Error mining in parsing results

Benoît Sagot and Éric de la Clergerie
Projet ATOLL - INRIA
Domaine de Voluceau, B.P. 105
78153 Le Chesnay Cedex, France
`{benoit.sagot,eric.de_la_clergerie}@inria.fr`

Abstract

We introduce an error mining technique for automatically detecting errors in resources that are used in parsing systems. We applied this technique on parsing results produced on several million words by two distinct parsing systems, which share the syntactic lexicon and the pre-parsing processing chain. We were thus able to identify missing and erroneous information in these resources.

1 Introduction

Natural language parsing is a hard task, partly because of the complexity and the volume of information that have to be taken into account about words and syntactic constructions. However, it is necessary to have access to such information, stored in resources such as lexica and grammars, and to try and minimize the amount of missing and erroneous information in these resources. To achieve this, the use of these resources at a large-scale in parsers is a very promising approach (van Noord, 2004), and in particular the analysis of situations that lead to a parsing failure: one can learn from one's own mistakes.

We introduce a probabilistic model that allows to identify forms and form bigrams that may be the source of errors, thanks to a corpus of parsed sentences. In order to facilitate the exploitation of forms and form bigrams detected by the model, and in particular to identify causes of errors, we have developed a visualization environment. The whole system has been tested on parsing results produced for several multi-million-word corpora and with two different parsers for French, namely SxLFG and FRMG.

However, the error mining technique which is the topic of this paper is fully system- and language-independent. It could be applied without any change on parsing results produced by any system working on any language. The only information that is needed is a boolean value for each sentence which indicates if it has been successfully parsed or not.

2 Principles

2.1 General idea

The idea we implemented is inspired from (van Noord, 2004). In order to identify missing and erroneous information in a parsing system, one can analyze a large corpus and study with statistical tools what differentiates sentences for which parsing succeeded from sentences for which it failed.

The simplest application of this idea is to look for forms, called *suspicious forms*, that are found more frequently in sentences that could not be parsed. This is what van Noord (2004) does, without trying to identify a suspicious form in any sentence whose parsing failed, and thus without taking into account the fact that there is (at least) one cause of error in each unparsable sentence.[1] On the contrary, we will look, in each sentence on which parsing failed, for the form that has the highest probability of being the cause of this failure: it is the *main suspect* of the sentence. This form may be incorrectly or only partially described in the lexicon, it may take part in constructions that are not described in the grammar, or it may exemplify imperfections of the pre-syntactic processing chain. This idea can be easily extended to sequences of forms, which is what we do by tak-

[1] Indeed, he defines the suspicion rate of a form f as the rate of unparsable sentences among sentences that contain f.

Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 329–336, Sydney, July 2006. ©2006 Association for Computational Linguistics

ing form bigrams into account, but also to lemmas (or sequences of lemmas).

2.2 Form-level probabilistic model

We suppose that the corpus is split in sentences, sentences being segmented in forms. We denote by s_i the i-th sentence. We denote by $o_{i,j}$, $(1 \leq j \leq |s_i|)$ the occurrences of forms that constitute s_i, and by $F(o_{i,j})$ the corresponding forms. Finally, we call error the function that associates to each sentence s_i either 1, if s_i's parsing failed, and 0 if it succeeded.

Let $\mathcal{O}_f$ be the set of the occurrences of a form f in the corpus: $\mathcal{O}_f = \{o_{i,j} | F(o_{i,j}) = f\}$. The number of occurrences of f in the corpus is therefore $|\mathcal{O}_f|$.

Let us define at first the *mean global suspicion rate* $\overline{S}$, that is the mean probability that a given occurrence of a form be the cause of a parsing failure. We make the assumption that the failure of the parsing of a sentence has a unique cause (here, a unique form…). This assumption, which is not necessarily exactly verified, simplifies the model and leads to good results. If we call $\mathrm{occ}_{\mathrm{total}}$ the total amount of forms in the corpus, we have then:

$$\overline{S} = \frac{\Sigma_i \mathrm{error}(s_i)}{\mathrm{occ}_{\mathrm{total}}}$$

Let f be a form, that occurs as the j-th form of sentence s_i, which means that $F(o_{i,j}) = f$. Let us assume that s_i's parsing failed: $\mathrm{error}(s_i) = 1$. We call *suspicion rate* of the j-th form $o_{i,j}$ of sentence s_i the probability, denoted by $S_{i,j}$, that the occurrence $o_{i,j}$ of form form f be the cause of the s_i's parsing failure. If, on the contrary, s_i's parsing succeeded, its occurrences have a suspicion rate that is equal to zero.

We then define the *mean suspicion rate* S_f of a form f as the mean of all suspicion rates of its occurrences:

$$S_f = \frac{1}{|\mathcal{O}_f|} \cdot \sum_{o_{i,j} \in \mathcal{O}_f} S_{i,j}$$

To compute these rates, we use a fix-point algorithm by iterating a certain amount of times the following computations. Let us assume that we just completed the n-th iteration: we know, for each sentence s_i, and for each occurrence $o_{i,j}$ of this sentence, the estimation of its suspicion rate $S_{i,j}$ as computed by the n-th iteration, estimation that is denoted by $S_{i,j}^{(n)}$. From this estimation, we compute the $n+1$-th estimation of the mean suspicion rate of each form f, denoted by $S_f^{(n+1)}$:

$$S_f^{(n+1)} = \frac{1}{|\mathcal{O}_f|} \cdot \sum_{o_{i,j} \in \mathcal{O}_f} S_{i,j}^{(n)}$$

This rate[2] allows us to compute a new estimation of the suspicion rate of all occurrences, by giving to each occurrence if a sentence s_i a suspicion rate $S_{i,j}^{(n+1)}$ that is exactly the estimation $S_f^{(n+1)}$ of the mean suspicion rate of S_f of the corresponding form, and then to perform a sentence-level normalization. Thus:

$$S_{i,j}^{(n+1)} = \mathrm{error}(s_i) \cdot \frac{S_{F(o_{i,j})}^{(n+1)}}{\sum_{1 \leq j \leq |s_i|} S_{F(o_{i,j})}^{(n+1)}}$$

At this point, the $n+1$-th iteration is completed, and we can resume again these computations, until convergence on a fix-point. To begin the whole process, we just say, for an occurrence $o_{i,j}$ of sentence s_i, that $S_{i,j}^{(0)} = \mathrm{error}(s_i)/|s_i|$. This means that for a non-parsable sentence, we start from a baseline where all of its occurrences have an equal probability of being the cause of the failure.

After a few dozens of iterations, we get stabilized estimations of the mean suspicion rate each form, which allows:

- to identify the forms that most probably cause errors,
- for each form f, to identify non-parsable sentences s_i where an occurrence $o_{i,j} \in \mathcal{O}_f$ of f is a main suspect and where $o_{i,j}$ has a very

[2]We also performed experiment in which S_f was estimated by an other estimator, namely the *smoothed mean suspicion rate*, denoted by $\tilde{S}_f^{(n)}$, that takes into account the number of occurrences of f. Indeed, the confidence we can have in the estimation $S_f^{(n)}$ is lower if the number of occurrences of f is lower. Hence the idea to smooth $S_f^{(n)}$ by replacing it with a weighted mean $\tilde{S}_f^{(n)}$ between $S_f^{(n)}$ and $\overline{S}$, where the weights λ and $1-\lambda$ depend on $|\mathcal{O}_f|$: if $|\mathcal{O}_f|$ is high, $\tilde{S}_f^{(n)}$ will be close from $S_f^{(n)}$; if it is low, it will be closer from $\overline{S}$:

$$\tilde{S}_f^{(n)} = \lambda(|\mathcal{O}_f|) \cdot S_f^{(n)} + (1 - \lambda(|\mathcal{O}_f|)) \cdot \overline{S}.$$

In these experiments, we used the smoothing function $\lambda(|\mathcal{O}_f|) = 1 - e^{-\beta|\mathcal{O}_f|}$ with $\beta = 0.1$. But this model, used with the ranking according to $M_f = S_f \cdot \ln|\mathcal{O}_f|$ (see below), leads results that are very similar to those obtained without smoothing. Therefore, we describe the smoothing-less model, which has the advantage not to use an empirically chosen smoothing function.

high suspicion rate among all occurrences of form f.

We implemented this algorithm as a *perl* script, with strong optimizations of data structures so as to reduce memory and time usage. In particular, form-level structures are shared between sentences.

2.3 Extensions of the model

This model gives already very good results, as we shall see in section 4. However, it can be extended in different ways, some of which we already implemented.

First of all, it is possible not to stick to forms. Indeed, we do not only work on forms, but on couples made out of a form (a lexical entry) and one or several token(s) that correspond to this form in the raw text (a token is a portion of text delimited by spaces or punctuation tokens).

Moreover, one can look for the cause of the failure of the parsing of a sentence not only in the presence of a form in this sentence, but also in the presence of a bigram[3] of forms. To perform this, one just needs to extend the notions of *form* and *occurrence*, by saying that a (generalized) form is a unigram or a bigram of forms, and that a (generalized) occurrence is an occurrence of a generalized form, i.e., an occurrence of a unigram or a bigram of forms. The results we present in section 4 includes this extension, as well as the previous one.

Another possible generalization would be to take into account facts about the sentence that are not simultaneous (such as form unigrams and form bigrams) but mutually exclusive, and that must therefore be probabilized as well. We have not yet implemented such a mechanism, but it would be very interesting, because it would allow to go beyond forms or n-grams of forms, and to manipulate also lemmas (since a given form has usually several possible lemmas).

3 Experiments

In order to validate our approach, we applied these principles to look for error causes in parsing results given by two deep parsing systems for French, FRMG and SXLFG, on large corpora.

[3]One could generalize this to n-grams, but as n gets higher the number of occurrences of n-grams gets lower, hence leading to non-significant statistics.

3.1 Parsers

Both parsing systems we used are based on deep non-probabilistic parsers. They share:

- the Le*fff* 2 syntactic lexicon for French (Sagot et al., 2005), that contains 500,000 entries (representing 400,000 different forms) ; each lexical entry contains morphological information, sub-categorization frames (when relevant), and complementary syntactic information, in particular for verbal forms (controls, attributives, impersonals,. . .),
- the SXPipe pre-syntactic processing chain (Sagot and Boullier, 2005), that converts a raw text in a sequence of DAGs of forms that are present in the Le*fff* ; SXPipe contains, among other modules, a sentence-level segmenter, a tokenization and spelling-error correction module, named-entities recognizers, and a non-deterministic multi-word identifier.

But FRMG and SXLFG use completely different parsers, that rely on different formalisms, on different grammars and on different parser builder. Therefore, the comparison of error mining results on the output of these two systems makes it possible to distinguish errors coming from the Le*fff* or from SXPipe from those coming to one grammar or the other. Let us describe in more details the characteristics of these two parsers.

The FRMG parser (Thomasset and Villemonte de la Clergerie, 2005) is based on a compact TAG for French that is automatically generated from a meta-grammar. The compilation and execution of the parser is performed in the framework of the DYALOG system (Villemonte de la Clergerie, 2005).

The SXLFG parser (Boullier and Sagot, 2005b; Boullier and Sagot, 2005a) is an efficient and robust LFG parser. Parsing is performed in two steps. First, an Earley-like parser builds a shared forest that represents all constituent structures that satisfy the context-free skeleton of the grammar. Then functional structures are built, in one or more bottom-up passes. Parsing efficiency is achieved thanks to several techniques such as compact data representation, systematic use of structure and computation sharing, lazy evaluation and heuristic and almost non-destructive pruning during parsing.

Both parsers implement also advanced error recovery and tolerance techniques, but they were

corpus	#sentences	#success (%)	#forms	#occ	$\overline{S}$ (%)	Date
MD/FRMG	330,938	136,885 (41.30%)	255,616	10,422,926	1.86%	Jul. 05
MD/SxLFG	567,039	343,988 (60.66%)	327,785	14,482,059	1.54%	Mar. 05
EASy/FRMG	39,872	16,477 (41.32%)	61,135	878,156	2.66%	Dec. 05
EASy/SxLFG	39,872	21,067 (52.84%)	61,135	878,156	2.15%	Dec. 05

Table 1: General information on corpora and parsing results

useless for the experiments described here, since we want only to distinguish sentences that receive a full parse (without any recovery technique) from those that do not.

3.2 Corpora

We parsed with these two systems the following corpora:

MD corpus : This corpus is made out of 14.5 million words (570,000 sentences) of general journalistic corpus that are articles from the *Monde diplomatique*.

EASy corpus : This is the 40,000-sentence corpus that has been built for the EASy parsing evaluation campaign for French (Paroubek et al., 2005). We only used the raw corpus (without taking into account the fact that a manual parse is available for 10% of all sentences). The EASy corpus contains several sub-corpora of varied style: journalistic, literacy, legal, medical, transcription of oral, e-mail, questions, etc.

Both corpora are raw in the sense that no cleaning whatsoever has been performed so as to eliminate some sequences of characters that can not really be considered as sentences.

Table 1 gives some general information on these corpora as well as the results we got with both parsing systems. It shall be noticed that both parsers did not parse exactly the same set and the same number of sentences for the MD corpus, and that they do not define in the exactly same way the notion of sentence.

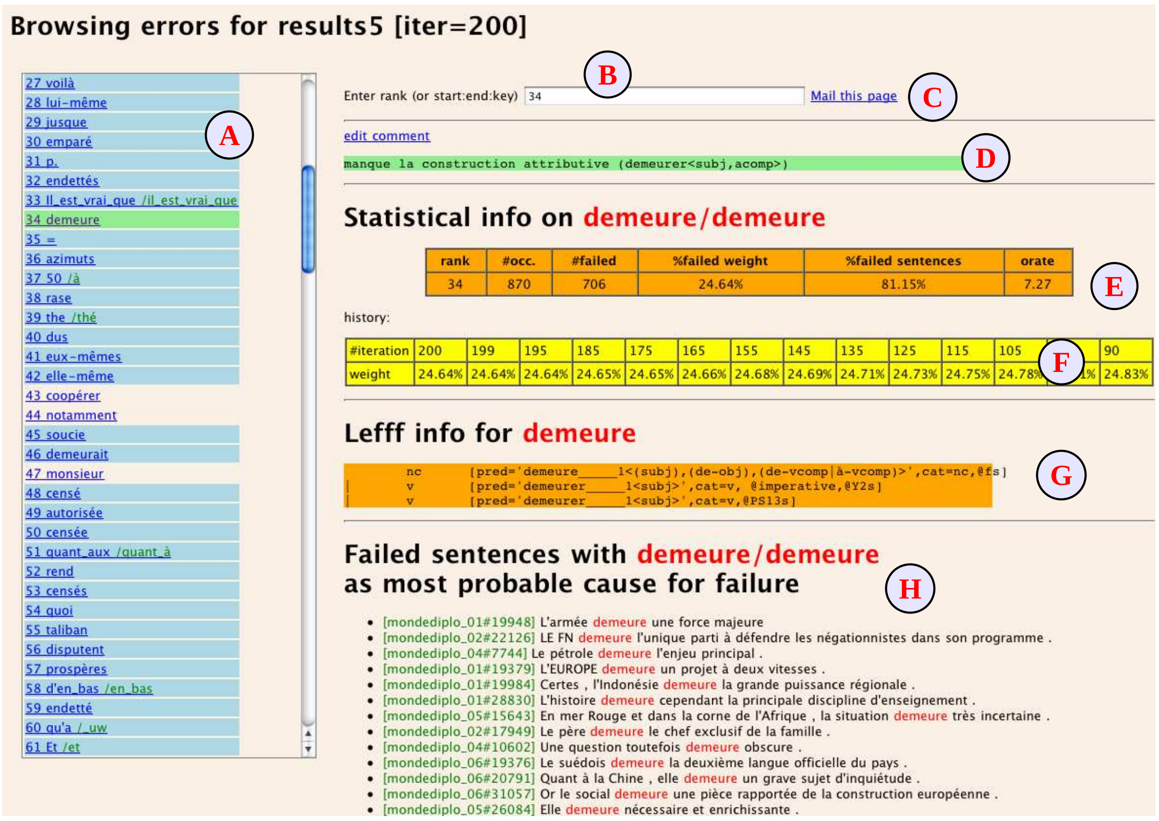
3.3 Results visualization environment

We developed a visualization tool for the results of the error mining, that allows to examine and annotate them. It has the form of an HTML page that uses dynamic generation methods, in particular *javascript*. An example is shown on Figure 1.

To achieve this, suspicious forms are ranked according to a measure M_f that models, for a given form f, the benefit there is to try and correct the (potential) corresponding error in the resources. A user who wants to concentrate on almost certain errors rather than on most frequent ones can visualize suspicious forms ranked according to $M_f = S_f$. On the contrary, a user who wants to concentrate on most frequent potential errors, rather than on the confidence that the algorithm has given to errors, can visualize suspicious forms ranked according to[4] $M_f = S_f|\mathcal{O}_f|$. The default choice, which is adopted to produce all tables shown in this paper, is a balance between these two possibilities, and ranks suspicious forms according to $M_f = S_f \cdot \ln |\mathcal{O}_f|$.

The visualization environment allows to browse through (ranked) suspicious forms in a scrolling list on the left part of the page (**A**). When the suspicious form is associated to a token that is the same as the form, only the form is shown. Otherwise, the token is separated from the form by the symbol "/". The right part of the page shows various pieces of information about the currently selected form. After having given its rank according to the ranking measure M_f that has been chosen (**B**), a field is available to add or edit an annotation associated with the suspicious form (**D**). These annotations, aimed to ease the analysis of the error mining results by linguists and by the developers of parsers and resources (lexica, grammars), are saved in a database (SQLITE). Statistical information is also given about f (**E**), including its number of occurrences occ_f, the number of occurrences of f in non-parsable sentences, the final estimation of its mean suspicion rate S_f and the rate $\mathrm{err}(f)$ of non-parsable sentences among those where f appears. This indications are complemented by a brief summary of the iterative process that shows the convergence of the successive estimations of S_f. The lower part of the page gives a mean to identify the cause of f-related errors by showing

[4]Let f be a form. The suspicion rate S_f can be considered as the probability for a particular occurrence of f to cause a parsing error. Therefore, $S_f|\mathcal{O}_f|$ models the number of occurrences of f that do cause a parsing error.



Browsing errors for results5 [iter=200]

27 voilà
28 lui-même
29 jusque
30 emparé
31 p.
32 endettés
33 Il_est_vrai_que /il_est_vrai_que
34 demeure
35 =
36 azimuts
37 50 /à
38 rase
39 the /thé
40 dus
41 eux-mêmes
42 elle-même
43 coopérer
44 notamment
45 soucie
46 demeurait
47 monsieur
48 censé
49 autorisée
50 censée
51 quant_aux /quant_à
52 rend
53 censés
54 quoi
55 taliban
56 disputent
57 prospères
58 d'en_bas /en_bas
59 endetté
60 qu'a /_uw
61 Et /et

Enter rank (or start:end:key) 34 Mail this page

edit comment

```
manque la construction attributive (demeurer<subj,acomp>)
```

Statistical info on demeure/demeure

rank	#occ.	#failed	%failed weight	%failed sentences	orate
34	870	706	24.64%	81.15%	7.27

history:

#iteration	200	199	195	185	175	165	155	145	135	125	115	105	[illegible]	90
weight	24.64%	24.64%	24.64%	24.65%	24.65%	24.66%	24.68%	24.69%	24.71%	24.73%	24.75%	24.78%	[illegible]	24.83%

Lefff info for demeure

```
        nc      [pred='demeure_____1<(subj),(de-obj),(de-vcomp|à-vcomp)>',cat=nc,@fs]
|       v       [pred='demeurer_____1<subj>',cat=v, @imperative,@Y2s]
|       v       [pred='demeurer_____1<subj>',cat=v,@PS13s]
```

Failed sentences with demeure/demeure as most probable cause for failure

- [mondediplo_01#19948] L'armée demeure une force majeure
- [mondediplo_02#22126] LE FN demeure l'unique parti à défendre les négationnistes dans son programme .
- [mondediplo_04#7744] Le pétrole demeure l'enjeu principal .
- [mondediplo_01#19379] L'EUROPE demeure un projet à deux vitesses .
- [mondediplo_01#19984] Certes , l'Indonésie demeure la grande puissance régionale .
- [mondediplo_01#28830] L'histoire demeure cependant la principale discipline d'enseignement .
- [mondediplo_05#15643] En mer Rouge et dans la corne de l'Afrique , la situation demeure très incertaine .
- [mondediplo_02#17949] Le père demeure le chef exclusif de la famille .
- [mondediplo_04#10602] Une question toutefois demeure obscure .
- [mondediplo_06#19376] Le suédois demeure la deuxième langue officielle du pays .
- [mondediplo_06#20791] Quant à la Chine , elle demeure un grave sujet d'inquiétude .
- [mondediplo_06#31057] Or le social demeure une pièce rapportée de la construction européenne .
- [mondediplo_05#26084] Elle demeure nécessaire et enrichissante .

Figure 1: Error mining results visualization environment (results are shown for MD/FRMG).

f's entries in the Le*fff* lexicon (**G**) as well as non-parsable sentences where f is the main suspect and where one of its occurrences has a particularly high suspicion rate[5] (**H**).

The whole page (with annotations) can be sent by e-mail, for example to the developer of the lexicon or to the developer of one parser or the other (**C**).

4 Results

In this section, we mostly focus on the results of our error mining algorithm on the parsing results provided by SxLfg on the MD corpus. We first present results when only forms are taken into account, and then give an insight on results when both forms and form bigrams are considered.

4.1 Finding suspicious forms

The execution of our error mining script on MD/SxLfg, with $i_{max} = 50$ iterations and when only (isolated) forms are taken into account, takes less than one hour on a 3.2 GHz PC running Linux with a 1.5 Go RAM. It outputs 18,334 *relevant* suspicious forms (out of the 327,785 possible ones), where a relevant suspicious form is defined as a form f that satisfies the following arbitrary constraints:[6] $S_f^{(i_{max})} > 1,5 \cdot \overline{S}$ and $|\mathcal{O}_f| > 5$.

We still can not prove theoretically the convergence of the algorithm.[7] But among the 1000 best-ranked forms, the last iteration induces a mean variation of the suspicion rate that is less than 0.01%.

On a smaller corpus like the EASy corpus, 200 iterations take 260s. The algorithm outputs less than 3,000 relevant suspicious forms (out of the 61,125 possible ones). Convergence information

[5] Such an information, which is extremely valuable for the developers of the resources, can not be obtained by global (form-level and not occurrence-level) approaches such as the $\mathrm{err}(f)$-based approach of (van Noord, 2004). Indeed, enumerating all sentences which include a given form f, and which did not receive a full parse, is not precise enough: it would show at the same time sentences wich fail because of f (e.g., because its lexical entry lacks a given sub-categorization frame) and sentences which fail for an other independent reason.

[6] These constraints filter results, but all forms are taken into account during all iterations of the algorithm.

[7] However, the algorithms shares many common points with iterative algorithm that are known to converge and that have been proposed to find maximum entropy probability distributions under a set of constraints (Berger et al., 1996). Such an algorithm is compared to ours later on in this paper.

is the same as what has been said above for the MD corpus.

Table 2 gives an idea of the repartition of suspicious forms w.r.t. their frequency (for FRMG on MD), showing that rare forms have a greater probability to be suspicious. The most frequent suspicious form is the double-quote, with (only) $S_f = 9\%$, partly because of segmentation problems.

4.2 Analyzing results

Table 3 gives an insight on the output of our algorithm on parsing results obtained by SXLFG on the MD corpus. For each form f (in fact, for each couple of the form *(token,form)*), this table displays its suspicion rate and its number of occurrences, as well as the rate $\mathrm{err}(f)$ of non-parsable sentences among those where f appears and a short manual analysis of the underlying error.

In fact, a more in-depth manual analysis of the results shows that they are very good: errors are correctly identified, that can be associated with four error sources: (1) the Le*fff* lexicon, (2) the SXPipe pre-syntactic processing chain, (3) imperfections of the grammar, but also (4) problems related to the corpus itself (and to the fact that it is a raw corpus, with meta-data and typographic noise).

On the EASy corpus, results are also relevant, but sometimes more difficult to interpret, because of the relative small size of the corpus and because of its heterogeneity. In particular, it contains e-mail and oral transcriptions sub-corpora that introduce a lot of noise. Segmentation problems (caused both by SXPipe and by the corpus itself, which is already segmented) play an especially important role.

4.3 Comparing results with results of other algorithms

In order to validate our approach, we compared our results with results given by two other relevant algorithms:

- van Noord's (van Noord, 2004) (form-level and non-iterative) evaluation of $\mathrm{err}(f)$ (the rate of non-parsable sentences among sentences containing the form f),
- a standard (occurrence-level and iterative) maximum entropy evaluation of each form's contribution to the success or the failure of a sentence (we used the MEGAM package (Daumé III, 2004)).

As done for our algorithm, we do not rank forms directly according to the suspicion rate S_f computed by these algorithms. Instead, we use the M_f measure presented above ($M_f = S_f \cdot \ln |\mathcal{O}_f|$). Using directly van Noord's measure selects as most suspicious words very rare words, which shows the importance of a good balance between suspicion rate and frequency (as noted by (van Noord, 2004) in the discussion of his results). This remark applies to the maximum entropy measure as well.

Table 4 shows for all algorithms the 10 best-ranked suspicious forms, complemented by a manual evaluation of their relevance. One clearly sees that our approach leads to the best results. Van Noord's technique has been initially designed to find errors in resources that already ensured a very high coverage. On our systems, whose development is less advanced, this technique ranks as most suspicious forms those which are simply the most frequent ones. It seems to be the case for the standard maximum entropy algorithm, thus showing the importance to take into account the fact that there is at least one cause of error in any sentence whose parsing failed, not only to identify a main suspicious form in each sentence, but also to get relevant global results.

4.4 Comparing results for both parsers

We complemented the separated study of error mining results on the output of both parsers by an analysis of merged results. We computed for each form the harmonic mean of both measures $M_f = S_f \cdot \ln |\mathcal{O}_f|$ obtained for each parsing system. Results (not shown here) are very interesting, because they identify errors that come mostly from resources that are shared by both systems (the Le*fff* lexicon and the pre-syntactic processing chain SXPipe). Although some errors come from common lacks of coverage in both grammars, it is nevertheless a very efficient mean to get a first repartition between error sources.

4.5 Introducing form bigrams

As said before, we also performed experiments where not only forms but also form bigrams are treated as potential causes of errors. This approach allows to identify situations where a form is not in itself a relevant cause of error, but leads often to a parse failure when immediately followed or preceded by an other form.

Table 5 shows best-ranked form bigrams (forms that are ranked in-between are not shown, to em-

#occ	$> 100\ 000$	$> 10\ 000$	> 1000	> 100	> 10
#forms	13	84	947	8345	40 393
#suspicious forms (%)	1 (7.6%)	13 (15.5%)	177 (18.7%)	1919 (23%)	12 022 (29.8%)

Table 2: Suspicious forms repartition for MD/FRMG

Rank	Token(s)/form	$S_f^{(50)}$	$\|\mathcal{O}_f\|$	$\mathrm{err}(f)$	M_f	Error cause
1	_____/_UNDERSCORE	100%	6399	100%	8.76	corpus: typographic noise
2	(...)	46%	2168	67%	2.82	SxPipe: should be treated as skippable words
3	2_]/_NUMBER	76%	30	93%	2.58	SxPipe: bad treatment of list constructs
4	privées	39%	589	87%	2.53	Le*fff*: misses as an adjective
5	Haaretz/_Uw	51%	149	70%	2.53	SxPipe: needs local grammars for references
6	contesté	52%	122	90%	2.52	Le*fff*: misses as an adjective
7	occupés	38%	601	86%	2.42	Le*fff*: misses as an adjective
8	privée	35%	834	82%	2.38	Le*fff*: misses as an adjective
9	[...]	44%	193	71%	2.33	SxPipe: should be treated as skippable words
10	faudrait	36%	603	85%	2.32	Le*fff*: can have a nominal object

Table 3: Analysis of the 10 best-ranked forms (ranked according to $M_f = S_f \cdot \ln |\mathcal{O}_f|$)

	this paper		global		maxent	
Rank	Token(s)/form	Eval	Token(s)/form	Eval	Token(s)/form	Eval
1	_____/_UNDERSCORE	++	*	+	pour	-
2	(...)	++	,	-	)	-
3	2_]/_NUMBER	++	livre	-	à	-
4	privées	++	.	-	qu'il/qu'	-
5	Haaretz/_Uw	++	de	-	sont	-
6	contesté	++	;	-	le	-
7	occupés	++	:	-	qu'un/qu'	+
8	privée	++	la	-	qu'un/un	+
9	[...]	++	étrangères	-	que	-
10	faudrait	++	lecteurs	-	pourrait	-

Table 4: The 10 best-ranked suspicious forms, according the the M_f measure, as computed by different algorithms: ours (*this paper*), a standard maximum entropy algorithm (*maxent*) and van Noord's rate $\mathrm{err}(f)$ (*global*).

Rank	Tokens and forms	M_f	Error cause
4	Toutes/toutes les	2.73	grammar: badly treated pre-determiner adjective
6	y en	2,34	grammar: problem with the construction *il y en a...*
7	in "	1.81	Le*fff*: *in* misses as a preposition, which happends before book titles (hence the ")
10	donne à	1.44	Le*fff*: *donner* should sub-categorize à-vcomps (*donner à voir...*)
11	de demain	1.19	Le*fff*: *demain* misses as common noun (standard adv are not preceded by prep)
16	(22/_NUMBER	0.86	grammar: footnote references not treated
16	22/_NUMBER)	0.86	as above

Table 5: Best ranked form bigrams (forms ranked inbetween are not shown; ranked according to $M_f = S_f \cdot \ln |\mathcal{O}_f|$). *These results have been computed on a subset of the MD corpus (60,000 sentences).*

phasize bigram results), with the same data as in table 3.

5 Conclusions and perspectives

As we have shown, parsing large corpora allows to set up error mining techniques, so as to identify missing and erroneous information in the different resources that are used by full-featured parsing systems. The technique described in this paper and its implementation on forms and form bigrams has already allowed us to detect many errors and omissions in the Le*fff* lexicon, to point out inappropriate behaviors of the SxPipe pre-syntactic processing chain, and to reveal the lack of coverage of the grammars for certain phenomena.

We intend to carry on and extend this work. First of all, the visualization environment can be enhanced, as is the case for the implementation of the algorithm itself.

We would also like to integrate to the model the possibility that facts taken into account (today, forms and form bigrams) are not necessarily certain, because some of them could be the consequence of an ambiguity. For example, for a given form, several lemmas are often possible. The probabilization of these lemmas would thus allow to look for most suspicious lemmas.

We are already working on a module that will allow not only to detect errors, for example in the lexicon, but also to propose a correction. To achieve this, we want to parse anew all non-parsable sentences, after having replaced their main suspects by a special form that receives under-specified lexical information. These information can be either very general, or can be computed by appropriate generalization patterns applied on the information associated by the lexicon with the original form. A statistical study of the new parsing results will make it possible to propose corrections concerning the involved forms.

References

A. Berger, S. Della Pietra, and V. Della Pietra. 1996. A maximun entropy approach to natural language processing. *Computational Linguistics*, 22(1):pp. 39–71.

Pierre Boullier and Benoît Sagot. 2005a. Analyse syntaxique profonde à grande échelle: SxLfg. *Traitement Automatique des Langues (T.A.L.)*, 46(2).

Pierre Boullier and Benoît Sagot. 2005b. Efficient and robust LFG parsing: SxLfg. In *Proceedings of IWPT'05*, Vancouver, Canada, October.

Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at `http://www.isi.edu/~hdaume/docs/daume04cg-bfgs.ps`, implementation available at `http://www.isi.edu/~hdaume/megam/`.

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

Benoît Sagot and Pierre Boullier. 2005. From raw corpus to word lattices: robust pre-parsing processing. In *Proceedings of L&TC 2005*, Poznań, Pologne.

Benoît Sagot, Lionel Clément, Éric Villemonte de la Clergerie, and Pierre Boullier. 2005. Vers un méta-lexique pour le français : architecture, acquisition, utilisation. Journée d'étude de l'ATALA sur l'interface lexique-grammaire et les lexiques syntaxiques et sémantiques, March.

François Thomasset and Éric Villemonte de la Clergerie. 2005. Comment obtenir plus des méta-grammaires. In *Proceedings of TALN'05*, Dourdan, France, June. ATALA.

Gertjan van Noord. 2004. Error mining for wide-coverage grammar engineering. In *Proc. of ACL 2004*, Barcelona, Spain.

Éric Villemonte de la Clergerie. 2005. DyALog: a tabular logic programming based environment for NLP. In *Proceedings of 2nd International Workshop on Constraint Solving and Language Processing (CSLP'05)*, Barcelona, Spain, October.