Statistical French dependency parsing: treebank conversion and first results

Marie Candito, Benoît Crabbé, Pascal Denis

Statistical French dependency parsing: treebank conversion and first results

Marie Candito, Benoît Crabbé, Pascal Denis

Equipe-Projet Alpage
INRIA & Université Paris 7
30 rue du Château des Rentiers, 75013 Paris, France
{marie.candito,bcrabbe@linguist.jussieu.fr}, pascal.denis@inria.fr

Abstract

We first describe the automatic conversion of the French Treebank (Abeillé and Barrier, 2004), a constituency treebank, into typed projective dependency trees. In order to evaluate the overall quality of the resulting dependency treebank, and to quantify the cases where the projectivity constraint leads to wrong dependencies, we compare a subset of the converted treebank to manually validated dependency trees. We then compare the performance of two treebank-trained parsers that output typed dependency parses. The first parser is the MST parser (Mcdonald et al., 2006), which we directly train on dependency trees. The second parser is a combination of the Berkeley parser (Petrov et al., 2006) and a functional role labeler: trained on the original constituency treebank, the Berkeley parser first outputs constituency trees, which are then labeled with functional roles, and then converted into dependency trees. We found that used in combination with a high-accuracy French POS tagger, the MST parser performs a little better for unlabeled dependencies (UAS=90.3% versus 89.6%), and better for labeled dependencies (LAS=87.6% versus 85.6%).

1. Introduction

The task of converting a constituency treebank into dependencies is interesting both for its result, a dependency treebank, and for the linguistic questions raised by the conversion itself.

Efficient dependency parsing algorithms now exist, that achieve state-of-the-art results in quadratic or even linear time, for a large number of languages (cf. the CoNLL 2006 and 2007 tasks on multilingual dependency parsing (Nivre et al., 2007)). Experiments on French could not be performed at that time due to the absence of a training resource[1]. And because dependency trees are closer to predicate-argument structures, they are often presented as more suitable syntactic representations for various NLP tasks such as information extraction or question answering, so there might be a practical advantage of having a dependency treebank. Further, a surface annotation scheme for French allows to compare parsing performances between a wider range of parsers, namely both constituency parsers and dependency parsers, because the output of constituency parsers can be converted into dependencies using the conversion tool we describe in this paper.

Moreover, the conversion itself is interesting from a linguistic point of view, as it renders explicit some assumptions underlying syntagmatic structures. Within a constituent tree, dependencies are assumed to hold between a phrase's head and its siblings, except for non-local dependents. But because the FTB does not contain special marking for non-local dependencies, so to list and quantify these cases is interesting to evaluate the amount of missing information. Another useful explicitation concerns how the types of the dependencies can be infered from the syntagmatic structure, in the case they're not explicitely stated in the constituent trees.

We describe in section 2. the conversion into surface dependencies and its evaluation against a small set of manually corrected dependency trees. We then describe in sections 3. and 4. the two parsing architectures that we tested for obtaining typed dependency parses. Then, we present and discuss experiments in section 6 and related work in section 7.

2. Converting the French Treebank into typed surface dependencies

The French Treebank (Abeillé and Barrier, 2004) (hereafter FTB) is made of 12,531 sentences from the *Le Monde* newspaper, annotated for morphology and phrase-structure. Further, some of the nodes are labeled with a grammatical function. This is necessary because a given structural position may correspond to

[1]The EAsy project (Paroubek et al., 2005) has released an annotated corpus for French, containing approximately 400000 words of texts of various domains, such as newspaper, but also literary, medical and oral texts. But the annotation scheme mixes chunks and relations, that cannot be converted easily into full surface dependency trees usable for the training a statistical dependency parser.

different grammatical relations. For instance Figure 2 shows on the left an FTB-style constituency tree. The postverbal NP is here a modifier, hence the functional annotation MOD, but the same tree shape appears in the more frequent case of direct object postverbal NPs. These grammatical functions are made explicit for dependents of verbs, or more precisely for clitics and for non-coordinate phrasal nodes that appear as siblings of a verb[2]. (Abeillé and Barrier, 2004) consider that the grammatical functions for dependents of other categories is encoded in the shape of the trees. Note though that this is not the case for the argument/adjunct distinction for dependents of nouns or adjectives, which is not encoded in the syntagmatic structure. We had to keep this distinction underspecified in our output dependency trees for prepositional dependents of non-verbal heads.

We describe first some automatic preprocessing of the FTB, and second the automatic procedure for outputing projective dependency trees. Then we present an evaluation of the converted treebank, and discuss linguistic cases for which the automatic procedure fails to recover the desired dependencies.

2.1. Preprocessing of the FTB

Undoing some compounds In the original treebank, 17% of the tokens belong to a compound. Compounds range from very frozen multi-word expressions like *y compris* (literally *there included*, meaning *including*) to named entities. They include syntactically regular compounds with compositional semantics, such as *loi agraire* (*land law*), that are encoded as compounds because of a non-free lexical selection. In most of the experiments that use the FTB, each compound is merged into a single token : `(N (N loi) (A agraire))` is merged as `(N loi_agraire)`. But this supposes a perfect compound recognition prior to parsing, which is not realistic. As a tradeoff, we created a new instance of the treebank (hereafter FTB-UC), where *syntactically regular* compounds are "undone", leaving it to a semantic analysis to recover maybe semantically non-compositional units. For instance, Figure 1 shows the "undoing" of the original compound *Union économique et monétaire* (*monetary and economic union*).

[2]More precisely as siblings of a verbal nucleus node, that includes clitics and auxiliaries. Grammatical functions are missing dependents of past participles employed without an auxiliary, for instance in adnominal participials. Functions are missing too on dependents of verbs that do not project a phrase, such as non-modified adverbs. Though most of them are modifiers, some are subcategorized locative complements.

Within the whole treebank, $3,072$ syntactically regular distinct compounds are "undone" out of a total of $6,125$ distinct compounds. The remaining compounds are merged into a single token. This leads to a total of 350951 tokens in the FTB-UC, while the number of tokens in the original FTB when all compounds are merged is 339522.

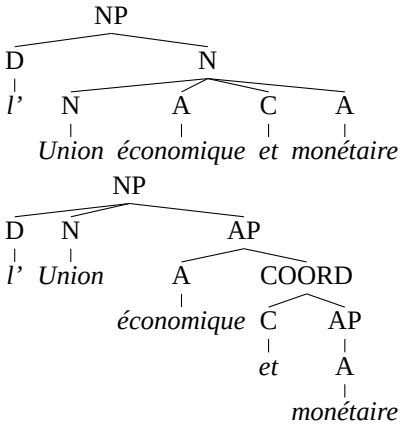


Figure 1: A NP with a compound in the original treebank (up) changed into a regular structure with simple words in the FTB-UC (bottom).

"Raising" of complementizers and prepositions We made some linguistic choices that sometimes contradict the flat annotation scheme of the treebank. For instance, in the FTB, a preposition projects a PP if it takes an NP complement, but is included in an infinitival VP if it introduces an infinitival complement.

In order to uniformly treat prepositions and complementizers as heads (whether semantically empty or not), we automatically transformed the phrase-structure trees, in a reversible manner, using the stanford tsurgeon tool (Levy and Andrew, 2006).

Tagset and features The dependency trees are ultimately output in CoNLL format[3]. For the coarse-grained part-of-speech column, the tagset is the original coarse-grained category of the FTB. For the fine-grained part-of-speech column, we use the 28 POS tagset described by (Crabbé and Candito, 2008), where tags are a combination of the coarse-grained category, verbal mood information, and some other distinctions, such as proper versus common nouns, wh-feature etc... The morphological features present in the original FTB are copied in the CoNLL feature and lemma columns.

[3]`http://nextens.uvt.nl/~conll/`

2.2. Conversion procedure

The conversion procedure is based on the classic technique of head propagation rules, first proposed for English in (Magerman, 1995). With that technique, ouptut dependency trees are necessarily projective, and extracted dependencies are necessarily local to a phrase. This leads to wrong dependencies in the case of non-locality (e.g. long distance extraction) : non local dependents receive a wrong governor. We could not automatically correct them because the treebank does not contain traces, nor any explicit marking for non-local dependencies.
We used a four-stage procedure, that outputs projective surface dependency trees : each token has exactly one governor, except the root[4] :

(i) The preprocessing described in section 2.1. is applied to the treebank.

(ii) 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. For French we used an enhanced version of (Arun and Keller, 2005) rules designed for the FTB annotation scheme.

(iii) Bilexical dependencies are extracted, using the lexical heads added at stage (ii). If the constituent node for the dependent bears a functional label, it is used as the label of the dependency.

(iv) Because the original treebank has functional labels for syntagmatic dependents of verbs only, there remain unlabeled dependencies, that are then labeled using heuristics. Hence the heuristics apply to dependents of non-verbs, dependents of verbs that do not project a phrase, coordinated phrases, and dependents of adnominal participials. For instance in *vêtements achetés par les Français* (*clothes bought by French people*) the PP *par les Français* does not bear a function. Heuristics tag it as a P_OBJ of the past participle, namely a prepositional argument.

2.3. Conversion evaluation and dependency annotation scheme

The annotation scheme of the converted treebank results from the major part on the linguistic choices underlying the constituents in the FTB. Yet some cases require specific choices, that are not totally induced by phrase-structure:

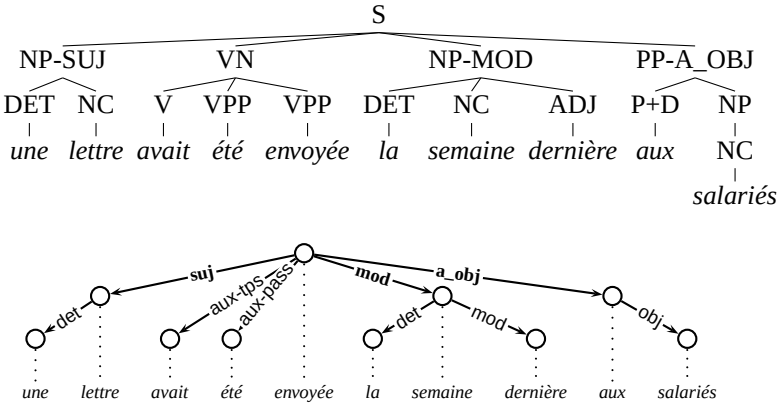
- Tense, passive and causative auxiliairies are treated as dependents of the past participle / infinitive they introduce.
- Prepositions and complementizers, whether semantically empty or not, are systematically treated as the head of the complement they introduce.
- For coordinated structures, the first conjunct is taken as the head.
- As we previously mentioned, the FTB does provide the grammatical functions of dependents of non-verbal heads. In the case of sentential or prepositional dependents, we use an underspecified *dep* label. For adverbial or adjectival dependents, they are encoded as modifiers.

As we stressed in section 2.2., the conversion procedure outputs projective dependency trees only, that are necessarily wrong in the case of non-local dependencies. In order to quantify these cases and to evaluate the overall quality of the conversion, we manually corrected a mini gold corpus (hereafter the FTB-120) for the first 120 sentences of the treebank (approximately 3000 tokens). We simultaneously designed a surface dependency annotation scheme, in which we precise the target annotation for some specific constructions, even though the current automatic procedure is known to fail to conform to this scheme. In particular, we have isolated the following cases of non-projectivity[5]:

- Extraction out of a sentence;
- Extraction out of an NP : relative pronoun *dont*. In the case of an extraction out a postverbal NP, the resulting dependency structure is non projective. For instance in *Lyonnaise Espana, dont le groupe français ne détiendra plus que 51 pourcent* (*Lyonnaise Espana, of-which the French group will not control more than 51 percent*), the relative pronoun depends on the noun *pourcent*.
- Extraction out of an NP : clitic pronoun *en*. For instance in *afin d'en améliorer l'efficacité* (*in order to of-it improve the efficiency*), the clitic *en* (*"of-it"*) depends on *efficacité*.

[4] As is usual for dependency treebanks, each tree contains an additional dummy root node, that cannot be the dependent of another node, so that it can be formally said that each sentence token has exactly one governor.

[5] A full description of the annotation scheme is available online at `http://www.linguist.univ-paris-diderot.fr/~mcandito/Rech/FTBDeps/index.html`.



Literally "*A letter had been sent the week before to the employees*"

Figure 2: An example of input tree of the FTB (up), and the resulting dependency tree (bottom)

- Further, we isolated a few other cases, such as comparatives and consecutives. In patterns like *plus/moins X que Y* (*more/less X than Y*) the complementizer depends on the comparative adverb (*more* or /em less etc...).

Then we were able to evaluate the conversion procedure on the FTB-120, using the unlabeled (respectively labeled) attachment scores (UAS/LAS), namely the percentage of non-punctuation tokens that are attached to the correct governor (respectively to the correct governor with the correct dependency label. When comparing the automatically converted trees to the manually corrected dependency trees for the FTB-120, we obtain a UAS of 98.78%, and a LAS of 98%. Hence, we can conclude the FTB-120 does not contain more than 1.22% non-projective links[6]. The additional labeling errors appear in cases where the labeling heuristics are too coarse.

The resulting converted treebank, and the manually corrected extract are available, in COnLL format, upon prior obtention of the FTB licence.

3. Constituency-dependency parsing

Our first dependency parser (hereafter BKY+FLABELER) is built in a way that mirrors and exploits the treebank conversion described above. That is, we first train a phrase-structure parser on an instance of the constituency-based FTB wherein the functional labels have been omitted. The parser that was used is the Berkeley parser (Petrov et al., 2006). Second, we use a discriminative classifier to automatically assign labels to the constituency nodes. And finally, we apply the conversion rules (i)-(iv) described in section 2.[7]

For the BKY algorithm uses the fact that some symbol splits are known to help PCFG learning. (Matsuzaki et al., 2005) proposed to automatize the splits, and (Petrov et al., 2006) proposed to score the splits in order to retain only the most beneficial ones, and keep the grammar size manageable. The algorithm starts with a binarized PCFG, and then performs split/merge/smooth cycles. It splits each symbol in two by adding latent variables to it, and uses EM to learn probabilities for the split symbols. It then merges back the less beneficial splits, that are scored using the loss in the likelihood of the treebank induced by the merge. (Seddah et al., 2009) showed that BKY outperforms various constituency statistical parsers for French.

We focus now on the functional labeling procedure. It relies on a supervised learning algorithm that allows for more expressivity than one can expect from a PCFG or its derivatives. The labeler concentrates on the 8 verbal-predicate functions annotated in the FTB (see left schema in Figure 2).The labelling task consists in predicting a sequence of n functional labels $f_1 \dots f_n$ given a governor g and n dependents $d_1 \dots d_n$ of a predicate. As a first approximation, we break down this task as a series of local classifications, in effect treating each dependent labeling as an independent event.

[6] We analyzed the 34 tokens that receive the wrong head. For 18 of them, the correct governor is non-local, for 8 of them, the head propagation rules are not precise enough to get the correct head. For the remaining 8 cases, the original FTB annotation is mistaken.

[7] (Candito et al., 2009) report that this sequential scheme achieves better performance than an integrated scheme wherein the BKY parser is trained on data where the functional labels are part of the grammatical symbols, due to data sparseness.

For the purpose of classification, we use a maximum entropy (MaxEnt) based classifier. The features used attempt to capture information related to bilexical dependencies and sentence configuration. As often in MaxEnt, all our features are binary indicator functions. The features are described in more detail in Table 1 and their extraction is illustrated graphically in Figure 3.

Features W_D, W_H, C_D, C_H, C_{CH}, and W_{CH} aim to capture bilexical dependencies between the head word and the dependent word, while including some redundancy like syntactic categories in order to count objects of different levels of granularity. This may be viewed as a counterpart to smoothing procedures used in lexicalised phrase-based parsers (Collins, 1999; Charniak, 2000).

Instead of entire word forms for W_D and W_H, we use stems containing only the first four characters of each word as a proper value. Hapax wordforms in the training corpus are replaced by a dummy token, allowing to deal with unknown words at classification by converting them to this dummy token. Finally, note that the continuous features $dist$ and $span$ have been split into buckets in order to reduce data sparseness.

By contrast, features C_P, LC_D, RC_D, $dist$, $span$, M_H, $rank$, wh, rel, $etre$, and inv target configurational information. For instance, the further a dependent is from its head, the more likely it is to be a modifier. Features such as LC_D and RC_D try to capture whether the dependent is surrounded partially or totally by punctuation marks. The intuition is that a punctuation mark between the head and the dependent indicates that the dependent is likely to be a modifier. Mood gives a penalty to subject assignment within infinitive or imperative clauses. The *être* auxiliary approximates the detection of passive clauses, hence providing a preference for assigning a `P-OBJ`[8] label to prepositional dependents.

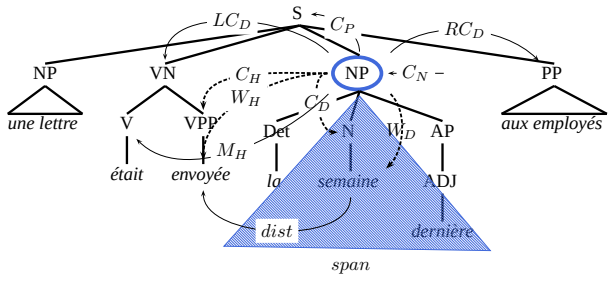


Figure 3: Main feature extraction patterns

[8]That is the function of a complement dependent introduced by a preposition. In the passive case, the complement is introduced by the preposition *par* (*by*).

4. Direct dependency parsing

Our second parser is the MST parser (McDonald et al., 2005), which is directly trained on the converted treebank. It is so-called because it treats the parsing problem as the search for the highest scoring maximum spanning tree in the complete directed graph over the input sentence.

(McDonald et al., 2005) proposed the *first-order* MST model, which uses an edge-based factorization: the score of a dependency tree is computed as the sum of the scores for all the tree edges. These scores are obtained as an inner product of a high dimension feature representation for the edge and a weight vector. The MST formulation has efficient parsing algorithms for both projective and non-projective structures: the Eisner (1996) algorithm for the projective case and the Chu-Liu-Edmonds (Chu and Liu, 1965; Edmonds, 1967) algorithm for the non-projective case. These algorithms have cubic and quadratic parsing times, respectively.

With *first-order* factorization, features are solely defined over single edges in the graph. These include parent and child words, their POS tags, and the POS tags of surrounding words and those of words between the child and the parent, as well as the direction and distance from the parent to the child. (McDonald and Pereira, 2006) complexified this model by using *second-order* factorization, in which the score of a dependency tree is computed as the sum of the scores of adjacent edge pairs (but only pairs where the dependents appear on the same side of the governor). The second-order features, which in fact subsume the first-order features, are built from the conjunctions of word and POS identity predicates for a parent and two siblings. See (McDonald and Pereira, 2006) for details.

Weight learning is performed using an extension of the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2001), an online learning algorithm particularly well-suited to structured classification problems. This algorithm is related to the averaged perceptron of Collins (2002), but it uses a different kind of update. Informally, MIRA's update works by finding the vector that results in the smallest norm update wrt to the current vector (i.e., it is conversative) while ensuring that the correct tree outscores the predicted tree by a margin proportional to the loss (computed in terms of the number of wrong dependencies) between the two trees (i.e., it is large-margin).

In order to produce *labeled* dependencies, (Mcdonald et al., 2006) also uses a two-step approach. That is, the MST parser is used to produce an unlabeled tree whose edges are then labeled using a separate classifier (which can incorporate features defined over the

Feature	Description
C_N	Syntactic category of the syntagmatic node to classify (D)
W_D	Dependent stemmed wordform (i.e. of the lexical head of D)
W_H	Verbal head stemmed wordform
C_D	Part-of-speech of the dependent
C_H	Part-of-speech of the verbal head
$dist$	Number of words between the dependent and the verbal head
$span$	number of words in the yield of the tree dominated by D
C_P	Syntactic category of the parent node (immediately dominating D)
LC_D	Syntactic category of the left adjacent node to D
RC_D	Syntactic category of the right adjacent node to D
C_{CH}	Syntactic category of the cohead (or `None` if it does not exist)
W_{CH}	Wordform of the cohead (or `None` if it does not exist)
M_H	Mood of the verbal head
$rank$	Index of the dependent d_i within the sequence $d_0, \ldots, d_n$
wh	The sentence is a question
rel	The clause is relative
$etre$	The verbal head's auxiliary is *être* (*to be*)
inv	The verbal head is built with a clitic inversion

Table 1: Main features used by the classifier

entire dependency tree, and not just first- and second-order edges). The labeling task is here performed as a sequence labeling task and uses a first-order Markov factorization (i.e., scores are computed on pairs of adjacent edges).

Finally, note that MST parser has been shown to provide state-of-the-art dependency parsing performance for various languages (including Arabic, Chinese, Czech, English, German, Japanese, and Turkish). Crucially, the MST parser has never been used for French.

5. Experiments and results

Protocol For our experiments, we used the split of the FTB described in (Crabbé and Candito, 2008)[9]. The metric used is the labeled (resp. unlabeled) attachment score (LAS/UAS), ignoring punctuation tokens, namely the percentage of tokens that are attached to the correct governor with correct dependency label (resp. the correct governor).[10]

BKY+FLABELER **settings** We used the first release of the Berkeley parser (Petrov et al., 2006), tuned for French unknown words by (Crabbé and Candito, 2008). The parameters of the MaxEnt functional labelling classifier were computed with the Megam package.[11]

MST settings We used the freely available implementation[12] of the parser described in (Mcdonald et al., 2006). We experimented with the first- and second-order models, with the default features. We used 1-best projective decoding, with the default number of iterations (i.e., 10); punctuation were not included in the loss calculation. As noted, this implementation also uses a two-step approach: unlabeled dependency parsing, and then labeling via arc classification.

To provide POS tags to the MST parser, we used the MElt tagger (Denis and Sagot, 2009). This tagger uses a Maximum Entropy Markov model that has been augmented with information from a large-scale external morphological dictionary.[13] The tagger was trained on the training set to provide POS tags for the development and test sets, and we used 10-way jackknifing to generate tags for the training set.

Results Performance scores on the test set are given in table 2.[14] The MElt +MST architecture slightly outperforms BKY for the unlabeled attachment score

[9]The first $1,235$ sentences as test set, the next $1,235$ sentences as development set, and the remaining as training set.

[10]We used the official CoNLL-2007 scoring script, available from `http://depparse.uvt.nl/depparse-wiki/SoftwarePage`.

[11]`http://www.cs.utah.edu/~hal/megam/`

[12]`http://sourceforge.net/projects/mstparser/`

[13](Denis and Sagot, 2009) report a tagging accuracy of 97.7 (90.1 on unknown words) on the test set of FTB-UC, using the same split as we did (and including punctuation).

[14]Results for the BKY are lower here than those reported in (Candito and Crabbé, 2009) for two reasons. First, contrary to a habit in phrase-structure statistical parsing, the results we give here are for all test sentences, irrespective of their length. Second, (Candito and Crabbé, 2009) use the development set as validation set for the BKY algorithm.

Parser	LAS	UAS	Tagging Acc.
BKY+FLABELER	85.55	89.63	96.97
MElt +MST 1	86.96	89.63	97.3
MElt +MST 2	87.58	90.28	97.3

Table 2: Tagging accuracy, Labeled and Unlabeled attachment scores, without punctuation. MST 1 and MST 2 stand for the MST parser with first- and second-model, respectively.

(by 0.65%[15]). Also, logically, the sequential functional labeler used in MST outperforms the point-wise labeler implemented in BKY+FLABELER, leading to a greater gap for LAS between the two architectures ($+2.02\%$). MElt +MST also provides better tagging accuracy, which is somewhat surprising given that BKY+FLABELER performs tagging as part of parsing (and has therefore access to more context). This better performance is likely to come from the high accuracy of the MElt tagger on unknown words. Indeed, BKY achieves only a 82.56% tagging accuracy for the unknown words in the development set (5.96% of the tokens), whereas MElt achieves 90.01%.
For comparison, note that the average UAS and LAS scores for MST during the 2007 CoNLL multilingual shared task were 87.0 and 80.8, respectively.

6. Related work

Treebank conversion into dependencies using head propagation rules has been performed for various constituency treebanks (for instance for the Penn Treebank (Yamada and Matsumoto, 2003), (de Marneffe et al., 2006)). (Johansson and Nugues, 2007) provide a more complex conversion procedure of the Penn Treebank, that makes use of a richer set of edge labels, and that produces potentially non-projective dependencies, by using the traces for non-local dependencies. The resulting treebank is more complex, but these authors show it is better suited for semantic tasks, such as semantic role labeling.
Previous work on statistical dependency parsing for French can be found in (Nasr, 2006), who report results for a supertagging + probabilistic dependency parser architecture. The results seem lower (p.151, LAS = 74%, including punctuation). (Schluter and van Genabith, 2007) aim at learning LFG structures for French. Their results cannot be compared to ours because they use a modified subset of the FTB, and they evaluate dependencies appearing in f-structures, which are deeper and hence more difficult to obtain than surface dependencies.

[15]The score differences between MST 2 and BKY+FLABELER are statistically significant ($p < 0.01$ using a chi-square test). Also note that scores for other languages were given without punctuation for the 2006 CoNLL task, but with punctuation for 2007 shared task. To compare with 2007 scores for other languages, here are the UAS scores with punctuation tokens : BKY+FLABELER: 86.85%, MST 1: 86.62%, MST 2: 87.52%.

7. Conclusion

The FTB converted into dependencies is available and suitable for training statistical parsers. We can thus compare, using the same evaluation protocol, a constituent-based parser and a genuine statistical dependency parser, which is usually difficult. The dependency-based MST parser, coupled to the high accuracy MElt tagger, gives slightly better results, both for unlabeled and labeled dependency accuracies.

Acknowledgements

We thank Joakim Nivre for useful discussions, and suggestions on previous versions of the converted treebank. We also thank the LREC reviewers for their comments.
This work was supported by the French National Research Agency (SEQUOIA project ANR-08-EMER-013).

8. References

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

Abhishek Arun and Frank Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: The case of french. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 306–313, Ann Arbor, MI.

Marie Candito and Benoît Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 138–141, Paris, France, October. Association for Computational Linguistics.

Marie Candito, Benoit Crabbé, Pascal Denis, and François Guérin. 2009. Analyse syntaxique du français : des constituants aux dépendances. In *Proc. of TALN'09*, Senlis, France.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st Annual Meeting of the North American Chapter of the ACL (NAACL)*, Seattle.

Y. J. Chu and T. H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400.

Michael Collins. 1999. *Head Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *EMNLP-2002*.

Benoit Crabbé and Marie Candito. 2008. Expériences d'analyse syntaxique statistique du français. In *Proc. of TALN'08*, pages 45–54, Senlis, France.

Koby Crammer and Yoram Singer. 2001. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. of LREC'06*, Genova, Italy.

Pascal Denis and Benoît Sagot. 2009. Coupling an annotated corpus and a morphosyntactic lexicon for state-of-the-art pos tagging with less human effort. In *Proc. of PACLIC*, Hong Kong, China.

Jack R. Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240.

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 340–345, Copenhagen, August.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proceedings of NODALIDA 2007*, Tartu, Estonia.

Roger Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *Proc. of LREC'06*, Geneva, Italy.

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

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic cfg with latent annotations. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 75–82.

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

Ryan T. McDonald, Koby Crammer, and Fernando C. N. Pereira. 2005. Online large-margin training of dependency parsers. In *Proc. of ACL'05*, Ann Arbor, USA.

Ryan Mcdonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proc. of CoNLL'06*, New York, USA.

Alexis Nasr. 2006. Grammaires de dépendances génératives probabilistes. modèle théorique et application à un corpus arboré du français. *Traitement Automatique des Langues*, 46(1).

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 *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June. Association for Computational Linguistics.

Patrick Paroubek, Louis-Gabriel Pouillot, Isabelle Robba, and Anne Vilnat. 2005. Easy : campagne d'évaluation des analyseurs syntaxiques. In *Proceedings of TALN'05, EASy workshop : campagne d'évaluation des analyseurs syntaxiques*, Dourdan.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, July. Association for Computational Linguistics.

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

Djamé Seddah, Marie Candito, and Benoit Crabbé. 2009. Cross parser evaluation and tagset variation: a french treebank study. In *Proceedings of the 11th Internation Conference on Parsing Technologies (IWPT'09)*, pages 150–161, Paris, France, October.

H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *The 8th International Workshop of Parsing Technologies (IWPT2003)*.