}@ufal.mff.cuni.cz

Abstract. In this paper, we show some properties of function words in dependency trees. Function words are grammatical words, such as articles, prepositions, pronouns, conjunctions, or auxiliary verbs. These words are often short and very frequent in texts and therefore many of them can be easily recognized. We formulate a hypothesis that function words tend to have a fixed number of dependents and we prove this hypothesis on treebanks. Using this hypothesis, we are able to improve unsupervised dependency parsing and outperform previously published state-of-the-art results for many languages.

1 Introduction

Function words (also known as grammatical words) are words which have no or very little lexical meaning, in contrast to content words (lexical words), which have some meaningful content. Function words are articles, pronouns, prepositions, conjunctions, particles or auxiliary verbs and all belong to closed-class words. They are used to express grammatical attributes of content words or grammatical relationships between two or more content words.

In some representations of linguistic structure, function words are treated differently from content words. Tesnière [1] introduces the notion of empty words (function words) and argues that they cannot occupy alone a position in the dependency structure. Functional Generative Description [2] uses so called tectogrammatical representation, in which only content words are represented by nodes and function words are there in forms of their attributes. In Logical Forms [3], some of the function words become labels of edges connecting content words. Another example where the function words are excluded from a sentence structure is the Abstract Meaning Representation [4]. Nevertheless, even within a chosen formalism, the boundaries between function and content words are not entirely straightforward and they are often very fuzzy.

This paper is organized as follows: In Section 2, we describe how to recognize function words in a language, if either only raw texts are available or words in the corpus are labeled with POS tags. Properties of function words in dependency structures are discussed in Section 3. Section 4 explains how these properties can be used in unsupervised dependency parsing task. Experiments are shown in Section 5 and Section 6 concludes.

A. Gelbukh (Ed.): CICLing 2014, Part I, LNCS 8403, pp. 250–261, 2014.

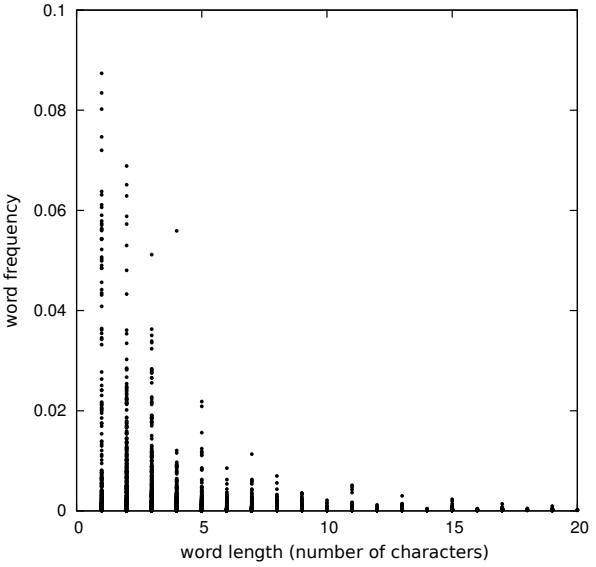

Fig. 1. Relationship between length (number of characters) of a word and its relative frequency. Each point in the graph corresponds to one word type. The words were taken from 20 different treebanks.

2 Recognition of Function Words

We introduce two simple function words properties that can be used for recognizing them in an unsupervised way (with no manual effort). The first one is their frequency – function words are more frequent in language than content words. The second one is their length – function words are relatively short. The well-known relationship between length of a word and its frequency is caused by the *economy* of language [5]; we show this relationship in Figure 1.

It is apparent that the frequent words are mostly short, however it is not true that the short words are frequent. Many short words are abbreviations or numbers, which are definitely not function words. Therefore, we decided to recognize function words using their relative frequency in corpus and not using their length. The relative frequency of word is computed simply by dividing the number of its occurrences by the number of all the words in the corpus:

$$F_W(w) = \frac{count(w)}{total}.$$

The more frequent a word is in a corpus, the more likely it is a function word.

If we have a corpus which is manually or automatically tagged with part-of-speech (POS) tags, we can compute the aggregated word frequency for individual POS tags. The aggregated frequency is computed by averaging the relative frequency of all tokens in the corpus labeled by that POS tag.

$$F_T(t) = \sum_{w;tag(w)=t} \frac{count(w,t)}{count(t)} \cdot \frac{count(w,t)}{total}$$

The formula above expresses the weighted average over the relative frequencies of word types, which is equal to uniform average over the tokens. The more frequent a word is, the higher influence on the aggregated frequency it has. Such weighted average of word frequencies seems to be able to to sufficiently separate the function words POS tags.

3 Properties of Function Words in Dependency Trees

3.1 Types of Function Words

For the purposes of this work, we divide function words into several groups:

- **Function words as content word modifiers** – They express attributes of content words. For example in majority of treebank annotations, *articles* modify nouns. They determine the definiteness and, in some languages, grammatical categories, e.g. the noun case in German. Another example may be *negative particles* that modify verbs and determine their negation. Such function words are mostly annotated as leaves in dependency treebanks.
- **Function words as grammatical relations between content words** – In the second group, there are function words connecting two content words, for example *prepositions* or *postpositions*. They usually connect a noun with another noun or verb and express the type of such connection. In the level of sentences, we have *subordinating conjunctions*, which have a similar role in connecting clauses. Such function words have often just one dependent – a content word which is in the relation with its grandparent through the parent function word.
- **Other function words** – The third group is for the rest of function words, which usually have more than one dependents. The forms of the verbs *'to be'*, *'to do'*, *'to have'*, etc. and their equivalents in other languages are the typical examples. These verbs should be treated as function words, nevertheless, they can be in a role of the main finite verb, in which they can have many dependents. The verb *'was'* in the sentence *'He was not yesterday in the bar with that girl.'* has five dependents and cannot be included into any of the previous two groups. Other *auxiliary* and *modal verbs* could be considered to belong to this group as well, since the content verbs are attached below them in many treebanks.

Note that the assignment of a function word to one of the proposed groups as well as the boundary between function and content words can differ across different linguistic theories. We do not want to argue about the correct annotation of function words.[1] We only show that function words can be easily grouped according to the number of their dependents.

[1] Differences in annotations over different treebanks are discussed e.g. in [6].

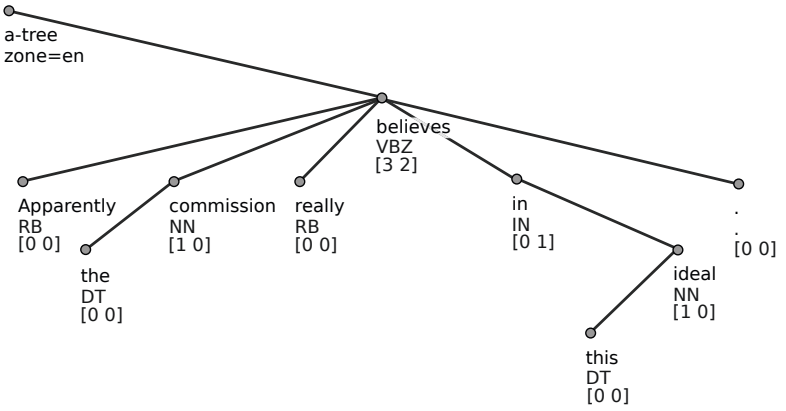

Fig. 2. An example of an English dependency tree. Fertility patterns are given in square brackets.

3.2 Fertility patterns

We use the term *fertility pattern* to express how many of left and right dependents (children) a word has in the dependency structure. We define it as a pair $[l, r]$, where l is the number of children preceding this word, and l is the number of children following this word. An example of word fertility patterns in an English sentence is given in Figure 2. The fertility patterns are shown in square brackets.

From the analysis given in Section 3.1, we can say that the fertility pattern of function words is often fixed. Majority of function words have either no dependents (pattern [0 0], e.g. articles, pronouns, auxiliary verbs) or just one dependent to the right (fertility pattern [0 1], e.g. prepositions or conjunctions) or to the left (fertility pattern [1 0], e.g. postpositions).

To support this hypothesis, we perform the following experiments that are run across 20 testing treebanks from CoNLL shared tasks 2006 [7] and 2007 [8].

3.3 Most Frequent Fertility Patterns for Word Forms

The first experiment explores the relation between the most frequent fertility pattern of a given word[2] and its relative frequency.

1. For each word in a treebank, we go through all its occurrences in the treebank and collect counts of its fertility patterns.
2. We find the most frequent fertility pattern for each word and denote its relative frequency as $HF(word)$. This score says how much the fertility pattern is stable (fixed) for the particular word.

[2] We choose relative frequency of the most frequent fertility pattern as a stability rate. Similarly, we could use entropy or other information theory quantities; this is left for further research.

Table 1. Statistics of fertility patterns for selected English words from the example in Figure 2

rank	the		commission		believes		in		this	
1st ($HF(word)$)	[0 0]	1.00	[0 0]	0.33	[1 1]	0.37	[0 1]	0.96	[0 0]	0.98
2nd	–	–	[3 0]	0.19	[1 2]	0.24	[0 0]	0.02	[1 0]	0.01
3rd	–	–	[2 0]	0.14	[2 1]	0.16	[1 1]	0.01	[0 2]	0.00
4th	–	–	[1 0]	0.12	[3 2]	0.05	[0 2]	0.00	[0 1]	0.00
5th	–	–	[1 1]	0.05	[4 1]	0.05	[0 1]	0.00	[0 3]	0.00

3. We compute the word relative frequencies $F_W(word)$ for each word in the treebank.
4. We plot the points representing individual words into the graph. Each word is parametrized by $HF(word)$ and $F_W(word)$.

An example is given in Table 1, in which the five most frequent fertility patterns are listed for selected words from the example in Figure 2. The most frequent pattern $HF(word)$ is the one in the first row. As it was expected, the function words (*the, in, this*) have one dominant fertility pattern (their HF is 1.00, 0.96, and 0.98 respectively), whereas the content words (*commission, believes*) have much more options.

The graph showing the relation between $HF(word)$ scores and $F_W(word)$ frequencies generally is depicted in Figure 3. There are all the words from all the 20 testing treebanks plotted in one graph. We can see that the very frequent words (with $F_W(word) > 0.02$) have often very stable fertility patterns ($HF(word) > 0.7$), which supports our previous hypothesis.

3.4 Most Frequent Fertility for Part-of-Speech Tags

We compute analogous statistics for part-of-speech (POS) tags.

1. For each POS tag in a treebank, we go through all its occurrences in the treebank and collect counts of its fertility patterns.
2. We find the most frequent fertility pattern for each POS tag and denote its relative frequency as $HF(tag)$. This score says how much the fertility pattern is stable for that POS tag.
3. We compute $F_T(tag)$, the aggregated relative frequency of words labeled by that POS tag as defined in Section 2, for all the POS tags in the treebank.
4. We plot the points representing individual POS tags into one graph. Each POS tag is parametrized by $HF(tag)$ and $F_T(tag)$. Moreover, we can express frequencies of individual POS tags by different sizes of the points. It is worth here since the POS tag relative frequency differs from the $F_T(tag)$.

The generated plot over all 20 treebanks is shown in Figure 4. We can observe a similar shape as for word forms in Figure 3. Almost all the tags with the

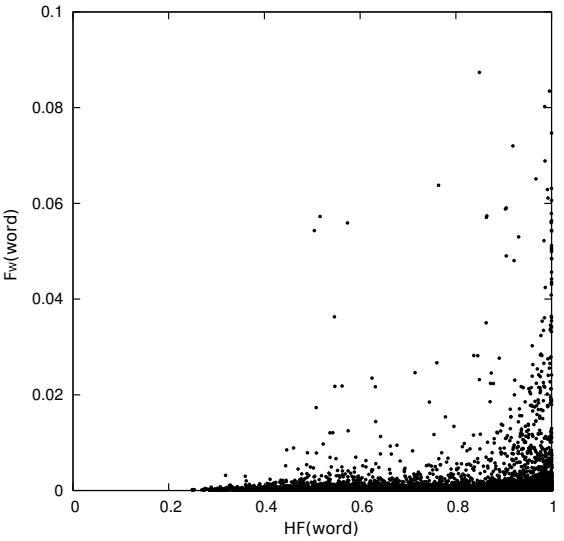

Fig. 3. Relationship between relative frequency of the most frequent fertility pattern of a word ($HF(word)$ as defined above) and the word relative frequency. There are all word types from 20 testing treebanks plotted into one graph.

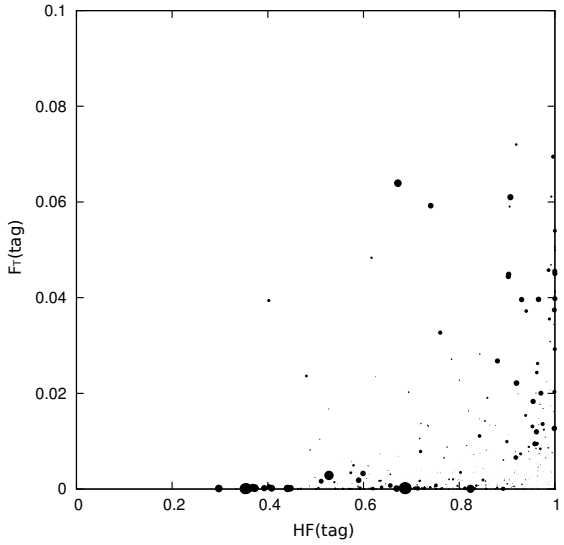

Fig. 4. Relationship between relative frequency of the most frequent fertility pattern of a POS tag ($HF(tag)$ as defined above) and the aggregated relative frequency of words labeled by that POS tag ($F_T(tag)$ as defined in Section 2). There are all POS tags from 20 testing treebanks plotted into one graph. The sizes of points shows relative frequencies of individual POS tags.

aggregated word frequency F_T higher than 0.02 have their most frequent fertility pattern $HF(tag)$ higher than 0.7. Therefore, we can say that our hypothesis holds for individual part-of-speech tags as well and across different treebanks.

4 Applying Function Word Properties in Unsupervised Parsing

In this section, we employ our previously described properties of function words in the task of unsupervised dependency parsing.

4.1 Introduction to the Unsupervised Dependency Parsing System

We use the unsupervised dependency parsing system described by Mareček and Straka in [9]. The software is freely available at `http://ufal.mff.cuni.cz/udp/`. The unsupervised parser is based on Dependency Model with Valence, which was introduced by Klein and Manning [10] and further improved by Headden et al. [11] and Spitkovsky et al. [12,13]. Inference procedure is based on blocked Gibbs sampling technique [14,15].

The Dependency Model with Valence is a generative model based on two probabilities. The first one, $P_{choose}(t_d|t_g, dir)$, expresses the probability of POS tag t_d of the dependent given the POS tag t_g of its governing word and the direction dir, which represents the *left* or *right* attachment.

The other one, $P_{stop}(x|t_g, dir, adj)$, expresses the probability that a word labeled by POS tag t_g has ($x = STOP$) or has not ($x = CONT$) children in the direction dir. The adjacency parameter adj determines whether we predict the first child in a given direction or a next child after one that already exists. For example, $P_{stop}(STOP|NN, left, 1)$ gives the probability that a noun (tag NN) has no children in the left direction; $P_{stop}(CONT|NN, left, 0)$ gives the probability that the noun that already has children in the left direction will have one more child in that direction.

Another important feature, from which the unsupervised dependency parser by Mareček and Straka [9] benefits, is so called reducibility. The idea is that if a word, or a short sequence of words $w_1, \ldots, w_n$, can be removed from a sentence without damaging its correctness, nothing else in the sentence can depend on any of the words $w_1, \ldots, w_n$. In other words, such a sequence of words forms a subtree or more adjacent subtrees in the respective dependency structure. By computing such statistics on large corpora, the prior probabilities for the P_{stop} model can be estimated and used to force the inference procedure to tend to better solutions.

4.2 Predicting P_{stop} Probabilities for Functional Words

As shown above, we hypothesize that if a word is frequent, it should have a stable fertility pattern, i.e. fixed number of left and right children.

To predict the number of children, we use the reducibility principle, which we described in [16]. A phrase (sequence of words) is reducible if the rest of the sentence after removing this phrase exists elsewhere in the corpus as a sentence. For example, the phrase *'on Monday'* is reducible from the sentence *'He arrived to London on Monday.'*, if the rest of the sentence *'He arrived to London.'* exists elsewhere in the corpus as well. The phrase *'arrived to'* is probably not reducible, since the sentence *'He London on Monday.'* can hardly be found in the corpus. It is evident that we can find only very few reducible sequences with this procedure. However, even if it leads to very sparse statistics on words, it is already sufficient for recognizing prototypical properties of individual part-of-speech tags. We search for reducible sequences on large collections of Wikipedia articles provided by [17] containing between 10 and 80 million words for each language.

By this simple procedure, we can search for reducible sequences. For our purposes, we need to compute the following counts:

- $red(t)$ – number of times a single word labeled by POS tag t can be removed from a sentence,
- $red_l(t)$ – number of times a two- or three-word phrase beginning with a word labeled by POS tag t can be removed,
- $red_r(t)$ – number of times a two or three word phrase ending with a word labeled by POS tag t can be removed.

Only the reducible sequences (phrases) consisting of at most three words are taken into account, since longer reducible sequences do not reflect grammatical properties and introduce a significant noise into the counts.

In the following experiments, we will use three rules designed to recognize fertility patterns of function words POS tags using the precomputed reducibility statistics $red(t)$, $red_l(t)$, and $red_r(t)$.

1. **function words with one right dependent** (fertility pattern [0 1]) are words for which $red_l(t) > 3\,red(t)$ and $red_l(t) > 3\,red_r(t)$,
2. **function words with one left dependent** (fertility pattern [1 0]) are words for which $red_r(t) > 3\,red(t)$ and $red_r(t) > 3\,red_l(t)$,
3. **function words with no dependents** (fertility pattern [0 0]) are words for which $red(t) > 10$ and $red(t) > red_l(t)$ and $red(w) > red_r(t)$.

We set the threshold for function words POS tags frequency to $F_T(t) = 0.005$. If the conditions of one of the three rules are fulfilled for a particular POS tag with $F_T(t) \geq 0.005$, we set its left and right P_{stop} prior probabilities to 1.0 or 0.0 according to the predicted fertility pattern. All the constants included in the proposed rules were set manually. Automatic optimization procedure was left for future research.

5 Experiments and Results

We follow the same experimental settings as Mareček and Straka [9]. To compute the reducibility scores of individual POS tags, we use Wikipedia articles

Table 2. Unlabeled attachment scores (in %) of unsupervised dependency parsing measured on 20 treebanks from CoNLL 2006 and 2007 shared tasks. The average across all the treebanks for each experiments is shown in the last row.

CoNLL		this work				previous works
language	year	baseline (mar13)	rules 1,2	rule 3	rules 1,2,3	spi13
Arabic	06	**35.3**	34.9	**35.3**	35.2	9.3
Arabic	07	38.2	43.0	38.6	**43.3**	26.8
Basque	07	**35.5**	35.3	**35.5**	35.3	24.4
Bulgarian	06	54.9	56.6	54.6	56.8	**63.4**
Catalan	07	67.0	43.8	65.0	43.9	**68.0**
Czech	06	52.4	**55.9**	53.7	55.6	44.0
Czech	07	51.9	**54.6**	54.2	54.3	34.3
Danish	06	41.6	**45.9**	42.5	45.5	21.4
Dutch	06	47.5	30.8	51.3	**54.8**	48.0
English	07	55.4	56.1	57.9	**58.5**	58.2
German	06	52.4	45.2	54.0	45.0	**56.2**
Greek	07	26.3	33.7	26.2	34.5	**45.4**
Hungarian	07	34.0	34.3	34.0	34.1	**58.3**
Italian	07	39.4	**51.0**	39.7	**51.0**	34.9
Japanese	06	61.2	61.0	62.2	61.9	**63.0**
Portuguese	06	69.6	46.2	72.0	**75.1**	74.5
Slovenian	06	35.7	47.4	35.9	47.3	**50.9**
Spanish	06	61.1	61.1	61.2	**61.7**	61.4
Swedish	06	54.5	54.2	55.6	**55.6**	49.7
Turkish	07	56.9	57.0	56.6	**57.0**	37.9
Average:		48.5	47.4	49.3	**50.3**	46.5

collection by [17], which was tagged using the TnT tagger [18]. On the same data, we compute the $red(t)$, $red_l(t)$, and $red_r(t)$ counts for the function words fertility pattern predictions. As the testing treebanks, we use 20 dependency treebanks from CoNLL shared tasks 2006 [7] and 2007 [8], which comprise 18 different languages.

Table 2 shows our results. We evaluate four different configurations – the baseline p_{stop} priors (results from our last work (mar13) [9]), p_{stop} priors with updated values for POS tags with fertility patterns [0 1] and [1 0] (rules 1 and 2), p_{stop} priors with updated values for POS tags with fertility pattern [0 0] (rule 3), and p_{stop} priors with values updated by all the three rules. We compare our results with the state-of-the-art results reported by Spitkovsky et al. [19] (spi13).

We can see that the configuration, in which all the three rules were applied, has the highest average attachment score (50.3%) over our 20 testing treebanks. It is also important that this configuration improved the scores for almost all the treebanks, compared to the baseline configuration. It worsened the attachment score significantly only for German and Catalan.

Table 3 provides a list of POS tags affected by our rules for all testing treebanks. Interestingly, the rule 2 was not applied at all. The reason may be the fact

Table 3. POS tags affected by the rules 1 and 3. Rule 2 did not affect any POS tag. Basic types of POS tags are in brackets: *prep* are prepositions, *punc* is punctuation, *det* are determiners, *pron* are pronouns, and *conj* are conjunctions. The last column (*err/all*) shows the number of POS tags, for which the pattern prediction was not correct, and the total number of POS tags affected.

CoNLL		pattern [0 1] (rule 1)	pattern [0 0] (rule 3)	err/all
Arabic	06	–	–	0/0
Arabic	07	P-*(prep)*	G-*(punc)*	0/2
Basque	07	–	PUNC*(punc)* KOMA*(punc)*	0/2
Bulgarian	06	R*(prep)*	–	0/1
Catalan	07	da*(det)* pr*(pron)*	Fc*(punc)*, Fp*(punc)*	**2**/4
Czech	06	R*(prep)*	:*(punc)*	0/2
Czech	07	R*(prep)*	:*(punc)*	0/2
Danish	06	SP*(prep)*	XP*(punc)*	0/2
Dutch	06	Prep*(prep)*	Punc*(punc)*, Art*(det)*	0/3
English	07	IN*(prep)*	,*(punc)* .*(punc)* DT*(det)*	0/4
German	06	APPR*(prep)* ART*(det)*	$(*(punc)* $,*(punc)* $.*(punc)*	**1**/5
Greek	07	AtDf*(det)* AsPpSp*(prep)* AsPpPa*(prep)*	PUNCT*(punc)*	**1**/5
Hungarian	07	Tf*(det)*	WP*(punc)* SP*(punc)* Cc*(conj)*	**1**/4
Italian	07	RD*(det)* E*(prep)*	PU*(punc)* C*(conj)*	**1**/4
Japanese	06	–	–	0/0
Portuguese	06	prp*(prep)*	punc*(punc)* art*(det)*	0/3
Slovenian	06	ad-pr*(prep)*	PUNC*(punc)*	0/2
Spanish	06	sp*(prep)*	Fc*(punc)* Fe*(punc)* Fp*(punc)* di*(det)* da*(det)*	0/6
Swedish	06	–	IK*(punc)* PO*(pron)* IP*(punc)*	0/3
Turkish	07	–	–	

that function words are often attached to (or govern) the following content word, at least for the languages we experiment with. Therefore the fertility pattern [0 1] is much more probable than the pattern [1 0]. The correct fertility pattern was predicted for 48 POS tags out of 54 POS tags for which the prediction was made (see the last column in Table 3).

We can also find out why the parsing accuracy of German and Catalan was worsened by our predictions. In both cases, the patterns for articles (German *ART* and Catalan *da*) was wrongly predicted as [0 1] instead of [0 0], probably because they are obligatory in that languages in majority of cases. This caused that the following nouns were more forced to become their dependents, which is not in accordance with the treebanks' annotation rules. However, note that choosing articles as the noun governors is not entirely an error. See the debate about the DP-hypothesis in [20]. The fertility pattern of Hungarian articles (*Tf*) was wrongly predicted as well, however, it does not affect the attachment score, since the same problem occurred in the baseline dependency trees.

6 Conclusions

We described the properties of function words and we proposed methods how to recognize them and how to predict whether they tend to be leaves in the dependency trees or they tend to have left or right dependents. We employed such methods in unsupervised dependency parsing system and show substantial improvement in attachment scores when testing on 20 different dependency treebanks from CoNLL shared tasks. To our knowledge, the achieved performance constitutes a new state of the art for about a half of the languages under study.

Acknowledgments. This research has been supported by the grant no. GPP406/14/06548P of the Grant Agency of the Czech Republic and by the European Union Seventh Framework Programme under grant agreement FP7-ICT-2013-10-610516 (QTLeap). This work has been using language resources developed and/or stored and/or distributed by the LINDAT/CLARIN project of the Ministry of Education of the Czech Republic (project LM2010013).

References

1. Tesnière, L.: Eléments de syntaxe structurale. Editions Klincksieck, Paris (1959)
2. Sgall, P., Hajičová, E., Panevová, J.: The Meaning of the Sentence in Its Semantic and Pragmatic Aspects. Reidel, Dordrecht (1986)
3. Menezes, A., Richardson, S.D.: A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In: Proceedings of the Workshop on Data-driven Methods in Machine Translation, vol. 14, pp. 1–8 (2001)
4. Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., Schneider, N.: Abstract meaning representation for sembanking. In: Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, pp. 178–186. Association for Computational Linguistics, Sofia (August 2013)
5. Zipf, G.K.: The Psychobiology of Language. Houghton Mifflin, Boston (1935)
6. Zeman, D., Mareček, D., Popel, M., Ramasamy, L., Štěpánek, J., Žabokrtský, Z., Hajič, J.: HamleDT: To Parse or Not to Parse? In: Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012). European Language Resources Association (ELRA), Istanbul (2012)
7. Buchholz, S., Marsi, E.: CoNLL-X shared task on multilingual dependency parsing. In: Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X 2006, pp. 149–164. Association for Computational Linguistics, Stroudsburg (2006)
8. Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D.: The CoNLL 2007 Shared Task on Dependency Parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, pp. 915–932. Association for Computational Linguistics, Prague (June 2007)
9. Mareček, D., Straka, M.: Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, vol. 1: Long Papers, pp. 281–290. Association for Computational Linguistics, Sofia (August 2013)

10. Klein, D., Manning, C.D.: Corpus-based induction of syntactic structure: models of dependency and constituency. In: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, ACL 2004. Association for Computational Linguistics, Stroudsburg (2004)
11. Headden III, W.P., Johnson, M., McClosky, D.: 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 2009, pp. 101–109. Association for Computational Linguistics, Stroudsburg (2009)
12. Spitkovsky, V.I., Alshawi, H., Jurafsky, D.: Punctuation: Making a point in unsupervised dependency parsing. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learning, CoNLL 2011 (2011)
13. Spitkovsky, V.I., Alshawi, H., Jurafsky, D.: Three Dependency-and-Boundary Models for Grammar Induction. In: Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP CoNLL 2012 (2012)
14. Gilks, W.R., Richardson, S., Spiegelhalter, D.J.: Markov chain Monte Carlo in practice. Interdisciplinary statistics. Chapman & Hall (1996)
15. Mareček, D., Žabokrtský, Z.: Gibbs Sampling with Treeness constraint in Unsupervised Dependency Parsing. In: Proceedings of RANLP Workshop on Robust Unsupervised and Semisupervised Methods in Natural Language Processing, Hissar, Bulgaria, pp. 1–8 (2011)
16. Mareček, D., Žabokrtský, Z.: Exploiting reducibility in unsupervised dependency parsing. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, pp. 297–307. Association for Computational Linguistics, Stroudsburg (2012)
17. Majliš, M., Žabokrtský, Z.: Language richness of the web. In: Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012). European Language Resources Association (ELRA), Istanbul (May 2012)
18. Brants, T.: TnT - A Statistical Part-of-Speech Tagger. In: Proceedings of the Sixth Conference on Applied Natural Language Processing, pp. 224–231 (2000)
19. Spitkovsky, V.I., Alshawi, H., Jurafsky, D.: Breaking out of local optima with count transforms and model recombination: A study in grammar induction. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1983–1995. Association for Computational Linguistics, Seattle (October 1995)
20. Abney, S.P.: The English Noun Phrase In Its Sentential Aspect. PhD thesis. MIT (1987)