Identifying sources of weakness in Syntactic Lexicon Extraction

Claire Gardent, Alejandra Lorenzo

CNRS/LORIA, Nancy (France)

Firstname.Secondname@loria.fr

Abstract

Previous work has shown that large scale subcategorisation lexicons could be extracted from parsed corpora with reasonably high precision. In this paper, we apply a standard extraction procedure to a 100 millions words parsed corpus of French and obtain rather poor results. We investigate different factors likely to improve performance such as in particular, the specific extraction procedure and the parser used; the size of the input corpus; and the type of frames learned. We try out different ways of interleaving the output of several parsers with the lexicon extraction process and show that none of them improves the results. Conversely, we show that increasing the size of the input corpus and modifying the extraction procedure to better differentiate prepositional arguments from prepositional modifiers improves performance. In conclusion, we suggest that a more sophisticated approach to parser combination and better probabilistic models of the various types of prepositional objects in French are likely ways to yield better results.

1. Introduction

A syntactic lexicon records for each verb, the nature and the type of its arguments. As has been repeatedly argued (Carroll and Fang, 2004; Jijkoun et al., 2004), a syntactic lexicon is an important resource for Natural Language Processing in that it provides valuable information e.g., for parsing, machine translation or surface realisation.

Relatedly, statistical methods have been developed which permit the automatic construction of subcategorisation lexicons (Briscoe and Carroll, 1997; Korhonen, 2002). Typically, these approaches proceed in two steps. First, a large corpus is parsed to extract verbs and their dependents from the available parse trees. Second, a statistical filter is applied to determine which of the extracted hypotheses are plausible. When using such a method for English, (Korhonen et al., 2006) reports a precision, recall and F-measure of 80.7%, 46.1% and 58.6%[1] respectively.

Clearly, however, these results depend on several language related factors such as the performance of the parser used and the inventory of subcategorisation frames to be extracted. Indeed, when applying an extraction procedure similar to that used by (Korhonen et al., 2006) to French, we obtain much lower results.

In this paper, we start by describing the application of the standard two-step procedure to French (section 2.). We then investigate different factors likely to improve performance namely, the specific extraction procedure and the parser used (sections 3. and 4.); the size of the input corpus (section 5.); and the type of frames learned (section 6.). To improve performance, we try out different ways of interleaving the output of several parsers with the lexicon extraction process and show that none of them improves the results (Section 4.). Conversely, we show that increasing the size of the input corpus (section 5.) and modifying the extraction procedure to better differentiate prepositional arguments from prepositional modifiers (section6.) improves performance. Finally, we perform a detailed result analysis which permits better identifying the source of errors (section 6.). In particular, this analysis permits pinpointing the impact of the syntactic frame vocabulary on the extraction results. Taking these observations into account, we show that by modifying the extraction procedure accordingly and by increasing the size of the parsed corpus, F-measure can be improved from 22.76% to 57.21% thereby getting much closer to the 58% results obtained by (Korhonen et al., 2006) for English. Based on the parsing and frame experiments, we conjecture that better results for French could be obtained using a more sophisticated approach to parser combination and better probabilistic models of the various types of prepositional objects typically distinguished by French grammarians. Additionally, of course, more accurate parsers, more data and the use of smoothing techniques using verb classes as backoff are all factors that could further improve subcategorisation extraction.

[1] We consider here the figures the article gives for the approach most similar to ours namely, Lexicon 2 in Table 2, an approach based on a relative frequency threshold filtering with no smoothing.

2. The base Extraction procedure

Following (Briscoe and Carroll, 1997), the base extraction procedure we use to construct a subcategorisation lexicon from a corpus proceeds in two steps. First, the corpus is parsed and verb dependents are extracted from the available parse trees (*Hypotheses extraction*). Second several statistical and symbolic filters are applied to determine which of the extracted hypotheses are plausible (*Hypotheses filtering*).

2.1. Hypotheses extraction

The corpus used is the CPC corpus for French[2], a 100 million words corpus containing data extracted in equal proportion from wikisource, frwiki, EstRepublicain, JRCacquis and Europarl.

To parse this corpus, during the first experiment we used the TagParser[3] which produces a dependency like annotation.

[2] http://atoll.inria.fr/passage/ressources.en.html

[3] www.tagmatica.com

For each verb instance in the resulting parsed corpus, we then extract the verb and the dependents of that verb which are related to it by one of the following relations : SUJ_V (subject), COD_V (object), CPL_V (prepositional complement), ATB (subject or object attribute).

2.2. Hypotheses filtering.

The second step of the extraction procedure carries out some format conversion and applies several symbolic and statistical filters.

Categories and relations are mapped to the Paris 7 Treebank (Abeillé and Barrier, 2004) format with syntactic categories NP, PP[PREP], VPinf, VPpart, Ssub, AP, AdvP and grammatical relations SUJ, OBJ, ATS, ATO, AOBJ, DEOBJ, POBJ. In particular, prepositional complements (CPL_V) are differentiated into POBJ, AOBJ and DEOBJ (i.e., complements introduced by the prepositions *à* and *de* respectively).

Passives are normalized to the corresponding active pattern. Multiple occurrences of dependents with identical function and category are reduced to one (because in a subcategorisation lexicon, a given grammatical function can only be required once by any given verb). Special cases such as clitics, merged determiners[4] and relative pronouns are dealt with[5]. Entries whose frame is not listed in a predefined list of acceptable subcategorisation frames (for French) are removed. Finally, lexical entries (i.e., verb/frame pairs) whose relative frequency is below a given threshold are filtered out.

2.3. Results

To evaluate the quality of the resulting lexicon (called EASYLEX), we compute precision, recall and F-measure with respect to a version of Dicovalence converted to our subcategorisation lexicon format (REF). Dicovalence is a subcategorisation lexicon specified by hand for 3 936 french verbs chosen among the most frequent (van den Eynde and Mertens, 2003). Table 1 gives the results obtained and compare them with some of the existing subcategorisation lexicons for French[6], namely TreeLex,SynLex and LexSchem. Recall (R) is the proportion of verb/frames pairs in the reference lexicon that are found by the system; precision (P), the proportion of verb/frames pairs in the extracted lexicon that are correct; and F-measure (F) is their harmonic mean namely $(2 * P * R)/(P + R)$. TreeLex (Kupsc and Abeillé, 2008) was automatically extracted from a hand-built treebank and manually validated. SynLex (Gardent et al., 2005) was automatically extracted from the LADL table and manually validated but is incomplete due to distribution restrictions on the LADL tables. Finally, LexSchem (Messiant, 2008; Messiant et al., 2008) was automatically extracted from a parsed corpora using techniques similar to ours. Note that all three lexicons had to be converted to a format compatible with that of EASYLEX and so conversion errors are possible. The figures given in Table 1 should therefore be considered with caution.

Lexicon	Verbs in REF	R	P	F
REF	3 936			
TreeLex	1 598	41.06	66.01	50.63
Synlex	2 467	51.07	42.88	46.62
Lexschem	1 299	40.33	76.6	52.8
EasyLex	3 649	69.2	13.62	22.76

Table 1: Results using the standard extraction procedure

As can be seen the results of our extraction are very low. As argued in (Gardent, 2009), this is partly due to the incompleteness of the reference lexicon. Note in particular, that although TreeLex was validated by hand and therefore only contains correct entries, its precision with respect to the reference is of only 66.01%. Thus Dicovalence fails to contain all correct entries. Conversely, however, a more complete gold standard is likely to decrease recall and possibly, F-measure. In any case, the results obtained by the standard extraction procedure are too low to be useful. They are also much lower than those obtained by the very similar approach used to construct LexSchem. In what follows, we investigate several components of the extraction procedure and experimentally assess their respective impact on the results with the aim of identifying clues for improvement.

First, we consider the extraction procedure used and show that a simple modification permits increasing F-measure by 15 points.

Second, we examine the impact of the parser. We first compare 4 parsers and then try different ways of combining the results obtained using these parsers and the extraction procedure. The best results are given by using the best parser alone.

Third, we test whether "More data is better data" by comparing the results obtained using more or less text data. We find that to gain a little under 1 point we need to multiply the size of the input corpus by 4.

Fourth, we investigate the impact of each grammatical relation on the results. We observe that by putting aside prepositional objects, the gain in precision and recall is of +1.48 and +8.29 points respectively.

3. Modifying the extraction procedure

The parser capacity to distinguish between modifiers and complements is an important factor in getting good extraction results. To assess the impact of this factor, we proceeded indirectly by augmenting the extraction procedure with (Zeman and Sarkar, 2000)'s iterative algorithm. The rationale for this is as follows. We observe that many of the frames produced by the extraction script are long and infrequent but include an acceptable frame. To capture such frames, we apply (Zeman and Sarkar, 2000)'s iterative treatment of frames with low relative frequency. The algorithm works as follows. Instead of being directly filtered out, frames whose relative frequency is below the set

[4]Merged determiners result from the fusion of a preposition and a determiner. For instance, when preceding the determiner *le* (*the*), the preposition *de* (*of*) becomes *du* and *le* is dropped.

[5]For a more precise description of how such special cases are handled, we refer the reader to (Gardent et al., 2009). Although the title is in French the content of this report is in English and gives full details about the extraction procedure.

[6]We did not consider all available lexicons as the conversion work required for the comparison is very time consuming.

Lexicon	R	P	F
EasyLex V1	69.2	13.62	22.76
EasyLex V2	62.97	27.72	38.19

Table 2: Results after integration of the iterative filter with a threshold of 0.1 (the one producing the best results). V1 and V2 are the extraction script without and with iterative procedure.

threshold are shortened by one argument and the resulting frame merged back into the current lexical entry. If the entry already contains this frame, the relative frequency of this frame and of the shortened frame are added. Else, the shortened frame inherits the relative frequency of the frames from which it is derived. The order of the dependents is normalized and the argument removed is always the right most one, that is, the most oblique one.

In effect, because it compacts several long frames with low frequency into shorter ones with higher frequency, (Zeman and Sarkar, 2000)'s iterative algorithm permits correcting the tendency of a parser to label as complements dependents which are, in fact, modifiers.

As Table 2 shows, the F-measure obtained when integrating this modification increases from 22.76 to 38.19.

4. Investigating the impact of parsing

Clearly, the particular parser used has an important impact on the results of lexical extraction.

Thus, a parser that is good at recognizing and labeling verb arguments will provide a better basis for lexical extraction than a parser that is not. Unfortunately, although a parsing evaluation campaign has recently been organized by the Passage project for French, the official results are not yet available so that the accuracy of the parser used for the extraction is not known with precision. It is therefore not possible to give the precise accuracy of the parser used. Nonetheless, preliminary results indicate an F-measure for the recognition of relations ranging from 57 to 68 (for the best parsers) i.e., much below the best Labeled Attachment Scores of the best dependency parsers for English.

Another relevant issue is that a parser which uses a syntactic lexicon to start with will indirectly provide this lexicon to the extraction process. Although this is an important point, we have not yet tried to measure the impact of the parser lexicon on the lexical extraction process. Instead, we use lexical extraction as a means to estimate the parsing system (including the grammar and the lexicon it uses). Note also that even if the extracted lexicon were not to differ from the subcategorisation used by the parser, the lexical extraction process would still be useful in that it provides frequency information about verb/frame pairs.

Given these caveats, we explore here two main points about the interaction between parsing and lexical extraction. First, we compare several parsers and measure their impact on the quality of the extracted subcategorisation lexicon. Second, we investigate whether different ways of interleaving the output of multiple parsers with the lexical extraction process permit improving results.

Parser	Threshold	R	P	F
Parser 1	0.13	54.68	59.14	56.82
Parser 2	0.15	50.80	58.89	54.55
Parser 3	0.15	51.48	56.25	53.76
Parser 4	0.15	50.99	55.64	53.21

Table 3: Comparing lexicons produced using different parsers. Note that the data set used is not the same than in the previous experiment (cf. Table 2). Hence the overall better results. The thresholds correspond to those producing the best results in each case.

4.1. Comparing parsers

To assess the impact of parsing on the results of the extraction, we used the results of the Passage parsing campaign mentioned above. The corpus used for this second experiment has a strong overlap with the corpus used in section 2. but is not identical. Further it has been submitted to a more extensive cleaning procedure and is probably cleaner as well. The results are shown in Table 3, where Parser 4 corresponds to the parser used in the first experiment (cf. Table 2).

As can be seen, the results are overall much better (the lowest F-measure is 53.21 against 38.19 in the previous experiment). Two facts might explain this strong increase in performance. First, in the period between our first experiment and the Passage parsing campaign, parser developers worked hard to improve their parser. Hence the same parser when used for the initial experiment and after submission to the parsing campaign is likely to exhibit an increased accuracy at the time of our second experiment. Second, the campaign organizers worked on further cleaning up the test corpus so that again, the corpora used in both experiments, although they cover roughly the same texts, might differ in how clean they are from non textual data.

Unsurprisingly, the experiment shows that the extraction results vary with the parser used. As Table 3 shows, using a different parser permits improving F-measure by 3.61 points.

4.2. Using several parsers

Next, we tried to improve results by combining the output of the four parsers. We tried several configurations, none of which resulted in a better lexicon.

4.2.1. Intersecting/Cumulating the lexicon extracted from several parsed corpora.

First we tried to improve results by taking either the union or the intersection of the lexicons extracted using different parsers. The results, shown in Table 4, are lower than the traditional approach using just one parser (cf. best result in Table 3). Intersection yields a slightly better result than union. However if we take the number of verbs covered into account, using union is more interesting as this small difference in performance is offset by a much larger number of verbs covered (e.g., 7 224 verbs using the union of 2 parsed corpora against 5 171 using their intersection).

		Intersection	Union
2 Parsers	Recall	48.76	57
	Precision	65.63	54.48
	F-measure	**55.95**	**55.71**
3 Parsers	Recall	46.65	58.72
	Precision	66.73	50.95
	F-measure	54.91	54.56
4 Parsers	Recall	45.65	59.37
	Precision	67.21	48.45
	F-measure	54.37	53.36

Table 4: Taking the union/intersection of several extracted lexicons

Nb of Parsers used	R	P	F
1 Parser	54.68	59.14	56.82
2 Parsers	53.94	58.68	56.21
3 Parsers	54.55	58.08	**56.26**
4 Parsers	54.36	57.73	55.99

Table 5: Filtering the hypotheses produced by several parsers. The values correspond to the best result in each case, which were always obtained using a threshold of 0.13.

4.2.2. Taking as input the hypotheses produced by several parsers

In the previous experiment we try, without success, to improve results by applying the union and the intersection operations on the final extraction output. A possible cause for the results is that, by the time these operations were applied, each lexicon had already gone through the hypotheses filtering step (cf. section 2.), possibly ruling out some verb/frame pairs which, if filtered after the application of the operations (particularly the union), would have remained. In this second experiment, we tried to overcome this problem by running the hypotheses filtering step on the hypotheses (i.e., Verb/Dependants patterns) produced not by one, but by several parsers. That is, we first combined the hypotheses produced by 2,3 and 4 parsers. We then the hypotheses filtering step on the resulting combination. We also experimented using different filtering thresholds. Table 5 shows the results. Again, they are lower than the traditional approach using just one parser.

4.2.3. Majority vote on the hypotheses

In this experiment, as done in the previous one, we aim to combine the hypotheses produced by several parsers before the filtering step. In an attempt to increase the precision of the results, we propose a new hypotheses combination, the majority vote, which works as follows. Given the Verb/dependents units produced by the different parsers during the hypothesis extraction step, we create a corresponding verb/frame pair only for these hypotheses whose relative frequency was higher than a given threshold t for at least n parsers (where n is a parameter ranging from 2 to 4). The frequency assigned to the resulting verb/frame pair is then the average of the frequencies of the n parsers. As shown in Table 6, the best results are obtained using decreasing thresholds as the number of parsers required to agree for the entry to be kept increases. Once again the best result (55.03) is lower than using just one parser (cf. Table 3).

2 Parsers Threshold=0.1	Recall	50.29
	Precision	60.21
	F-measure	54.81
3 Parsers Threshold=0.08	Recall	48.51
	Precision	63.43
	F-measure	54.98
4 Parsers Threshold=0.05	Recall	48.91
	Precision	62.91
	F-measure	**55.03**

Table 6: Using a Majority vote on the hypotheses

Initial data	Recall	54.68
	Precision	59.14
	F-measure	**56.82**
4 times more data	Recall	55.02
	Precision	59.58
	F-measure	**57.21**

Table 7: Using more data

In sum, it seems unlikely that a naive combination of parsers helps improve lexical extraction no matter how such combination interacts with the extraction process. Presumably, a more sophisticated approach is needed which builds on the specific strengths of each parser using e.g., the NIST ROVER (Recognizer output voting error reduction, (Fiscus, 1997)). Indeed since the four parsers used do not differ very strongly in terms of lexical extraction performance, it is likely that to improve lexical extraction, a better combining approach would be one that weighs each hypothesis differently depending on which parser has produced it and how well this parser is known to deal with the dependencies (e.g., object, a-object, etc.) involved in that particular hypothesis.

5. More data is better data

Finally, we tried augmenting the size of the corpus input to the parser. For this experiment we used Parser 1 (cf. Table 3), which is the parser with best results. As shown in Table 7, we found that a 409.9% increase in size only yields a 1 point performance improvement.

6. The impact of the chosen frame vocabulary

To get a clearer picture of what dependents are more difficult to recognize, we conducted an evaluation on sub-lexicons of EASYLEX which were obtained by systematically removing all entries containing either a given category, a given grammatical relation or a given frame. We focus here on the results concerning the relations. Thus the table below indicates the gain/decrease in precision and

recall for the lexicons obtained by retrieving from the reference lexicon and from the extracted lexicon, these entries which contain a given relation.

Recall	+8.291 POBJ	+5.597 AOBJ	+4.2409 DEOBJ	-1.84 OBJ
Precision	+1.482 POBJ	+1.326 DEOBJ	+0.945 AOBJ	-1.83 OBJ

As can be seen, recall and precision increase most when ignoring prepositional objects (POBJ). This confirms the observation made above that the parser used typically has difficulty differentiating between prepositional complements and prepositional modifiers. To improve recall, one possibility would be to extract not only verbal prepositional complements (CPL_V dependents) but also verb modifiers (marked as MOD_V). Similarly, AOBJ and DEOBJ complements are often not identified presumably because they are classified by the parser as MOD_V or by the extractor as POBJ. To improve recall on these categories, further criteria need to be introduced which permit better distinguishing AOBJ and DEOBJ from POBJ and MOD_V.

For each of these categories, the increase in precision is less than the increase in recall which suggests that the heuristics used by the extraction procedure to classify POBJ, DEOBJ and AOBJ complements are correct (although insufficient). Finally, removing entries that contain object complements decreases both recall and precision indicating that the parser and the extraction procedure are good both at finding and classifying these complements.

Generally, the above data suggests that the fine grained distinction required in French between POBJ, AOBJ and DEOBJ makes the lexical extraction process more complex and consequently, that more sophisticated models are needed which better account for this distinction.

7. Conclusion

In this paper, we have quantified the impact of several main factors on the automatic acquisition of a subcategorisation lexicon for French namely, the definition of the extraction procedure, the parser used, the size of the input corpus and the range of relations that has to be learned. In particular, we have shown that important sources of errors are related to poor parsing precision and to the poor ability of the parser to correctly classify prepositional dependents into modifiers, POBJ, DEOBJ and AOBJ.

Accordingly, we are currently investigating two main ways of improving the automated extraction of a subcategorisation lexicon for French. First, ROVER type techniques should be used to counteract the impact of parsers deficiencies by drawing on the output of several parsers rather than one. Second, a more sophisticated probabilistic model should be specified so as to better tailor the extraction procedure to the specificities of French such as in particular, the presence of clitics and relative pronouns which are strong (although ambiguous) indicators of a given grammatical function and of the various types of prepositional objects (AOBJ, DEOBJ and POBJ).

Acknowledgments

This research was jointly supported by the French National Research Agency (ANR) in the context of the Passage project (ANR-06-MDCA-013) and by the TALC theme of the CPER "Modélisation, Information et Systèmes Numériques" funded by the Région Lorraine.

8. References

A. Abeillé and N. Barrier. 2004. Enriching a french treebank. In *Proceedings of Language Ressources and Evaluation Conference (LREC)*, Lisbon.

E. Briscoe and J. Carroll. 1997. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing*, pages 356–363, Washington, DC.

J. Carroll and A. Fang. 2004. The automatic acquisition of verb subcategorisations and their impact on the performance of an HPSG parser. In *Proceedings of the 1st International Joint Conference on Natural Language Processing (IJCNLP)*, pages 107–114, Sanya City, China.

J. G. Fiscus. 1997. A post-processing system to yield reduced error word rates: Recognizer output voting error reduction (ROVER). In *IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 347–354.

C. Gardent, B. Guillaume, G. Perrier, and I. Falk. 2005. Extracting subcategorisation information from Maurice Gross' Grammar Lexicon. *Archives of Control Sciences*, 15(LI):253–264.

C. Gardent, C. Mouton, and E. de la Clergerie. 2009. D10 - technique d'acquisition de connaissances lexicales à partir de corpus analysés en syntaxe. Technical report, CNRS/LORIA, Nancy.

C. Gardent. 2009. Evaluating an automatically extracted lexicon. In *4th Language & Technology Conference*, Poznan, Poland.

V. Jijkoun, J. Mur, and M. de Rijke. 2004. Information extraction for question answering: Improving recall through syntactic patterns. In *COLING-2004*.

A. Korhonen, Y. Krymolowski, and T. Briscoe. 2006. A large subcategorization lexicon for natural language processing applications. In *Proceedings of LREC'06*, Genova, Italy.

A. Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, University of Cambridge.

A. Kupsc and A. Abeillé. 2008. Growing treelex. In Alexander F. Gelbukh, editor, *CICLing*, volume 4919 of *Lecture Notes in Computer Science*, pages 28–39. Springer.

C. Messiant, A. Korhonen, and T. Poibeau. 2008. Lexschem : A large subcategorization lexicon for french verbs. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, Marrakech.

C. Messiant. 2008. A subcategorization acquisition system for French verbs. In *Proceedings of the ACL-08: HLT Student Research Workshop*, pages 55–60, Columbus, Ohio, June. Association for Computational Linguistics.

K. van den Eynde and P. Mertens. 2003. La valence: l'approche pronominale et son application au lexique verbal. *Journal of French Language Studies 13, 63-104*.

D. Zeman and A. Sarkar. 2000. Learning verb subcategorization from corpora: Counting frame subsets. In *In Proceedings of LREC'00*, pages 227–233.