Automatically enriching spoken corpora with syntactic information for linguistic studies

Alexis Nasr, Frederic Bechet, Benoit Favre, Thierry Bazillon, Jose Deulofeu, Andre Valli

Aix Marseille Université, CNRS, LIF UMR 7279, Marseille, France
alexis.nasr@lif.univ-mrs.fr

Abstract

Syntactic parsing of speech transcriptions faces the problem of the presence of disfluencies that break the syntactic structure of the utterances. We propose in this paper two solutions to this problem. The first one relies on a disfluencies predictor that detects disfluencies and removes them prior to parsing. The second one integrates the disfluencies in the syntactic structure of the utterances and train a disfluencies aware parser.

Keywords: speech processing; parsing; syntactic annotation

1. Introduction

Corpus-based linguistic studies require large amounts and a wide variety of syntactically annotated corpora. In this paper, we present some results on the automatic syntactic annotation of spoken corpora containing spontaneous speech. This study takes place in the framework of the ORFEO project. The goal of this project is to provide to the scientific community access to a large number of spoken and written corpora annotated with a common scheme to facilitate linguistic studies across genre and modality (written and oral). To this purpose, in a first phase of the project, it was decided to annotate automatically all corpora with a common part of speech (POS) and syntactic dependency labels tagset. If this process is rather straightforward for written corpora, it raises two major issues when dealing with spoken corpora containing spontaneous speech. The first one concerns the notion of sentence , which is important for syntactic parsing but does not have a clear definition when dealing with speech data. The second one concerns the presence, in spoken transcriptions, of specific phenomena, often referred as disfluencies, such as filled pauses, discourse markers, repetitions, false starts, which are characteristics of spontaneous speech.

We focus is this paper on the second issue: dealing with disfluencies that break the syntactic structure of utterances, and propose two ways to deal with them: either integrate them in the syntax and learn parsers that are able to deal with them or devise a specific module that detects and removes them prior to parsing. In order to compare these two approaches we performed our experiments on the RATP-DECODA corpus. This corpus is particularly well suited to this study as it contains both very spontaneous speech and linguistic annotations (disfluencies, POS and syntactic dependencies). The models and tagging/parsing strategies trained on this corpus will be applied in a second phase to all the other ORFEO corpora.

The structure of the paper is the following: Section 2. describes the DECODA corpus and the annotation schema, section 3. describes the POS tagger used. Although tagging is not the main topic of this paper, it plays an important role both for disfluency detection and parsing. Sections 4. and 5. respectively describe the disfluency prediction module and the parser. Section 6. concludes the paper.

2. The RATP-DECODA Corpus

The RATP-DECODA corpus (Bechet et al., 2012) collected within the DECODA project is made of 2100 dialogs, corresponding to about 74 hours of sound recordings. It contains dialogs recorded at the Paris Transport Authority (RATP) call-center between operators and users of Paris public transports. The average duration of a dialog is about 3 minutes. This corpus is fully anonymized, manually segmented, transcribed and annotated.

Five levels of linguistic annotations have been performed on the transcriptions of the DECODA corpus: Disfluencies, POS tags, Syntactic dependencies, Chunks and Named Entities. Chunks and Named Entities have are not covered in this study. More details on the annotation process can be found in (Bazillon et al., 2012).

The disfluencies considered are repetitions, such as "*le le*" (*the the*), false starts, such as "*bonj-*" (the beginning of the word "*bonjour*" (*good morning*)) and discourse markers, such as "*euh*" or "*bien*" (*well*). The category discourse marker also includes the filled pauses. The repetitions considered are: single words "*le le*" (*the the*); wor sequences "*on est on est*" (*we are we are*); or approximate repetitions with synonyms "*je voudrais je veux*" (*I would like I want*). False starts have been marked following the ESTER (Galliano et al., 2009) conventions for speech transcription: each false start is marked with the symbol *()* added to the part of work pronounced. For example: "*met()*" for "*metro*".

Each word of the corpus is labelled with a disfluency label chosen among the set (REP, DM, FALSTART, NULL).

28% of the speech segments contain at least one discourse marker (or a filled pause), 8% contain a repetition and 1% contain a false start.

The ORFEO POS tagset is made of 17 tags, described in table 1. Words that are part of a disfluent expression have been assigned a POS. For example, a repetition such as: "*je je je veux*" (*I I I want*) is tagged: "CLI CLI CLI VRB".

The OREEO syntactic dependency labels tagset is restricted to 12 labels, described in table 2. The DISFLINK dependency is introduced in order to link disfluent words to the syntactic structure of the utterance. Disfluent words are systematically linked to the preceding word in the utterance. There is no deep

ADJ	Adjective
ADN	Negative Adverb
ADV	Adverb
CLI	Clitic
CLN	Negative Clitic
COO	Coordinating Conjunction
CSU	Subordinating Conjunction
DET	Determiner
INT	Interjection
NOM	Name
PRE	Preposition
PRO	Pronoun
PRQ	Relative and Interrogative Pronoun
VNF	Verbe Infinitive Form
VPP	Past Participle
VPR	Present Participle
VRB	Finite Tense Verb

Table 1: The ORFEO part of speech tagset

SUJ	Subject
OBJ	Direct Object
OBL	Indirect Object
AUX	Auxiliary
AFF	Affix
DET	Determiner
MOD_REL	Relative Clause
MOD	Modifier
COORD	Coordination
DEP_COORD	Coordinated Element
ROOT	Utterance Root
DISFLINK	Disfluency

Table 2: The ORFEO dependencies label tagset

linguistic reason for this, the only aim is to keep the tree structure of the syntactic representation. When a disfluent word starts an utterance, it is linked to an phony empty word that starts all sentences.

An utterance of the DECODA corpus is shown in table 3 (*yes so there are there are three uh three trains every four*). The first column corresponds to the position of the token in the utterance, the second to the token itself, the third to its POS, the fourth to its disfluency status (repetition, discourse marker or regular word) the fifth indicates the position of the syntactic governor and the last one is the dependency label. This utterance contains two discourse markers, the leading *oui* and the hesitation *euh*. It also contains two repetitions, a complex one *il y a* and a simple one *trois* that is separated from its copy by the discourse marker *euh*.

In order to train and evaluate the NLP software used in this study, the RATP-DECODA corpus has been divided into a training and a test set. This partition is presented in Table 4. We have discarded the false starts disfluencies from the corpus since they are non-ambiguous and can be removed directly.

It is important to note that the test set of the DECODA corpus has been manually validated while the train set has been semi automatically corrected, through an iterative process that is

1	oui	ADV	DM	0	DISFLINK
2	donc	COO	NULL	0	ROOT
3	il	CLI	REP	2	DISFLINK
4	y	CLI	REP	3	DISFLINK
5	a	VRB	REP	4	DISFLINK
6	il	CLI	NULL	8	SUJ
7	y	CLI	NULL	8	AFF
8	a	VRB	NULL	2	DEP_COORD
9	trois	ADJ	REP	8	DISFLINK
10	euh	INT	DM	9	DISFLINK
11	trois	DET	NULL	12	DET
12	RER	NOM	NULL	8	OBJ
13	sur	PRE	NULL	12	MOD
14	quatre	ADJ	NULL	13	OBJ

Table 3: An annotated utterance from the DECODA corpus

described in (Bazillon et al., 2012)

	turn nb.	token nb		
		TOTAL	REP	DM
TRAIN	93,561	521,377	15,484	35,183
TEST	3,639	25,231	882	1692

Table 4: Disfluency statistics on the DECODA corpus

3. Tagging

The tagger used in this study is based on a sequential Conditional Random Field predicting the sequence of tags given features anchored at each word: word n-grams up to 3, morphological traits (prefixes and suffixes of length up to 4, capitalization, character classes) and POS tag bigrams. The model is trained with LBFGS using crfsuite (Okazaki, 2007). At decoding time, we use a lexicon of possible tags for known forms and allow the prediction of any tag for unknown forms.

Table 5 reports the result of the tagging experiment. The first column indicates whether the tagger was trained on a corpus where the disfluencies have been kept (DISF) or removed (NODISF).

The second column specifies whether the test corpus contains disfluencies or not. The third, fourth, fifth and sixth columns indicate respectively parsing accuracy on the whole test corpus (TOTAL), on the non disfulent tokens of the corpus (NULL), on the repetition disfluencies (REP) and on discourse marker disfluencies (DM).

The results show that the disfluencies aware tagger gives better results on the test corpus with disfluences, which does not come as a surprise. They also show that the disfluencies aware tagger behaves well on non disfluent parts of the corpus.

This experiment clearly shows that disfulent input can be tagged with the same accuracy as non disfluent input when the tagger is trained on disfluent data. There is therefore no need to try to remove the disfluencies prior to tagging a corpus of spoken transcriptions.

It is therefore the disfluencies aware tagger that is used for the experiments in the next sections.

TRAIN	TEST	TOTAL	NULL	REP	DM
NODISF	NODISF	97.27	–	–	–
	DISF	96.38	97.18	95.69	85.81
DISF	NODISF	97.21	–	–	–
	DISF	97.30	97.18	95.91	99.70

Table 5: Tagging accuracy according to whether the model is trained on disfluent speech or not, and broken down by disfluency category.

4. Disfluencies Prediction

We have seen in the previous section that disfluencies do not affect tagging performance. However it is important to be able to identify them in order to remove them for other processes such as dependency parsing as we will see in the next section. We investigate in this section the performance of a disfluency tagger trained with different set of features.

We will focus on the tagging of repetitions (REP) and discourse markers (DM), as they represent ambiguous disfluency categories: discourse markers such as "*oui*" (*yes*) or "*bien*" (*good*) can also be non-disfluencies; repetitions like "*nous nous*" (*we we*) in French can also be non-disfluencies in sentences such as: "*nous nous en allons*" (*we are leaving*). And of course, approximate repetitions are another source of ambiguities.

The tagging approach chosen is the state-of-the-art Conditional Random Field (CRF) approach with the tool *CRF++*[1]. Three sets of features are compared:

1. **word n-gram**: word features with 1,2,3-grams
2. **repetition features**: each word is associated with its number of occurrence in a fixed window (5 words) in front of the current word position
3. **POS n-gram**: POS features with 1,2,3-grams

The results are given in Table 6 for the test corpus which contains 71K tokens with 6.8K disfluencies (discourse markers and repetitions). The POS labels are obtained automatically with the tagger presented in the previous section.

As we can see that adding POS features brings a substantial improvement compared to lexical features alone. This confirms the fact that disfluencies follow some kind of syntactic pattern, at least locally at the POS level. We will investigate the same hypothesis at the sentence level with dependency parsing in the next section.

Features	PREC	REC	F-MEAS.
word n-gram	98.1	76.2	85.8
word + rep. feat	96.7	81.1	88.2
word + POS n-gram	97.5	83.5	89.9
word + POS + rep. feat.	96.0	85.1	90.2

Table 6: Disfluency prediction performances for different feature sets

[1]crfpp.googlecode.com

5. Parsing

Three series of experiments have been conducted with respect to syntactic parsing. The first one corresponds to the situation where we do not have access to a syntactically annotated dialog corpus. In the second situation we have access to a dialog corpus that has been annotated with POS and disfluency tags and in the third situation we have at our disposal a syntactically annotated dialog corpora on which a parser can been trained.

The parser used for these experiments is a graph based parser (McDonald et al., 2005) implemented in the MACAON tool suite (Nasr et al., 2011). It is a parser that produces dependency trees. For a given sentence, the parser considers all its possible projective dependency trees and selects the tree that maximizes a score. In its simplest form (called first order), the score of a tree is the sum of the scores of every dependency that make up the tree. The score of a dependency is itself the sum of the weights of the features that correspond to this dependency. Feature weights are learned from training data using the perceptron algorithm. A more sophisticated scoring function, known as second order, computes the score of a tree as the sum of scores of subtrees made of one or two dependencies. Second order models usually yield better results than first order ones at the cost of higher computational complexity.

Two standard metrics are used to measure the quality of the syntactic trees produced by the parser. The Unlabeled Attachment Score (UAS) which is the proportion of words in a sentence for which the right governor has been assigned by the parser and the Labeled Attachment Score (LAS) which also takes into account the label of the dependency that links a word to its governor.

In the three subsection below we describe experiments corresponding to the three scenarios described above.

5.1. Training on written material

In this experiment, the parser has been trained on the training set of the French Treebank (FTB) (Abeillé et al., 2003). The FTB corpus is a collection of newspaper articles from the French journal *Le Monde*. It is therefore quite different from the genre of data that we want to parse. The parser has been evaluated on three corpora: the test set of the FTB (for sanity check), the test set of the DECODA corpus without disfluencies and the test set of the DECODA corpus with disfluencies. It must be noted that the second experiment (predicting parse trees on the test set of the DECODA corpus without disfluencies) is artificial since the disfluencies have been manually removed from the corpus. It represents an upper bound of the parsing accuracy if we had at our disposal a perfect disfluency predictor.

The results are reported in Table 7.

		FTB	DECODA NODISF	DECODA DISF
1^{st} order	UAS	87.92	71.01	65.78
	LAS	85.54	64.28	58.28
2^{nd} order	UAS	89.71	71.87	66.09
	LAS	87.32	65.30	58.70

Table 7: Parsing accuracy of a parser trained on FTB

Table 7 shows that while the performances of the parser are state of the art on the FTB test set, they drop sharply for the DECODA corpus without disfluencies and continue to drop for the DECODA corpus with disfluencies. This does not come as a surprise since the parser is evaluated on corpora that are further and further away from the FTB training corpus.
The table also shows that second order models behave better on all corpora, although the difference is more important on written corpora.

5.2. Automatically removing disfluencies

In this experiment, the parser has been trained, as before, on the FTB corpus and is evaluated on the DECODA corpus from which disfluencies have been automatically removed using the disfluency predictor described in Section 4.. Results are reported in Table 8 (columns DECODA NODISF and DECODA DISF have been reproduced from Table 7 for readability purpose).

		DECODA AUTO	DECODA NODISF	DECODA DISF
1^{st} order	UAS	71.66	*71.01*	*65.78*
	LAS	65.19	*64.28*	*58.28*
2^{nd} order	UAS	72.29	*71.87*	*66.09*
	LAS	65.88	*65.30*	*58.70*

Table 8: Parsing accuracy computed on the DECODA corpus from which disfluencies have been automatically removed

Table 8 shows that the strategy of automatically removing the disfluencies prior to parsing is an interesting one. The LAS jumps from 58.28 to 65.19 for second order models. It must be noted that the results on DECODA DISF and DECODA AUTO are directly comparable since after parsing, the disfluencies that have been automatically detected and removed are reintroduced in the output of the parser and linked to the preceding word with a DISF dependency, as described in Section 2..

5.3. Training on the DECODA corpus

In this last series of experiments, the parser is trained on the DECODA corpus and evaluated on our four test corpora. The evaluation on the FTB corpus is not directly relevant to our purpose, it has been added for completeness. The results are reported in Table 9.

		DECODA			FTB
		AUTO	NODISF	DISF	
1^{st} order	UAS	85.90	86.47	85.83	77.93
	LAS	83.86	84.60	83.62	73.80
2^{nd} order	UAS	85.63	86.07	85.62	77.25
	LAS	83.61	84.19	83.56	73.20

Table 9: Parsing accuracy of a parser trained on DECODA

This table shows several interesting features. The first one is that the parser trained on the DECODA corpus yields much better results than a parser trained on the FTB corpus even when disfluencies are removed. The LAS jumps from 65.19 to 83.86.

Two reasons can explain this result. The first one is that the DECODA corpus has a quite restricted and specific vocabulary and the parser used is quite good at learning lexical affinities.
The second one is that the DECODA corpus has a rather simple syntax with utterances generally restricted to simple clauses and less common ambiguities, such as prepositional attachment and coordination, than written texts.
The rather simple syntax of the DECODA corpus also explains that first order models behave as well as second order ones, as shown in Table 9.
In this experiment, besides parsing, the parser acts as a disfluency predictor since every dependent of a dependency labeled DISFLINK is considered as a disfluency by the parser. The prediction of the parser on this task has a recall of 71.49, a precision of 88.00 and an F-measure of 78.89. It is interesting to compare these scores with the scores of the disfluencies predictor of Section 4. which had a recall of 94.1, a precision of 96.9 and an F-measure of 95.5. The predictions of the parser are therefore significantly lower than the predictions of the disfluencies predictor. The poor results obtained by the parser as a disfluencies predictor might be due to the fact that the parser is not good at detecting repetitions, which account for 35% of the disfluencie, whereas the disfluencies predictor has specific features that model repetitions. The low accuracy of disfluencies detection explains that the accuracy of the disfluencies aware parser on the DECODA test set from which disflencies have been automatically removed is higher than the accuracy of the same parser on the DECODA test set with disluencies.
In the last experiments, a parser is trained on the non disfluent part of the DECODA corpus. The results are reported in Table 10. As expected, the parser behaves poorly on the DECODA test set but its performances are better on the AUTO corpus (LAS = 83.45 UAS = 85.52). These figures can be directly compared with the performances of the disfluencies aware parser evaluated on the DECODA test set (LAS = 83.86 UAS = 85.90). As one can see, the performances are very close.

		DECODA			FTB
		AUTO	NODISF	DISF	
1^{st} order	UAS	85.52	86.50	80.08	77.71
	LAS	83.45	84.70	77.87	73.67
2^{nd} order	UAS	85.03	86.06	79.61	76.77
	LAS	82.96	84.26	77.40	72.76

Table 10: Parsing accuracy of a parser trained on DECODA without disfluencies

6. Conclusions

We have shown in this paper that high accuracy parsing can be performed on spontaneous speech despite disfluencies. Two methods have been proposed and evaluated. In the first one a disfluency predictor is run on a test corpus. The detected disfluencies are then removed and the corpus is sent to the parser. In the second one, a parser is trained on a syntactically annotated corpus containing disfluencies. The test corpus with disfluencies is then parsed with the disfluencies aware parser. In

this setting, parsing and disfluencies detection are performed jointly by the parser. The performances of the two methods are very close.
This process has been tuned on the RATP-DECODA, it is now applied to all the other spoken corpora of the ORFEO project.

7. Acknowledgments

This work has been funded by the French *Agence Nationale pour la Recherche* (ANR), through the project ORFEO (ANR-12-CORP-0005).

8. References

Abeillé, A., Clément, L., and Toussenel, F. (2003). Building a treebank for french. In Abeillé, Anne, editor, *Treebanks*. Kluwer, Dordrecht.

Bazillon, Thierry, Deplano, Melanie, Bechet, Frederic, Nasr, Alexis, and Favre, Benoit. (2012). Syntactic annotation of spontaneous speech: application to call-center conversation data. In *Proceedings of LREC*, Istambul.

Bechet, Frederic, Maza, Benjamin, Bigouroux, Nicolas, Bazillon, Thierry, El-Bèze, Marc, Mori, Renato De, and Arbillot, E. (2012). Decoda: a call-center human-human spoken conversation corpus. In *International Conference on Language Resources and Evaluation (LREC)*.

Bohnet, B. (2010). Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of ACL*, pages 89–97.

Galliano, S., Gravier, G., and Chaubard, L. (2009). The ester 2 evaluation campaign for the rich transcription of french radio broadcasts. In *Tenth Annual Conference of the International Speech Communication Association*.

McDonald, R., Pereira, F., Ribarov, K., and Hajič, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP*, pages 523–530.

Nasr, Alexis, Bechet, Frederic, Rey, Jean-Francois, and Roux, Joseph Le. (2011). Macaon:a linguistic tool suite for processing word lattices. In *The 49th Annual Meeting of the Association for Computational Linguistics: demonstration session*.

Okazaki, Naoaki. (2007). Crfsuite: a fast implementation of conditional random fields (crfs).