LASSY: LARGE SCALE SYNTACTIC ANNOTATION OF WRITTEN DUTCH

Deliverable 4-2: Evaluation of Alpino on Lassy Small

1 Background

Lassy Small is the Lassy corpus in which the syntactic annotations have been manually verified. This part contains one million words. The composition of the corpus is detailed in deliverable 1.1.

Lassy Large is a much larger corpus in which syntactic annotations have been assigned automatically. The dependency structure annotations are assigned by the Alpino parser. In addition, postag and lemma annotations have been added by TadPole [2]. The composition of Lassy Large is documented in deliverable 1.2.

In this deliverable we report on the accuracy of the Alpino parser on Lassy Small. The purpose of this exercise to be able to estimate the quality of the syntactic dependency structures assigned by Alpino in Lassy Large.

The version of Alpino that was used in the following experiments is release 19076, October 1st, 2010. The version of Lassy Small that was used is release 19076, October 1st, 2010.

2 Results

The Lassy Small corpus is composed of a number of sub-corpora. Each sub-corpus is composed of a number of documents. In the experiment, Alpino was applied to a single document, using the options

```
user_max=190000 -test1 -veryfast
```

With these options, the parser delivers a single parse, which it believes is the best parse according to a variety of heuristics. These include the disambiguation model and various optimizations of the parser presented in [4], [1] and [3]. Furthermore, the parser cannot spend more than 190 seconds on a single sentence. If no result is obtained within this time, the parser is assumed to have returned an empty set of dependencies, and hence such cases have a very bad impact on accuracy.

The same options have been used for the construction of the Lassy Large corpus.

Below we list mean accuracy in terms of *named dependency accuracy*, as defined in [3], and repeated below. This metric is argued to be more appropriate than an evaluation in terms of precision, recall and f-score of dependencies. For completeness sake, we give those numbers also.

Let D_p^i be the number of dependencies produced by the parser for sentence i, D_g^i is the number of dependencies in the treebank parse, and D_o^i is the number of correct dependencies produced by the parser. If no superscript is used, we aggregate over all sentences of the test set, i.e.,:

$$D_p = \sum_i D_p^i; \quad D_o = \sum_i D_o^i; \quad D_g = \sum_i D_g^i$$

Using these definitions, it is straightforward to define precision (P) as D_o/D_p. Recall (R) is given by D_o/D_g. F-score is defined in terms of precision and recall as usual: $2\mathrm{P}\cdot\mathrm{R}/(\mathrm{P}+\mathrm{R})$.

An alternative similarity score is based on the observation that for a given sentence of n words, a parser would be expected to return (about) n dependencies. In such cases, we can simply use the percentage of correct dependencies as a measure of accuracy. To allow for some discrepancies between the number of expected and returned dependencies, we divide by the maximum (per sentence) of both. This leads to the following definition of *named dependency accuracy*.

$$\text{Acc} = \frac{D_o}{\sum_i \max(D_g^i, D_p^i)}$$

In the presentation of the results, we aggregate over sub-corpora. In table 1 we show the composition of each of these sub-corpora. The various `dpc-` sub-corpora are taken from the Dutch Parallel Corpus, and meta-information should be obtained from that corpus. The various `WR-` and `WS` corpora are obtained from D-Coi. The `wiki-` subcorpus contains wikipedia articles, in many cases about topics related to Flanders.

Parsing results are listed in table 2. As can be observed from this table, parsing accuracies are fairly stable across the various sub-corpora. An outlier is the result of the parser on the WR-P-P-G sub-corpus (legal texts), both in terms of accuracy and in terms of parsing times. We note that the parser performs best on the dpc-bal- subcorpus, a series of speeches by prime-minister Balkenende.

The experiments were performed on 64bit Linux workstations with 24Gb core memory and Six-Core AMD Opteron Processor 2435 cpu. Only a single core is used by the parser. Finally, the experiments were run with the environment variable PROLOGMAXSIZE set to 2000M. This implies that a single Alpino process cannot ever use more than 2 Gb of core memory.

sub-corpus	docs	sents	words
dpc-bal-	4	620	8825
dpc-bmm-	41	794	15589
dpc-cam-	11	508	9961
dpc-dns-	6	264	3833
dpc-eli-	12	603	11309
dpc-eup-	4	233	6085
dpc-fsz-	4	574	10967
dpc-gaz-	1	210	3806
dpc-ibm-	9	419	8473
dpc-ind-	22	1650	33928
dpc-kam-	1	52	1329
dpc-kok-	4	101	1846
dpc-med-	9	650	13575
dpc-qty-	9	618	13720
dpc-riz-	14	210	4217
dpc-rou-	21	1356	22640
dpc-svb-	3	478	7570
dpc-vhs-	7	461	6649
dpc-vla-	4	1915	32156
wiki	111	7341	98107
WR-P-E-C	5	1014	12239
WR-P-E-E	3	90	1813
WR-P-E-H	13	2832	32222
WR-P-E-I	44	9785	199150
WR-P-E-J	26	699	15015
WR-P-P-B	1	275	2008
WR-P-P-C	33	5648	83590
WR-P-P-E	3	306	5808
WR-P-P-F	3	397	6499
WR-P-P-G	5	279	6468
WR-P-P-H	109	2267	37241
WR-P-P-I	263	5789	115934
WR-P-P-J	4	1264	30021
WR-P-P-K	1	351	6982
WR-P-P-L	2	1115	20662
WS	99	14032	205940
total	911	65200	1096177

Table 1: Composition of the Lassy Small corpus

sub-corpus	prec	rec	f-score	Acc	msec/sent
dpc-bal-	92.78	92.75	92.77	92.54	1668
dpc-bmm-	88.30	87.32	87.81	87.11	4096
dpc-cam-	91.93	91.63	91.78	91.48	2913
dpc-dns-	90.56	90.40	90.48	90.26	1123
dpc-eli-	89.97	89.65	89.81	89.40	4453
dpc-eup-	90.91	88.88	89.88	88.67	8642
dpc-fsz-	86.50	84.99	85.74	84.72	4492
dpc-gaz-	89.03	87.99	88.51	87.83	3410
dpc-ibm-	90.26	90.01	90.13	89.56	4753
dpc-ind-	91.25	91.04	91.14	90.82	4010
dpc-kam-	90.51	89.14	89.82	88.95	4671
dpc-kok-	88.16	87.83	88.00	87.69	2546
dpc-med-	90.41	90.14	90.28	89.84	3906
dpc-qty-	90.05	89.68	89.86	89.50	7044
dpc-riz-	86.99	86.23	86.61	86.11	4926
dpc-rou-	91.58	91.42	91.50	91.15	2218
dpc-svb-	89.91	89.46	89.69	89.12	1839
dpc-vhs-	91.25	90.42	90.83	90.33	1819
dpc-vla-	90.75	90.38	90.57	90.07	2545
wiki	89.08	88.62	88.85	88.36	1940
WR-P-E-C	85.01	84.53	84.77	84.25	1827
WR-P-E-E	82.68	82.54	82.61	81.87	3599
WR-P-E-H	88.14	88.06	88.10	87.61	2110
WR-P-E-I	88.08	87.47	87.78	87.22	4051
WR-P-E-J	87.93	87.46	87.69	87.05	5276
WR-P-P-B	92.13	92.02	92.07	91.82	318
WR-P-P-C	88.29	87.87	88.08	87.44	2089
WR-P-P-E	89.33	88.94	89.14	88.52	3759
WR-P-P-F	84.00	82.23	83.11	81.92	4362
WR-P-P-G	81.44	79.23	80.32	78.72	10410
WR-P-P-H	91.48	91.37	91.42	91.09	2109
WR-P-P-I	90.51	90.35	90.43	90.05	3369
WR-P-P-J	87.40	86.19	86.79	85.79	6278
WR-P-P-K	89.49	89.25	89.37	88.85	3715
WR-P-P-L	89.00	88.39	88.70	88.02	3406
WS	90.48	90.31	90.40	90.12	1596
total	89.38	88.96	89.17	88.68	2819

Table 2: Parsing results of Alpino on the Lassy Small corpus

References

[1] Robbert Prins and Gertjan van Noord. Reinforcing parser preferences through tagging. *Traitement Automatique des Langues*, 44(3):121–139, 2003.

[2] Antal van den Bosch, Bertjan Busser, Sander Canisius, and Walter Daelemans. An efficient memory-based morphosyntactic tagger and parser for Dutch. In Peter Dirix, Ineke Schuurman, Vincent Vandeghinste, and Frank van Eynde, editors, *Computational Linguistics in the Netherlands 2006. Selected papers from the seventeenth CLIN meeting*, LOT Occassional Series, pages 99–114. LOT Netherlands Graduate School of Linguistics, 2007.

[3] Gertjan van Noord. Learning efficient parsing. In *EACL 2009, The 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 817–825, Athens, Greece, 2009.

[4] Gertjan van Noord and Robert Malouf. Wide coverage parsing with stochastic attribute value grammars. Draft available from the authors. A preliminary version of this paper was published in the Proceedings of the IJCNLP workshop Beyond Shallow Analyses, Hainan China, 2004., 2005.