Effectively long-distance dependencies in French : annotation and parsing evaluation

Marie Candito[⋆] and Djamé Seddah[⋆◇]

[⋆] Alpage (Univ. Paris Diderot & INRIA), 175 rue du Chevaleret, 75013 Paris, France
[◇] Univ. Paris Sorbonne, 28, rue Serpente, 75006 Paris, France
marie.candito@linguist.jussieu.fr, djame.seddah@paris-sorbonne.fr

Abstract

We describe the annotation of cases of extraction in French, whose previous annotations in the available French treebanks were insufficient to recover the correct predicate-argument dependency between the extracted element and its head. These cases are special cases of LDDs, that we call *effectively* long-distance dependencies (eLDDs), in which the extracted element is indeed separated from its head by one or more intervening heads (instead of zero, one or more for the general case). We found that extraction of a dependent of a finite verb is very rarely an eLDD (one case out of 420 000 tokens), but eLDDs corresponding to extraction out of infinitival phrase is more frequent (one third of all occurrences of accusative relative pronoun *que*), and eLDDs with extraction out of NPs are quite common (2/3 of the occurrences of relative pronoun *dont*). We also use the annotated data in statistical dependency parsing experiments, and compare several parsing architectures able to recover non-local governors for extracted elements.

1 Introduction

While statistical parsers obtain high overall performance, they exhibit very different performance across linguistic phenomena. In particular, most statistical parsers perform poorly on long-distance dependencies (LDDs), which, though rare, are important to fully recover predicate-argument structures, which are in turn needed for semantic applications of parsing. Poor performance on LDDs is known of English statistical parsers, even though the training data does contain information for resolving unbounded dependencies (the Penn Treebank, or the specific dataset evaluated by Rimell et al. [17]). For French, the situation is worse, since the usual training data, the French Treebank (Abeillé and Barrier [1]), is a surface syntagmatic treebank that does not contain indications of LDDs : extracted elements bear a grammatical function, but no annotation indicates their embedded head. Hence syntagmatic stochastic French parsers cannot capture LDDs. Concerning dependency parsing, French dependency parsers can be learnt on the DEPFTB, resulting from the automatic conversion of the FTB into projective dependency trees (Candito et al. [3]). But we will show that this automatic conversion leads to wrong dependencies for particular cases of LDDs - cases we call effectively-long-distance dependencies (eLDDs), in which the fronted element is extracted from an actually embedded phrase -, and thus statistical parsers learnt on the DEPFTB are unable to recover such cases correctly.

In this paper, we describe the manual annotation, performed on the FTB and the Sequoia treebank (Candito and Seddah [5]), of the correct dependencies in eLDDs, leading to non-projective dependency treebanks. We then evaluate several dependency parsing architectures able to recover eLDDs.

2 Target linguistic phenomena

Extraction is a syntactic phenomena, broadly attested across languages, in which a word or phrase (the extracted element) appears in a non-canonical position with respect to its head. For a given language, there are well-defined contexts that involve and licence an extraction. In English or French, the non-canonical position corresponds to a fronting of the extracted element.

A first type of extraction concerns the fronting of the dependent of a verb, as in the four major types topicalization (1), relativization (2), questioning (3), it-clefts (4), in which the fronted element (in italics) depends on a verb (in bold):

(1) *À nos arguments*, (nous savons que) Paul **opposera** les siens.
To our arguments, (we know that) Paull will-oppose his.

(2) Je connais l' homme *que* (Jules pense que) Lou **épousera**.
I know the man that (Jules thinks that) Lou will-marry.

(3) Sais-tu *à qui* (Jules pense que) Lou **donnera** un cadeau?
Do-you-know *to whom* (Jules thinks that) Lou will-give a present?

(4) C' est Luca *que* (Jules pense que) Lou **épousera**.
It is Luca that (Jules thinks that) Lou will-marry.

Since transformational grammar times, any major linguistic theory has its own account of these phenomena, with a vocabulary generally bound to the theory. In the following we will use 'extracted element' for the word or phrase that appears fronted, in non canonical position. One particularity of extraction is 'unboundedness': stated in phrase-structure terms, within the clause containing the extraction, there is no limit to the depth of the phrase the extracted element comes from. In dependency syntax terms, for a fronted element f appears either directly to the left of the domain of its governor g, or to the left of the domain of an ancestor of g. We focus in this paper on the latter case, that we call **effectively** long-distance dependencies (**eLDD**). In examples (1) to (4), only the versions with the material in brackets are eLDDs.

In French, another case of eLDD is when a PP is extracted from a predicative complement either nominal or adjectival:

(5) la mélancolie à laquelle il est enclin
the melancholy to which he is prone (*'the melancholy he is prone to'*)

Other cases of eLDDs arise for some PPs with preposition *de*, which under some well-studied conditions (see for instance Godard [8]) can be extracted from subject or direct object NPs, as in (6).

(6) un échec *dont* (Léo me dit que) les *causes* sont bien connues
a failure of-whom (Léo to-me says that) the causes are well known
'a failure whose causes (Leo tells me) are well known'

Those *de*-phrases are precisely the ones that can be cliticized, with the anaphoric clitic *en* (*of-it*) appearing on the verb governing the NP, as in (7).[1]

(7) Tu *en* connais bien les **raisons**
you of-it know well the reasons 'You know well the reasons for it'

To sum up, eLDDs comprise any case of extraction out of predicative complements, nominal subjects or objects (examples (5) to (7)), and cases of extraction of a dependent of a verb (examples (1) to (4)) only when involving intervening heads, (such as *to think* in these examples).

3 Target French Treebanks

3.1 French treebank and Sequoia treebank

Our objective is to obtain a dependency treebank for French with correct governors for extracted elements. This treebank will thus contain non-projective links. We perform our annotation of cases of extraction on two treebanks :

- the French Treebank (Abeillé and Barrier [1]) (hereafter FTB), a constituency treebank made of 12351 sentences[2] from the national newspaper *Le Monde*
- the Sequoia treebank (Candito and Seddah [5]), an out-of-domain corpus annotated following the FTB's annotation scheme. It contains roughly 1000 sentences from the French wikipedia, 1000 sentences from the medical domain (from medicines' marketing authorization reports from the European Medecine Agency), 500 sentences from Europarl and 500 sentences from the regional newspaper *L'Est Républicain*.

These are constituency treebanks, in which the dependents of verbs are labeled with a grammatical function, since a given structural position may correspond to different grammatical relations. Candito et al. [3] describe a tool for the automatic

[1]Note that in that case, the dependency between the clitic and the noun is bounded, but we still consider it a eLDD, because the clitic appears locally to the head (the verb) of the NP it depends on.

[2]As distributed in 2007. The current release has around 4000 additional sentences.

conversion of such constituency trees into surface dependency trees.[3] We briefly describe below this procedure, and detail the incorrect result obtained for eLDDs.

3.2 Automatic conversion to surface dependencies

The conversion procedure is based on the classic technique of head propagation rules, proposed for English by Magerman [10], and outputs projective surface dependency trees (each token has exactly one governor, except the root) : (i) Nodes in phrase-structure trees are annotated with their lexical head, using head-propagation rules, that state how to find the syntactic head in the right-hand side of a CFG rule; (ii) Using the lexical heads, bilexical dependencies are extracted. If the constituent node for the dependent bears a functional label, it is used as the label of the dependency; (iii) Remaining unlabeled dependencies are labeled using heuristics.

With that technique, output dependency trees are necessarily projective, and non-local dependents are wrongly attached to the local lexical head, as exemplified in Figure 1 in which the accusative relative pronoun *que* is wrongly attached to the local head *semblaient* (*seemed*) instead of its actual syntactic head *partager* (*to share*). A crucial point here is that a wrong dependency arises only for LDDs that are eLDDs. For instance from a simplified version of tree (a), without the raising verb, we would obtain the correct dependency between *que* and the verb *partager*.[4]

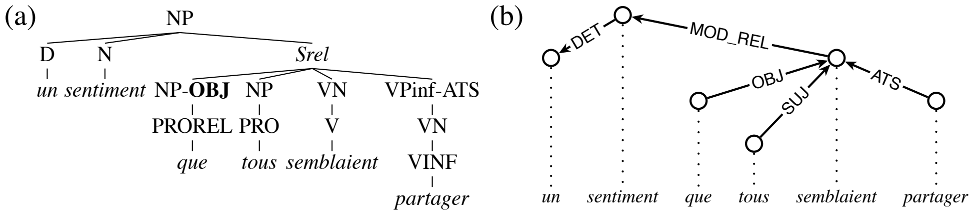

Figure 1: Left: An NP as annotated in the original French Treebank scheme (for *a feeling that (they) all seemed (to) share*). Right: corresponding automatically derived dependency tree, with wrong governor for the wh-word *que*.

4 Manual annotation

4.1 Selection of occurrences to annotate

One major difficulty to annotate extractions is that though ubiquitous in the linguistic literature, they are quite rare in actual texts. Further, some cases like topicalization (cf. example (1) above), involve structural clues only, and no lexical clue, namely no *wh*-words. Topicalization does exist in French, but is much rarer than

[3]Included in the BONSAI toolkit (http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html).

[4]Note that the anaphoric relation between the pronoun and its antecedent is then trivial to recover, provided the relative clause is correctly attached.

relativization, it-clefts or questioning, and is restricted to the extraction of prepositional phrases. Hence, because we could not afford to scan the whole treebank to look for extractions, and also because it is unclear whether such dependencies can be recovered with current parsing techniques, we chose as a first step to focus on words known to be likely to involve an extraction, namely the clitic *en* and wh-words (relative and interrogative pronouns and determiners). This allows to capture the cases exemplified in section 2, except topicalization.

We examined each occurrence of wh-word and of the clitic *en*, and annotated the correct dependency in case of non-local dependency.

4.2 Annotation scheme and methodology

4.2.1 Functional paths

We chose to annotate non-locality using *functional paths*, made of sequences of dependency labels, as proposed in the LFG framework. We were inspired by Schluter and van Genabith [18], who have used them to annotate a manually modified version of half the French Treebank.[5] Before formalizing this notion, let us take as example the automatically-derived dependency tree in Figure 1, in which the relative pronoun *que* is wrongly attached to the local governor *semblaient*. The non-local governor is *partager*. We define the functional path for *que* to be the sequence of labels appearing on the path between *que* and its correct governor, namely $OBJ.ATS$, which can be read as *"'que' should be the OBJ of the ATS of its local governor"*.

More generally, let d be a lexical item that should depend on a non-local governor nlg. Let ADT be the dependency tree obtained by automatic conversion from the source constituency tree, and CDT the correct dependency tree that we target. Since nlg is non-local, the tree ADT contains a wrong dependency $lg \xrightarrow{l_0} d$, while CDT contains $nlg \xrightarrow{l_0} d$ (we suppose here that the label l_0 is correct is ADT). For such cases, we define a functional path as the sequence of labels $l_0.l_1....l_n$ that appear, in the incorrect tree ADT, on the path between the dependent d and its non-local governor nlg.

The manual annotation consists in making functional paths explicit : in the previous formalization, it amounts to modifying ADT into ADT', by labeling the dependency $lg \xrightarrow{l_0} d$ with the functional path as label : $lg \xrightarrow{l_0.l_1...l_n} d$. Note that eLDDs are exactly the cases involving a functional path of length > 1 instead of a simple functional tag (which can be regarded as a functional path of length 1).

4.2.2 Automatic interpretation of functional paths

This kind of annotation can be used in a straightforward way to recover the correct governor of extracted elements. We give in figure 2 the algorithm used to interpret functional paths as the indication of non-local dependencies : it changes a tree

[5]But these authors obtain only 65 cases in total, suggesting the linguistic constructions covered are few. Note we chose to use reverse functional paths, for easier reading.

containing functional paths into a corresponding tree, in which dependents that are non-local, are attached to their non-local governors.

```
0  • $ADT' \leftarrow ADT$
1  • while $ADT'$ contains a dependency $a$ of the form $lg \xrightarrow{l_0.l_1...l_n} d$, with $n > 0$, do
2    • $i \leftarrow n$, $g_i \leftarrow lg$
3      • while $i > 0$ do
4        • $G \leftarrow$ nodes $x$ such as $g_i \xrightarrow{l_i} x$ and $x \neq d$
5          • if $G \neq \emptyset$, choose the left-most most appropriate node $x$, and set $g_{i-1} \leftarrow x$
6          • else, continues to next element in while 1
7        • $i \leftarrow i - 1$
8      • replace in $ADT'$ the dependency $a$ by $g_0 \xrightarrow{l_0} d$
```

Figure 2: Algorithm to interpret functional paths : find the non-local governors and change the dependency tree accordingly

If we go back to the example of Figure 1, the manual annotation applied to tree (b) is given in tree (c) in Figure 3. The interpretation of the sole functional path in tree (c) outputs the tree (d), which is non-projective.

The functional paths can be inconsistent with respect to the tree they appear in. This results in obtaining an empty set at line 4 in Figure 2. During the manual annotation phase, such cases can be used as warnings for a wrong functional path. When using the procedure in the parsing phase (see section 5), such functional paths are simply discarded and the label $l_0.l_1...l_n$ is replaced by l_0.

Further, the functional paths can be ambiguous. This is the case when the set G obtained at line 4 contains more than one element, at any point in the functional path. In these cases, the procedure prefers verbal governors over any other POS, and leftmost governor in case of remaining ambiguity.

This procedure is similar to the deprojectivization procedure defined by Nivre and Nilsson [15], when these authors use the encoding scheme they called 'Head'. More precisely, both procedures are equivalent in case of a functional path of length 2. We needed to define a procedure able to cope with paths of arbitrary length, in order to interpret any manually annotated functional path.

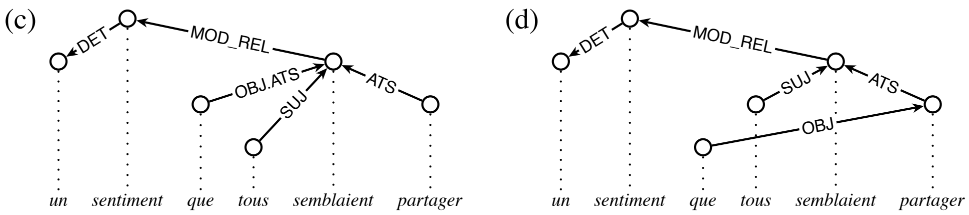

Figure 3: Left: Tree (c) is a modification of tree (b), with (manually) annotated functional path for the extracted element *que*. Right: Tree (d) is the output (non-projective) tree after using the algorithm Figure 2 to interpret functional paths.

4.2.3 Annotation methodology

Because the original treebanks we use have been annotated in constituency format, we chose to annotate the above-mentioned functional paths in the constituency trees, in order to retain the property that the dependency treebank can be automatically converted from the constituency trees. So, for the example of Figure 1, we annotate the functional path *OBJ.ATS* on the *NP* node that dominates the relative pronoun : NP-OBJ is replaced by NP-OBJ.ATS, then the usual constituency-to-dependency conversion produces the left tree of Figure 3, and the functional path interpretation procedure produces the right tree of Figure 3.[6]

We performed manual annotation of functional paths on a bracketed version of the French Treebank (called the FTB-UC by (Candito et al. [3])), and on the bracketed version of the Sequoia Treebank, using the WordFreak tool (Morton and LaCivita [13]), customized to handle the relevant sets of POS, non terminals and functional labels. The annotation for the words *en* and *dont* were performed independently by two annotators using WordFreak, and adjudicated by an expert annotator. The annotation for all the other wh-words were directly performed by a single expert annotator, because they potentially involve longer dependencies than for the word *en*, hence requiring to define more difficult (longer) functional paths. Then we applied the functional path interpretation algorithm (Figure 2), to obtain dependency versions of the FTB and the SEQTB with correct governors in case of eLDD. The resulting annotations are freely available.[7]

4.3 Quantitative characteristics

The resulting annotation provides a picture of the prevalence of extraction phenomena for a corpus of journalistic text (FTB) and for a corpus with mixed genres (the Sequoia corpus). We give various numbers of tokens for the annotated data in table 1, for the FTB, the SEQTB, and for the concatenation of both corpora.

The first observation is that the cases of extraction leading to eLDDs are very rare : for the whole FTB + SEQTB corpus, 0.16% of the tokens received a non-local governor (i.e. a functional path of length > 1). Further, more than 80% of eLDDs have a functional path of length 2, namely are "not-so-long" distance dependencies.

Focusing on projectivity, we see that only around two thirds of the eLDDs are non-projective (359 out of 618 dependencies). This is because most eLDDs are extractions from a subject NP, as in (6) (noted with functional path 'DEP.SUJ' : the extracted element is the dependent of the subject of the local (wrong) head). In

[6]We are aware that such an annotation task supposes a very good understanding of both the linguistic phenomena at play, and of the ad-hoc conversion to dependency procedure. Yet this has the advantage, over more traditional annotation using coindexed traces, that annotation on bracketed constituency trees is easier than on dependency structure.

[7]The SEQTB with correctly annotated eLDDs is available at https://www.rocq.inria.fr/alpage-wiki/tiki-index.php?page=CorpusSequoia. The FTB with non-local dependencies is available on request, provided you have the license for the FTB.

general, no dependent of the local verbal head intervenes between the subject and the extracted element (*en* or *dont*) as in (6), projectivity is preserved.[8]

	Number of tokens					
Treebank	total	with eLDD (%)	fpl=2	fpl=3	fpl>3	non projective
FTB	350931	555 (0.16 %)	466	69	20	317
SEQTB	69238	63 (0.09 %)	47	13	3	42
FTB + SEQTB	420169	618 (0.15%)	513	82	23	359

Table 1: For the FTB and the SEQTB, total number of tokens, tokens with non-local dependency, i.e. with length of functional path (fpl) > 1, tokens with fpl=2, fpl=3, fpl > 3; Number of tokens with fpl>1 that have a non projective dependency.

If we focus on the top three lexical items that exhibit eLDDs, we obtain the accusative relative pronoun *que*, the genitive relative pronoun *dont* and the anaphoric clitic *en*. As can be seen in table 2, these three lemmas totalize 570 out of the 618 cases of annotated eLDD in the FTB + SEQTB treebanks.[9] Note that though we saw eLDDs are very rare, these particular three lemmas have a substantial proportion of eLDD occurrences, especially *dont* (65.7% of its occurrences).

The most frequent element extracted from a verbal phrase is the relative pronoun *que*. A striking observation is that for all the 152 eLDD cases involving *que*, the embedded phrase *que* originates from is infinitival (as in the example Figure 3), and none is a finite clause.[10]

However, though the relative pronoun *dont* and the clitic *en* can either depend on verbs, nouns or adjectives, the majority of eLDDs are cases of extraction out of NPs. More precisely, out of subject NPs for *dont* (251 cases of functional paths DEP.SUJ out of 329) and out of object NPs for *en* (51 cases of functional paths DEP.OBJ, out of 89). The other prevalent case for clitic *en* is the extraction out of predicative complement (24 cases of functional paths DEP.ATS).

5 Parsing experiments

In this section we list and evaluate various parsing strategies able to output eLDDs. Though the overall parsing performance is unlikely to be affected by this parameter (given the very small amount of annotated eLDDs), it seems interesting to focus on attachment scores for the specific tokens that do often trigger eLDDs.

We relate experiments both using the "**local**" dependency trees, namely trees automatically converted from constituency trees without functional paths, and us-

[8]Non projectivity also arises when extracting a more deeply embedded dependent within the subject NP, as in *"une entreprise* dont *la moyenne d'âge des* **salariés** *dépasse 40 ans"* (*a company of-which the average of age of the employees is over 40*).

[9]Other cases concern (i) extractions of prepositional phrases (pied-piping), such as in example 3, for which the token that will bear the eLDD is the head preposition of the PP (*à* in example 3) or (ii) rare cases of extraction of NPs with wh-determiner, for which the noun bears the eLDD.

[10]We found no extraction out of an embedded finite clause in the FTB, and one in the SEQTB, in which the extracted element is a PP.

		Number of occurrences in FTB + SEQTB				
Lemma	POS	total	local gov	non-local gov	Top-3 most frequent fct paths	
que	relative pronoun	616	464	152 (32,8%)	OBJ.OBJ	77
					OBJ.OBJ.OBJ	22
					OBJ.OBJ.DE_OBJ	19
dont	relative pronoun	501	172	329 (65.7%)	DEP.SUJ	251
					DEP.OBJ	29
					DEP.ATS	27
en	accusative clitic	411	322	89 (21,7%)	DEP.OBJ	51
					DEP.ATS	24
					DEP.SUJ	11

Table 2: Statistics for the three lexical items having a non-local governor most frequently: total number of occurrences, number with and without local governor, and top-three most frequently-annotated functional paths.

ing the "**non local**" dependency trees, which have corrected dependencies for eLDDs (obtained via the procedure described in section 3.1). The local trees are projective, whereas the non local trees contain a few non projective links (two thirds of the eLDDs are non projective, cf. section 4.3).

We use the usual split for the 2007 version of the FTB (1235, 1235 and 9881 sentences for test, development and training sets respectively). For evaluation, we use as test sets the whole SEQTB on top of the usual FTB development and test sets.[11] In all our experiments the predicted parses are evaluated against the *non local* (pseudo-)gold dependency trees (while for training, we sometimes use the *local* dependency trees). We provide unlabeled and labeled attachment scores for all tokens except punctuation, and for the three lexical items that have a non local governor most frequently (the three lemma+POS pairs of Table 2).

We use MaltParser (Nivre et al. [14]), version 1.7 and MSTParser (McDonald [11]), version 0.5.0.[12] For the features, we use the best ones from a benchmarking of dependency parsers for French (Candito et al. [4]), except we removed unsupervised word clusters as features.[13] For MaltParser, we always use the arc-eager parsing algorithm, that can provide projective trees only.[14] For MSTParser, we either use order 1 factors and the Chu-Liu-Edmonds algorithm, that can output non projective trees, or order 2 factors, with the Eisner algorithm, restricted to projective trees. We also test pseudo-projective parsing (Nivre and Nilsson [15]), where non projective trees are transformed into projective ones, with label marking to allow the reverse the graph transformation: we use MaltParser to obtain projectivized versions of the non local trees, then either MaltParser or MSTParser to train a (projective) parser on these, and MaltParser again to de-projectivize the obtained

[11]As we did no parameter tuning, we did not reserve a test set for final test.

[12]Available at http://www.maltparser.org and http://sourceforge.net/projects/mstparser.

[13]POS are predicted using MORFETTE (Chrupała et al. [6]), and lemmas and morphological features are predicted using the BONSAI toolkit.

[14]Experiments with the swap algorithms showed degraded performance, supposedly due to the feeble amount of non-projective links.

predicted parses.[15]

			SeqTb			Ftb dev			Ftb test		
Parser	Type of training trees	Parsing algo	LAS	UAS	UAS on the 216 *que*, *en dont*	LAS	UAS	UAS on the 126 *que*, *en dont*	LAS	UAS	UAS on the 174 *que*, *en dont*
MALT	local	arc-eager	85.0	88.0	57.9 (125)	86.6	89.0	55.2 (75)	87.3	89.6	54.0 (94)
MALT	pproj	arc-eager	84.9	88.0	77.3 (167)	86.7	89.1	77.2 (105)	87.4	89.7	76.4 (133)
MST	non local	o1 np	85.0	88.2	71.8 (155)	86.5	89.2	80.2 (109)	87.3	89.9	80.5 (140)
MST	local	o2 proj	85.6	88.9	56.0 (121)	87.6	90.3	58.8 (80)	88.2	90.8	56.3 (98)
MST	non local	o2 proj	85.6	89.0	68.1 (147)	87.5	90.2	74.3 (101)	88.1	90.7	73.0 (127)
MST	pproj	o2 proj	85.7	89.1	71.3 (154)	87.6	90.3	80.9 (110)	88.4	91.0	78.7 (137)

Table 3: Parsing performance for malt or mst, with training on either local, non local, or projectivized (*pproj*) trees.

If we look at the overall performance, we observe, as usual, that MSTParser order 2 performs better than MaltParser. Further, training on the local, non local or projectivized trees has practically no impact (differences are not significant). When focusing on the lexical items that are likely to exhibit a non local dependency, the picture is quite different. First, it can be noted that using the non projective decoding for MSTParser, because it implies using order 1 factors, is not worth: the gain obtained for the eLDDs is too small to counterbalance the drop in performance when using order 1 instead of order 2 factors. Second, when comparing performance across the various training trees types, we note that pseudo-projectivization performs best on eLDDs. Moreover performance on eLDDs with training on projectivized data is comparable for MaltParser and MSTParser. The best strategy both overall and for eLDDs seems to be to use pseudo-projectivization with MSTParser and projective decoding, in order to use higher order factors.

6 Related Work

The Penn Treebank's annotation style of all types of long distance dependencies considerably eases the extraction of wide coverage grammars, and the building of high performing deep syntax parsers, as long as their underlying linguistic framework is able to cope with those complex phenomena (Hockenmaier et al. [9] for CCG; Cahill et al, [2] for LFG based parsing; Miyao et al. [12] for HPSG). Dependency parsers, when trained on a non projective dependency version of the Ptb, and with the use of post-parsing heuristics, exhibit a similar range of performance than the parsers cited above (Nivre et al. [16]). For French, as noted in introduction, non local dependencies were not natively annotated in the Ftb. This complicates of-course direct comparison of our work. Schluter and van Genabith [19] focus on

[15]We report results with the 'head' marking strategy of Nivre and Nilsson, performing very slightly better than the other two strategies (the difference not being significant).

producing treebank based LFG approximations for French, using for this a modified version of part of the FTB, where they annotated functional paths for a few cases. Clergerie [7] proposes a deep symbolic parser for French, which can recover LDDs directly (including eLDDs), but currently quantitative evaluation of LDDs for this parser remains to be performed. More generally, works on LDDs do not focus specifically on eLDDs, though they are the most difficult type of LDDs. For example, Rimell et al. [17] describe a 800 English sentences corpus used for the evaluation of wide coverage parsers on unbounded dependencies, but over all eight types of unbounded dependencies only one is exclusively of eLDD type.

7 Conclusion

We have annotated cases of *effectively long* distance dependencies for two French treebanks, totalizing over 15000 sentences. We could verify that non local dependencies are very rare in actual French texts: we annotated a non local governor for 513 tokens only out of over 420000 tokens. Yet, they are massive within the occurrences of the relative pronoun *dont*, and to a lesser extent *que*, and also frequent for the clitic *en*. We noted that extraction out of finite verbal clause is totally absent from the corpora we've annotated (one case only for over 15000 sentences), but extraction out of infinitival clause accounts for one third of the occurrences of the relative pronoun *que*. Further only 2 thirds of annotated eLDDs give rise to non projective links. As far as parsing is concerned, the best results, both overall and on eLDDs, are obtained when combining pseudo-projectivization and MSTParser with order 2 factors and projective decoding.

References

[1] Anne Abeillé and Nicolas Barrier. Enriching a french treebank. In *Proc. of LREC'04*, Lisbon, Portugal, 2004.

[2] Aoife Cahill, Michael Burke, Ruth O'Donovan, Josef van Genabith, and Andy Way. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In *Proc. of ACL'04*, pages 320–327, Barcelona, Spain, 2004.

[3] Marie Candito, Benoit Crabbé, and Pascal Denis. Statistical french dependency parsing : Treebank conversion and first results. In *Proc. of LREC'2010*, Valletta, Malta, 2010.

[4] Marie Candito, Joakim Nivre, Pascal Denis, and Enrique Henestroza Anguiano. Benchmarking of statistical dependency parsers for french. In *Proc. of COLING 2010*, pages 108–116, 2010.

[5] Marie Candito and Djamé Seddah. Le corpus sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In *Proc. of TALN 2012 (in French)*, Grenoble, France, June 2012.

[6] Grzegorz Chrupała. Morfette: A tool for supervised learning of morphology. `http://sites.google.com/site/morfetteweb/`, 2010. Version 0.3.1.

[7] Éric De La Clergerie. Convertir des dérivations TAG en dépendances . In *Proc. of TALN 2010*, Montreal, Canada, 2010. ATALA.

[8] Danièle Godard. Extraction out of NP in French. *Natural Language and Linguistic Theory*, 10(2):233–277, 1992.

[9] Julia Hockenmaier. *Data and models for statistical parsing with Combinatory Categorial Grammar*. PhD thesis, 2003.

[10] David M. Magerman. Statistical decision-tree models for parsing. In *Proc. of ACL'95*, pages 276–283, Morristown, NJ, USA, 1995.

[11] Ryan T. McDonald and Fernando C. N. Pereira. Online learning of approximate dependency parsing algorithms. In *Proc. of EACL'06*, 2006.

[12] Yusuke Miyao and Jun'ichi Tsujii. Probabilistic disambiguation models for wide-coverage hpsg parsing. In *Proc. of ACL 2005*, pages 83–90, 2005.

[13] Thomas Morton and Jeremy LaCivita. Wordfreak: an open tool for linguistic annotation. In *Proc. of NAACL 2003, Demos*, pages 17–18, 2003.

[14] Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proc. of LREC'06*, pages 2216–2219, Genova, Italy, 2006.

[15] Joakim Nivre and Jens Nilsson. Pseudo-projective dependency parsing. In *Proc. of ACL'05*, pages 99–106, Stroudsburg, PA, USA, 2005.

[16] Joakim Nivre, Laura Rimell, Ryan McDonald, and Carlos Gomez-Rodriguez. Evaluation of dependency parsers on unbounded dependencies. In *Proc. of COLING 2010*, pages 833–841, 2010.

[17] Laura Rimell, Stephen Clark, and Mark Steedman. Unbounded dependency recovery for parser evaluation. In *Proc. of EMNLP 2009*, pages 813–821, 2009.

[18] Natalie Schluter and Josef van Genabith. Preparing, restructuring, and augmenting a French Treebank: Lexicalised parsers or coherent treebanks? In *Proc. of PACLING 07*, Melbourne, Australia, 2007.

[19] Natalie Schluter and Josef Van Genabith. Dependency parsing resources for french: Converting acquired lexical functional grammar f-structure annotations and parsing f-structures directly. 2009.