Improving a symbolic parser through partially supervised learning

Éric Villemonte de la Clergerie

Improving a symbolic parser through partially supervised learning

Éric Villemonte de la Clergerie
INRIA - Rocquencourt - B.P. 105
78153 Le Chesnay Cedex, FRANCE
Eric.De_La_Clergerie@inria.fr

Abstract

Recently, several statistical parsers have been trained and evaluated on the dependency version of the French TreeBank (FTB). However, older symbolic parsers still exist, including FRMG, a wide coverage TAG parser. It is interesting to compare these different parsers, based on very different approaches, and explore the possibilities of hybridization. In particular, we explore the use of partially supervised learning techniques to improve the performances of FRMG to the levels reached by the statistical parsers.

1 Introduction

Most stochastic parsers are trained and evaluated on the same source treebank (for instance the Penn TreeBank), which, by definition, avoid all problems related to differences between the structures returned by the parsers and those present in the treebank. Some symbolic or hybrid parsers are evaluated on a treebank specifically designed for their underlying formalism, possibly by converting and hand-correcting the treebank from some other annotation scheme (as done in (Hockenmaier and Steedman, 2007)). Besides the cost of the operation, an issue concerns the comparison with other parsers. By contrast, the most problematic case remains the evaluation of a parser on an unrelated treebank and scheme (Sagae et al., 2008).

This situation arose for French with the recent emergence of several statistical parsers trained and evaluated on the French TreeBank (FTB) (Abeillé et al., 2003), in particular under its dependency version (Candito et al., 2010b) represented in CONLL format (Nivre et al., 2007). On the other hand, older parsing systems still exist for French, most of them not based on statistical approaches and not related to FTB. For instance, FRMG is a wide coverage symbolic parser for French (de La Clergerie, 2005), based on Tree Adjoining Grammars (TAGs), that has already participated in several parsing campaigns for French. It was important to be able to compare it with statistical parsers on their native treebank, but also possibly to extend the comparison for other treebanks.

A first necessary step in this direction was a conversion from FRMG's native dependency scheme into FTB's dependency scheme, a tedious task highlighting the differences in design at all levels (segmentation, parts of speech, representation of the syntactic phenomena, etc.). A preliminary evaluation has shown that accuracy is good, but largely below the scores reached by the statistical parsers.

A challenge was then to explore if training on the FTB could be used to improve the accuracy of a symbolic parser like FRMG. However, the main difficulty arises from the fact that FTB's dependency scheme has little in common with FRMG's underlying grammar, and that no reverse conversion from FTB to FRMG structures is available. Such a conversion could be investigated but would surely be difficult to develop. Instead, we tried to exploit directly FTB data, using only very minimal assumptions, nevertheless leading to important gains and results close to those obtained by the statistical parsers. The interest is that the technique should be easily adaptable for training data with different annotation schemes. Furthermore, our motivation was not just to improve the performances on the FTB and for the annotation scheme of FTB, for instance by training a reranker (as often done for domain adaptation), but to exploit the FTB to achieve global improvement over all kinds of corpora and for FRMG native annotation scheme.

Section 2 provides some background about FRMG. We expose in Section 3 how partially supervised learning may be used to improve its performances. Section 4 briefly presents the French TreeBank and several other corpora used for training and evaluation. Evaluation results are presented and discussed in Section 5 with a preliminary analysis of the differences between FRMG and the other statistical parsers.

2 FRMG, a symbolic TAG grammar

FRMG (de La Clergerie, 2005) denotes (a) a French meta-grammar; (b) a TAG grammar (Joshi et al., 1975) generated from the meta-grammar; and (c) a chart-like parser compiled from the grammar. As a parser, FRMG parses DAGs of words, built with SxPipe (Sagot and Boullier, 2008), keeping all potential segmentation ambiguities and with no prior tagging. The parser tries to get *full parses* covering the whole sentence, possibly relaxing some constraints (such as number agreement between a subject and its verb); if not possible, it switches to a *robust mode* looking for a sequence of *partial parses* to cover the sentence.

All answers are returned as shared TAG derivation forests, which are then converted into dependency shared forests, using the anchors of the elementary trees as sources and targets of the dependencies. Some elementary trees being not anchored, pseudo empty words are introduced to serve as source or target nodes. However, in most cases, by a simple transformation, it is possible to reroot all edges related to these pseudo anchors to one of their lexical child.

Finally, the dependency forests are disambiguated using heuristic rules to get a tree. The local *edge rules* assign a positive or negative weight to an edge e, given information provided by e (form/lemma/category/...for the source and target nodes, edge label and type, anchored trees, ...), by neighbouring edges, and, sometimes, by competing edges. A few other *regional rules* assign a weight to a governor node G, given a set of children edges forming a valid derivation from G. The disambiguation algorithm uses dynamic programming techniques to sum the weights and to return the best (possibly non-projective) dependency tree, with maximal weight.

Several conversion schemes may be applied on FRMG's native dependency trees. A recent one returns dependency structures following the annotation scheme used by the dependency version of the French TreeBank and represented using the column-based CONLL format (Nivre et al., 2007). The conversion process relies on a 2-stage transformation system, with constraints on edges used to handle non-local edge propagation, as formalized in (Ribeyre et al., 2012). Figure 1 illustrates the native FRMG's dependency structure (top) and, on the lower side, its conversion to FTB's dependency scheme (bottom). One may observe differences between the two dependency trees, in particular with a (non-local) displacement of the root node. It may be noted that FTB's scheme only considers projective trees, but that the conversion process is not perfect and may return non projective trees, as shown in Figure 1 for the `p_obj` edge.

Let's also mention an older conversion process from FRMG dependency scheme to the EASy/Passage scheme, an hybrid constituency/dependency annotation scheme used for the first French parsing evaluation campaigns (Paroubek et al., 2009). This scheme is based on a set of 6 kinds of chunks and 14 kinds of dependencies.

3 Partially supervised learning

The set of weights attached to the rules may be seen as a statistical model, initially tailored by hand, through trials. It is tempting to use training data, provided by a treebank, and machine learning techniques to improve this model. However, in our case, the "annotation schemes" for the training data (FTB) and for FRMG are distinct. In other words, the training dependency trees cannot be immediately used as oracles as done in most supervised learning approaches, including well-known perceptron ones. Still, even partial information extracted from the training data may help, using partially supervised learning techniques.

Figure 2 shows the resulting process flow. Learning is done, using the disambiguated dependency trees produced by FRMG on training sentences, with (partial) information about the discarded alternatives. The resulting statistical model may then be used to guide disambiguation, and be improved through iterations. Actually, this simple process may be completed with the construction and use of (imperfect) oracles adapted to FRMG. The learning component can produce such an oracle but can also exploit it. Even better, the oracle can be directly used to guide the disam-

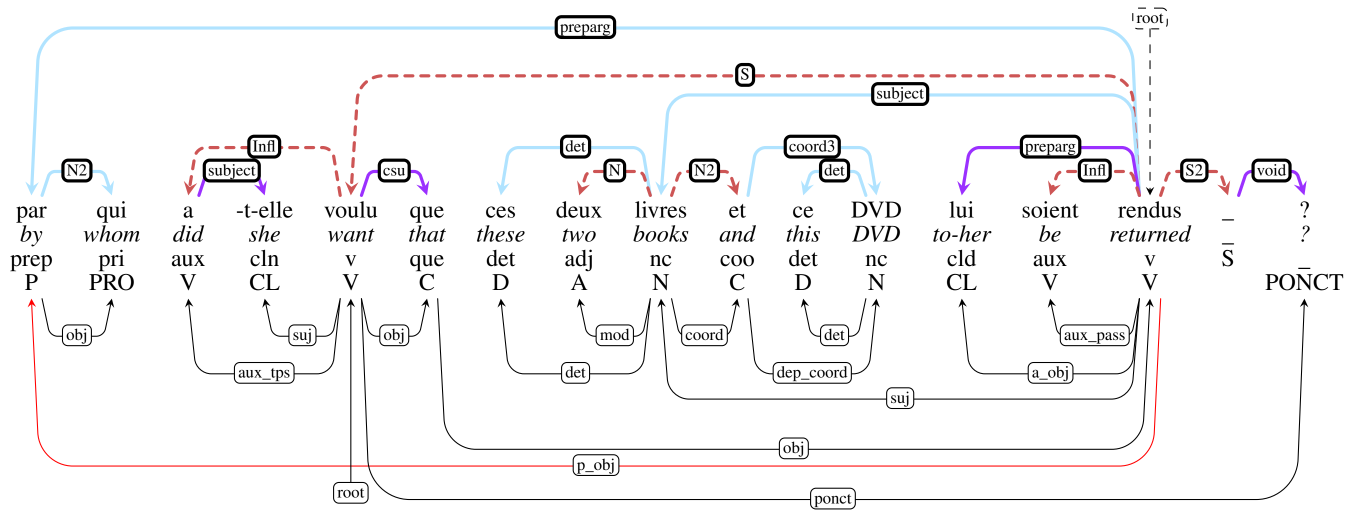


Figure 1: Sample of disambiguated FRMG output, without conversion (top) and with FTB conversion (bottom)

biguation process. Again, by iterating the process, one can hopefully get an excellent oracle for the learning component, useful to get better models.

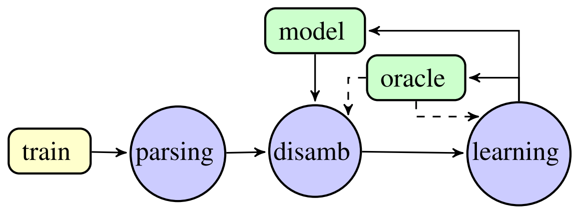


Figure 2: Parsing and partially supervised learning with imperfect oracles

The minimal information we have at the level of a word w is the knowledge that the its incoming dependency d proposed by conversion is correct or not, leading to 4 situations as summarized in Table 1.

	d is correct	d is not correct
selected D	favor r	penalize r'
competitor D'	penalize r'	??

Table 1: Handling weight updates for rules

For the FTB conversion, when d is correct, we can generally assume that the FRMG incoming dependency D for w is also correct and that disambiguation is correct in selecting D. We can then consider than any edge disambiguation rule r applicable on D should then be favored by (slightly) increasing its weight and rules applying on competitors of D should see their weight decrease (to reinforce the non-selection of D).

On the other hand, when d is not correct, we should penalize the rules r applying on D and try to favor some competitor D' of D (and favor the rules r' applying to D'). However, we do not know which competitor should be selected, except in cases where there is only one possible choice. By default, we assume that all competitors have equal chance to be the correct choice and favor/penalize in proportion their rules. If we have n competitors, we can say that it is a bad choice not to keep D' in $\frac{1}{n}$ cases (and should favor rules r') and it is a bad choice to keep D' in $\frac{n-1}{n}$ cases (and should penalize rules r'). So, practically, the dependency D' falling in the problematic case is distributed between the keep/bad case (with weight $\frac{1}{n}$) and the discard/bad case (with weight $\frac{n-1}{n}$). These proportions may be altered if we have more precise information about the competitors, provided by an oracle (as hinted in Figure 2), weights, ranks, or other elements. For instance, if we known that d is not correct but has the right dependency label or the right governor, we use this piece of information to discard some competitors and rerank the remaining ones.

Of course, the update strategy for the problematic case will fail in several occasions. For instance, maybe D is the right FRMG dependency to keep, but the conversion process is incorrect and produces a bad

FTB dependency d. Maybe FRMG is incomplete and has no correct source dependency D. Finally, maybe d (with target word w) derives from some source dependency $D_{w'}$ for some other target word w'. We assume that these cases remain limited and that improving edge selection for the easy cases will then guide edge selection for the more complex cases.

The learning algorithm could be used online, adjusting the weights when processing a sentence. However, we have only implemented an offline version where the weights are updated after considering all training sentences (but discarding some long sentences and sentences with low accuracy scores).

More formally, given the parses for the training sentences, for any edge disambiguation rule r and value tuple $\mathbf{v}$ for a feature template $\mathbf{f}$, we compute the number $n_{r,\mathbf{f}=\mathbf{v}}$ of occurrences of r in context $\mathbf{f} = \mathbf{v}$, and $\text{keep}^{\text{ok}}_{r,\mathbf{f}=\mathbf{v}}$ the number of occurrences where the edge was selected and it was a correct choice. Similarly, but taking into account the above-mentioned redistribution, we compute $\text{discard}^{\text{ok}}_{r,\mathbf{f}=\mathbf{v}}$, $\text{keep}^{\text{bad}}_{r,\mathbf{f}=\mathbf{v}}$, and $\text{discard}^{\text{bad}}_{r,\mathbf{f}=\mathbf{v}}$.

These figures are used to compute an adjustment $\delta_{r,\mathbf{f}=\mathbf{v}}$ added to the base weight w_r of r for context $\mathbf{f} = \mathbf{v}$, using Eq (1), where θ denotes a *temperature*:

$$(1)\quad \delta_{r,\mathbf{f}=\mathbf{v}} = \theta . a_{r,\mathbf{f}=\mathbf{v}} . \begin{cases} \text{discard}^{\text{bad}}_{r,\mathbf{f}=\mathbf{v}} & \text{if } a_{r,\mathbf{f}=\mathbf{v}} > 0 \\ \text{keep}^{\text{bad}}_{r,\mathbf{f}=\mathbf{v}} & \text{otherwise} \end{cases}$$

The $a_{r,\mathbf{f}=\mathbf{v}}$ factor is related to the direction and force of the expected change[1], being positive when selecting an edge thanks to r tends to be a good choice, and negative otherwise (when the edge should rather be discarded), as expressed in the following formula:

$$a_{r,\mathbf{f}=\mathbf{v}} = \frac{\text{keep}^{\text{ok}}}{\text{keep}^{\text{ok}} + \text{keep}^{\text{bad}}} - \frac{\text{discard}^{\text{ok}}}{\text{discard}^{\text{ok}} + \text{discard}^{\text{bad}}}$$

The last factor in Eq (1) is the number of edges whose status (selected or discarded) should ideally change.

The process is iterated, reducing the temperature at each step, and we keep the best run. At each iteration, the edges found to be correctly kept or discarded are added to an oracle for the next iteration.

[1]It may be noted that the interpretation of $a_{r,\mathbf{f}=\mathbf{v}}$ may sometimes be unclear, when both $\text{keep}^{\text{ok}}_{r,\mathbf{f}=\mathbf{v}}$ and $\text{discard}^{\text{ok}}_{r,\mathbf{f}=\mathbf{v}}$ are low (i.e., when neither keeping or discarding the corresponding edges is a good choice). We believe that these cases signal problems in the conversion process or the grammar.

We use standard features such as form, lemma, pos, suffixes, sub-categorization information, morphosyntactic features, anchored TAG trees for words (dependency heads and targets, plus adjacent words); and dependency distances, direction, type, label, and rank for the current dependency and possibly for its parent. For smoothing and out-of-domain adaptation, we add a *cluster* feature attached to forms and extracted from a large raw textual corpus using Brown clustering (Liang, 2005). It may noted that the name of a disambiguation rule may be considered as the value of a `rule` feature. Each feature template includes the label and type for the current FRMG dependency.

It seems possible to extend the proposed learning mechanism to adjust the weight of the regional rules by considering (second-order) features over pairs of adjacent sibling edges (for a same derivation). However, preliminary experiments have shown an explosion of the number of such pairs, and no real gain.

4 The corpora

The learning method was tried on the French TreeBank(Abeillé et al., 2003), a journalistic corpus of 12,351 sentences, annotated in morphology and constituency with the Penn TreeBank format, and then automatically converted into projective dependency trees, represented in the CONLL format (Candito et al., 2010a). For training and benchmarking, the treebank is split into three parts, as summarized in Table 2.

		Cover.	Time (s)	
Corpus	**#sent.**	**(%)**	**Avg**	**Median**
FTB train	9,881	95.9	1.04	0.26
FTB dev	1,235	96.1	0.88	0.30
FTB test	1,235	94.9	0.85	0.30
Sequoia	3,204	95.1	1.53	0.17
EASyDev	3,879	87.2	0.87	0.14

Table 2: General information on FTB and other corpora

To analyze the evolution of the performances, we also consider two other corpora. The **Sequoia** corpus (Candito and Seddah, 2012) also uses the FTB dependency scheme (at least for its version 3), but covers several styles of documents (medical, encyclopedic, journalistic, and transcription of political discourses). The **EASyDev** corpus also covers various styles (journalistic, literacy, medical, mail, speech, ...), but was annotated following the EASy/Passage scheme for evaluation campaigns (Paroubek et al., 2006).

Table 2 shows that coverage (by full parses) is high for all corpora (slightly lower for **EASyDev** because of the mail and speech sub-corpora). Average time per sentence is relatively high but, as suggested by the much lower median times, this is largely due to a few long sentences and due to a large timeout.

5 Results and discussions

	FTB			other corpora	
system	train	dev	test	Sequoia	EASy
init	79.95	80.85	82.08	81.13	65.92
+restr	80.67	81.72	83.01	81.72	66.33
+tuning	86.60	85.98	87.17	84.56	69.23
BKY	–	86.50	86.80	–	–
MALT	–	86.90	87.30	–	–
MST	–	87.50	88.20	–	–

Table 3: Performances of various systems on French data

system	emea-test	ftb-test	loss
BKY (evalb)	80.80	86.00	5.20
FRMG+tuning (LAS)	84.13	87.17	3.04

Table 4: Evolution for an out-of-domain medical corpus

Table 3 shows evaluation results for different versions of FRMG on each corpus. On **FTB** and **Sequoia**, we use Labelled Attachment Scores (LAS) without taking into account punctuation, and, on **EASyDev**, F1-measure on the dependencies[2]. The *init* system corresponds to a baseline version of FRMG with a basic set of rules and hand-tailored weights. The *+restr* version of FRMG adds *restriction rules*, exploiting attachment preferences and word (semantic) similarities extracted from a very large corpus parsed with FRMG, using Harris distributional hypothesis[3]. This version shows that unsurpervised learning methods already improve significantly the performances of a symbolic parser like FRMG for all corpora. The *+tuning* version of FRMG keeps the restriction rules and adds the partially supervised learning method. We observe large improvements on the FTB dev and test parts (between 4 and 5 points), but also on **Sequoia** (almost 3 points) on different styles of documents. We also get similar gains on **EASyDev**, again for a large diversity of styles, and, more interestingly, for a different annotation scheme and evaluation metric.

The bottom part of Table 3 lists the accuracy of 3 statistical parsers on FTB as reported in (Candito et al., 2010b). The Berkeley parser (BKY) is a constituent-based parser whose parses are then converted into FTB dependencies (using the same tool used to convert the FTB). MALT parser is a greedy transition-based parser while MST (*maximum spanning tree*) globally extracts the best dependency tree from all possible ones. We see that FRMG (with tuning) is better than BKY on the test part (but not on the dev part), close to MALT, and still below MST. Clearly, tuning allows FRMG to be more competitive with statistical parsers even on their native treebank.

We do not have results for the 3 statistical parsers on the **Sequoia** corpus. However, (Candito and Seddah, 2012) reports some results for Berkeley parser on constituents for the medical part of **Sequoia**, listed in Table 4. The metrics differ, but we observe a loss of 5.2 for BKY and only of 3.04 for FRMG, which tends to confirm the stability of FRMG across domains, possibly due to the constraints of its underlying linguistically-motivated grammar (even if we observe some over-fitting on FTB).

Figure 3 shows the evolution of accuracy on the 3 components of FTB during the learning iterations. We observe that learning is fast with a very strong increase at the first iteration, and a peak generally reached at iterations 3 or 4. As mentioned in Section 3, rule names may be seen as feature values and it is possible to discard them, using only a single *dummy* rule. This *dummy* edge rule checks nothing on the edges but only acts as a default value for the `rule` feature. However, old experiments showed a LAS on FTB dev of 84.31% keeping only a dummy rule and of 85.00% with all rules, which seems to confirm the (global) pertinence of the hand-crafted disambiguation rules. Indeed, these rules are able to consult additional information (about adjacent edges and alternative edges)

[2]F1-measures on chunks are less informative.

[3]We used a 700Mwords corpus composed of AFP news, French Wikipedia, French Wikisource, etc.. The attachment weights are used for handling PP attachments over verbs, nouns, adjectives, but also for relatives over antecedents, or for filling some roles (subject, object, attribute). Similarities between words are used for handling coordination.

not available through the other features. [4]

As suggested in Figure 2, the oracle built by the learning component on FTB train may be used during disambiguation (on FTB train) by setting a very high weight for the edges in the oracle and a very low weight for the others. The disambiguation process is then strongly encouraged to select the edges of the oracle (when possible). Iterating the process, we reach an accuracy of 89.35% on FTB train, an interesting first step in direction of a FRMG version of the FTB[5].

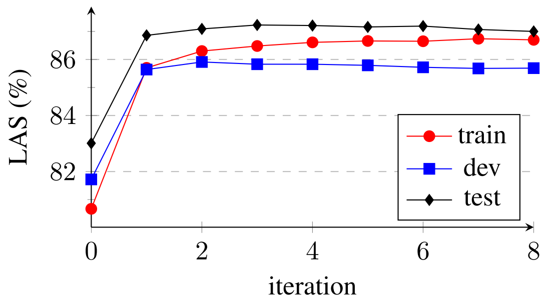


Figure 3: LAS evolution on FTB train per iteration

One reason still explaining the differences between FRMG and the statistical parsers arises from the conversion process to FTB annotation scheme being not perfect. For instance, FRMG and FTB do not use the same list of multi-word expressions, leading to problems of mapping between words and of dependency attachments, in particular for complex prepositions and conjunctions. The segmenter SxPipe also recognizes named entities such as *Communauté européenne* (*European Community*), *5 millions*, or *Mardi prochain* (*next Tuesday*) as single terms whereas FTB adds internal dependencies for these expressions. During the conversion phase, most of the missing dependencies are added leading to an accuracy of 75.38% on the specific dependencies in FTB train (around 3.5% of all dependencies), still largely below the global accuracy (86.6%). There are also 1259 sentences in FTB train (12.7%) where FRMG produces non-projective trees when FTB expects projective ones.[6]

[4]However, it is clear that some disambiguation rules are redundant with the other features and could be discarded.

[5]The problem is that the treebank would have to be regenerated to follow the evolution of FRMG.

[6]It does not mean that so many FRMG trees are non-projective, just that the conversion builds non-projective trees, because of edge movement. A quick investigation has shown that many cases were related to punctuation attachment.

Then, following (McDonald and Nivre, 2007), we tried to compare the performance of FRMG, MST, and MALT with respect to several properties of the dependencies. Figure 4(a) compares the recall and precision of the systems w.r.t. the distance of the dependencies (with, in background, the number of gold dependencies). We observe that all systems have very close recall scores for small distances, then MST is slightly better, and, at long distance, both MST and MALT are better. On the other hand, FRMG has a much better precision than MALT for long distance dependencies. One may note the specific case of null distance dependencies actually corresponding to root nodes, with lower precision for FRMG. This drop corresponds to the extra root nodes added by FRMG in robust mode when covering a sentence with partial parses.

As shown in Figure 4(b), the recall curves w.r.t. dependency depths are relatively close, with FRMG slightly below for intermediate depths and slightly above for large depths. Again, we observe a precision drop for root nodes (depth=0) which disappears when discarding the sentences in robust mode.

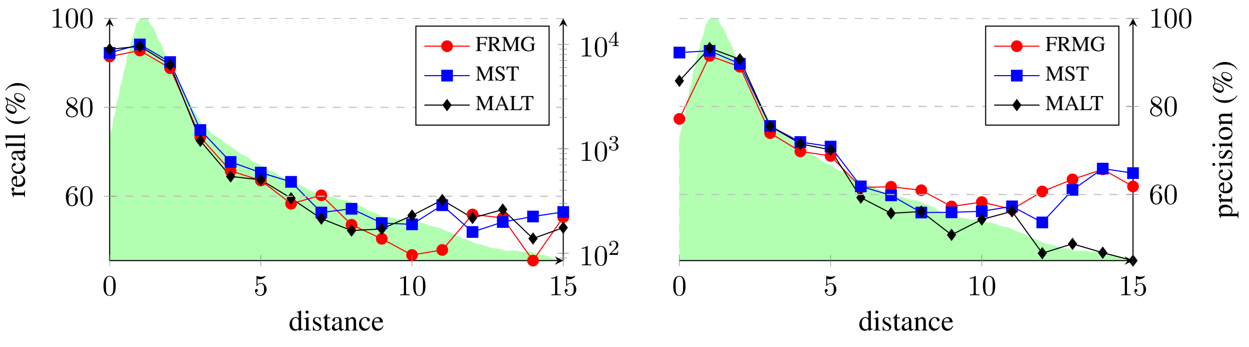
In Figure 4(c), we get again a lower recall for large numbers of sibling edges with, surprisingly, a much higher precision for the same values.

Figure 4(d) compares recall and precision w.r.t. dependency rank[7], with again the lower precision due to the extra root nodes (rank=0) and again a lower recall and higher precision for large absolute ranks.

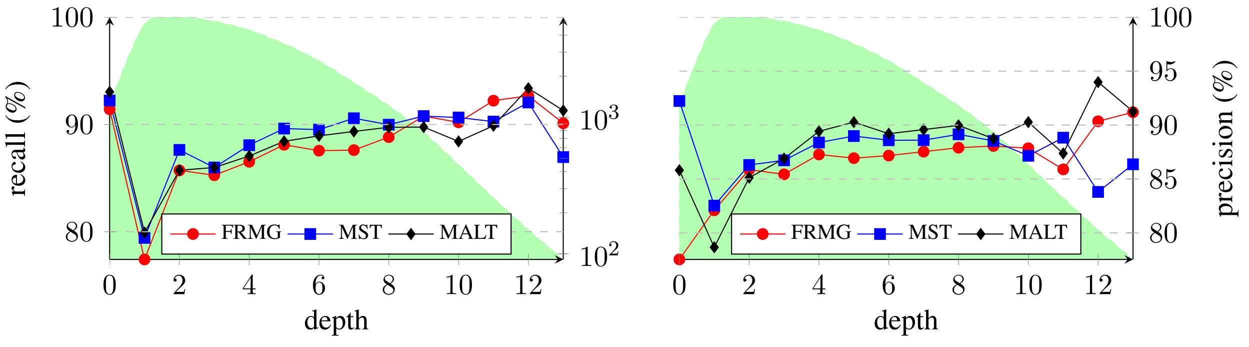
More generally, FRMG tends to behave like MST rather than like MALT. We hypothesize that it reflects than both systems share a more global view of the dependencies, in particular thanks to the domain locality provided by TAGs for FRMG.

Figure 5 shows recall wrt some of the dependency labels. The most striking point is the weak recall for coordination by all systems but, nevertheless, the better score of FRMG. We observe a lower recall of FRMG for some verbal prepositional arguments (`a_obj`, `de_obj`) that may be confused with verb modifiers or attached to a noun or some other verb. Verbal modifiers (`mod`), a category covering many different syntactic phenomena, seem also difficult, partly due to the handling of prepositional attachments. On

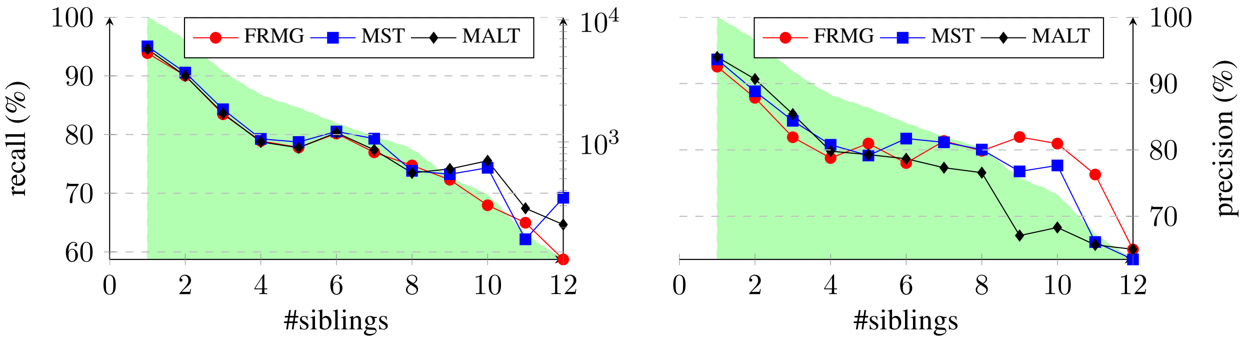
[7]defined as the number of siblings (plus 1) between a dependant and its head, counted positively rightward and negatively leftward.



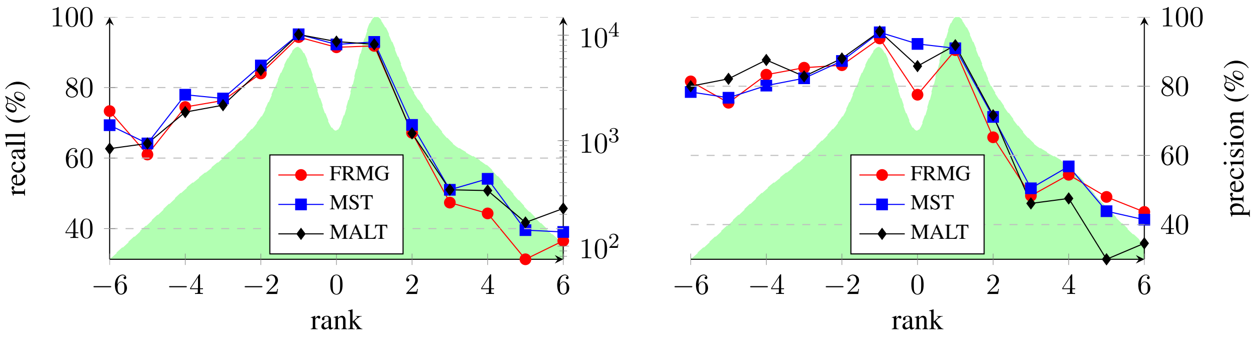
(a) w.r.t. dependency distance



(b) w.r.t. dependency depth



(c) w.r.t. number of siblings



(d) w.r.t. dependency rank

Figure 4: System comparison

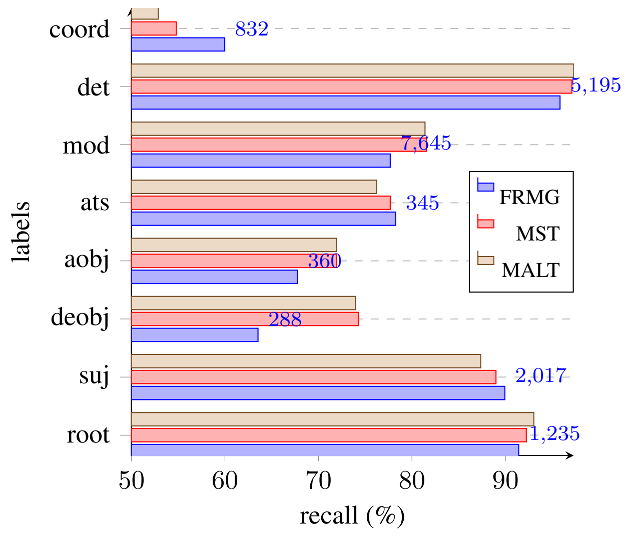


Figure 5: System comparison w.r.t. dependency labels

the other hand, FRMG has a better recall for subjects, possibly because the grammar accepts a large range of positions and realization for subjects.

6 Conclusion

We have presented a new partially supervised learning approach exploiting the information of a training treebank for tuning the disambiguation process of FRMG, a symbolic TAG-based parser. Even considering minimal assumptions for transferring oracle information from the training treebank, we strongly improve accuracy, allowing FRMG to be on par with statistical parsers on their native treebank, namely the French TreeBank. Even if the gains are important, several extensions of the learning algorithm have still to be explored, in particular to build and exploit better oracles, and to incorporate more higher order features, such as sibling features.

The approach explored in this paper, even if tried in the specific context of FRMG, is susceptible to be adapted for other similar contexts, in particular when some imperfect annotation conversion process takes place between a disambiguation process and a training treebank. However, some work remains be done to get a better characterization of the learning algorithm, for instance w.r.t. perceptrons.

We are aware that some of the data collected by the learning algorithm could be used to track problems either in the conversion process or in FRMG grammar (by exploring the cases where neither selecting or discarding an edge seems to be a good choice). We would like to fix these problems, even if most of them seem to have very low frequencies. The conversion process could also be improved by allowing some non-deterministic choices, again controlled by probabilistic features. However, it is not yet clear how we can couple learning for the disambiguation process and learning for the conversion process.

More investigations and comparisons are needed, but some hints suggest that an underlying linguistically-motivated grammar ensures a better robustness w.r.t. document styles and domains. On the other hand, the evaluation shows that the choices made in FRMG to handle lack of full coverage using partial parses should be improved, maybe by using some guiding information provided by a statistical parser to handle the problematic areas in a sentence.

References

Anne Abeillé, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In Anne Abeillé, editor, *Treebanks*. Kluwer, Dordrecht.

Marie Candito and Djamé Seddah. 2012. Le corpus sequoia: annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. In *TALN 2012-19e conférence sur le Traitement Automatique des Langues Naturelles*.

Marie Candito, Benoît Crabbé, and Pascal Denis. 2010a. Statistical french dependency parsing: treebank conversion and first results. In *Proceedings of the 7th Language Resources and Evaluation Conference (LREC'10)*, La Valette, Malte.

Marie Candito, Joakim Nivre, Pascal Denis, and Enrique Henestroza Anguiano. 2010b. Benchmarking of statistical dependency parsers for french. In *Proceedings of COLING'2010 (poster session)*, Beijing, China.

Éric de La Clergerie. 2005. From metagrammars to factorized TAG/TIG parsers. In *Proceedings of IWPT'05 (poster)*, pages 190–191, Vancouver, Canada.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.

Aravind K. Joshi, Leon Levy, and Makoto Takahashi. 1975. Tree Adjunct Grammars. *Journal of Computer and System Science 10*, 10(1):136–163.

Percy Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

Ryan T. McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *EMNLP-CoNLL*, pages 122–131.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *The CoNLL 2007 shared task on dependency parsing*.

Patrick Paroubek, Isabelle Robba, Anne Vilnat, and Christelle Ayache. 2006. Data, Annotations and Measures in EASy, the Evaluation Campaign for Parsers of French. In *Proceedings of the 5th international conference on Language Resources and Evaluation (LREC'06)*, Gênes, Italie.

Patrick Paroubek, Éric Villemonte de la Clergerie, Sylvain Loiseau, Anne Vilnat, and Gil Francopoulo. 2009. The PASSAGE syntactic representation. In *7th International Workshop on Treebanks and Linguistic Theories (TLT7)*, Groningen, January.

Corentin Ribeyre, Djamé Seddah, and Éric Villemonte De La Clergerie. 2012. A Linguistically-motivated 2-stage Tree to Graph Transformation. In Chung-Hye Han and Giorgio Satta, editors, *TAG+11 - The 11th International Workshop on Tree Adjoining Grammars and Related Formalisms - 2012*, Paris, France. INRIA.

K. Sagae, Y. Miyao, T. Matsuzaki, and J. Tsujii. 2008. Challenges in mapping of syntactic representations for framework-independent parser evaluation. In *Proceedings of the Workshop on Automated Syntatic Annotations for Interoperable Language Resources at the First International Conference on Global Interoperability for Language Resources (ICGL'08)*, Hong-Kong, January.

Benoît Sagot and Pierre Boullier. 2008. SXPIPE 2 : architecture pour le traitement présyntaxique de corpus bruts. *Traitement Automatique des Langues (T.A.L.)*, 49(2):155–188.