Robust dependency parsing for Spoken Language Understanding of spontaneous speech

Frederic Bechet[1], *Alexis Nasr*[2]

[1] LIA / Université d'Avignon, 339 chemin des Meinajariès, 84911 Avignon
[2]LIF - CNRS / Université Aix-Marseille
frederic.bechet@univ-avignon.fr, alexis.nasr@lif.univ-mrs.fr

Abstract

We describe in this paper a syntactic parser for spontaneous speech geared towards the identification of verbal subcategorization frames. The parser proceeds in two stages. The first stage is based on generic syntactic resources for French. The second stage is a reranker which is specially trained for a given application. The parser is evaluated on the MEDIA corpus.

1. Introduction

The semantic interpretation of an utterance can be split into a two-level process: a translation process projecting lexical items into basic conceptual constituents and a composition process that takes as input these basic constituents and combine them in a possibly complex semantic interpretation of the utterance, represented, for example, as a set of semantic Frames. Various methods, reviewed in [6], have been proposed for both levels of this process, from statistical tagging approaches to parsing methods.

Syntactic information is useful to perform such an understanding process: at the concept level, syntax can help disambiguating concepts when computing their syntactic function in the utterance; at the semantic Frame level, syntactic dependencies can be projected into semantic dependencies to obtain structured semantic objects.

Despite its usefulness, syntactic parsing is not always considered when building a Spoken Language Understanding (SLU) system dedicated to process spontaneous speech because of two main issues: firstly transcriptions obtained through an Automatic Speech Recognition (ASR) process contain errors, the amount of errors increasing with the level of spontaneity in speech; secondly, spontaneous speech transcriptions are often difficult to parse using a grammar developed for written text due to the specificities of spontaneous speech syntax (agrammaticality, disfluences such as repairs, false starts or repetitions).

Concerning the first issue, one way of dealing with ASR errors is to take into account not only the best word string produced by the ASR process but multiple hypotheses encoded as an n-best list, a word lattice or a confusion network. The second issue leads often to develop a new parser for each new application to deploy in order to properly characterize the language register and the lexical choices used by the callers of a given speech service, for example. This is done by collecting and annotating samples characteristic of the targeted application. This task is labour intensive, needs a high level of expertise to be done properly and is one of the major bottlenecks in the deployment of spoken dialogue applications. An alternative solution is to consider, among all the resources needed to perform syntactic analysis, those that can be derived from generic resources already available and those that are linked to the applicative domain. This is, for example, the approach followed in [8] where a generic syntactic parser is used to obtain a set of parse trees on the 1-best word string of a speech transcription. A domain-specific SVM classifier based on a tree-kernel method uses all these parse trees in order to predict semantic hypotheses. The lack of precision in the syntactic analysis due to the use of a generic parser is balanced by the robustness of the SVM classifier trained on in-domain data.

We are following a similar approach in this paper, based on a 2-step process. The first step consists in using generic syntactic resources in order to parse the output of an ASR system represented as a word lattice. This process is based first on a shallow parsing process producing a lattice of syntactic chunks from a lattice of words. Then, and this is one of the originalities of this work, on the combination of two existing linguistic resources: a French tree-adjoining grammar, called FXMG [5], and a valency dictionary of French verbs called Dicovalence [11].

The second step in the process consists in reranking all the possible dependency analyses produced by the parser on each verb in the lattice. This reranking phase is done thanks to a classifier based on a boosting algorithm and trained on a corpus specific to the application targeted.

The projection from these syntactic dependencies to semantic dependencies is made by selecting first a set of *target verbs* for a given application, then associating to each subcat frames of these verbs a corresponding semantic structure for the application targeted.

The originalities of this work are the following: firstly to integrate closely the speech decoding process and the syntactic analysis by taking into account word lattices with scores produced by an ASR system; secondly the use of rich generic linguistic resources to perform dependency parsing; finally the adaptation to a specific applicative domain is only done at the reranking level, and therefore reduces very significantly the amount of corpus and expertise needed in order to build a robust parser specific to a given application.

This paper is organized as follows: in section 2 we describe the syntactic parser; section 3 describes the reranking process and section 4 presents the first evaluation of our models performed on the spoken dialogue corpus MEDIA.

This work is supported by the 6th Framework Research Programme of the European Union (EU), Project LUNA, IST contract no 33549. The authors would like to thank the EU for the financial support. For more information about the LUNA project, please visit the project home-page, www.ist-luna.eu.

2. Syntactic Parsing

The linguistic analysis of a ASR system output is classically decomposed into a series of processes that constitute a processing chain: the input of a process corresponds to the output of the preceding one.

The first modules of the chain are standard, we will quickly mention them and spend more time on the last one, the partial dependency parser.

The word lattice, produced by the ASR system is first processed by a lexical module which regroups some sequences of words to produce complex grammatical words as (*au dessus de (above) , à mesure que (as)* ...) or compound nouns (*pomme de terre (potatoe), couvre chef (hat)* ...). This module is based on the L*efff*, a large lexicon of French fledged forms [9]. In case of overlap of complex lexical units, the module produces a lattice containing all the different solutions.

This lattice is the input of a part of speech tagger which produces a lattice of part of speech tags, every tag being associated with a lexical unit of the input lattice. The tagger is based on a standard HMM which parameters have been estimated on the FrenchTreeBank [1].

The part of speech lattice is then processed by a lemmatiser which associates to every couple made of a lexical unit and a part of speech, one or several lemmas. This module is also based on the L*efff*.

The fourth step is a chunker which takes as input a part of speech lattice and produces a chunk lattice. Every chunk is a non recursive and non ambiguous grouping of part of speech tags. The chunking is realised by a cascade of finite state transducers, as in [2]. Every transducer describes the structure of a type of chunk. It specifies furthermore the head of the chunk.

The chunk lattice produced is the input of a less standard partial dependency parser, which is described in the following section.

The Partial Dependency Parser

The aim of this module is to find the syntactic arguments (subject, direct object, indirect object ...) of a predicative element, in the output of a ASR system which has been processed with the modules mentioned above. We will restrict ourselves here to the syntactic arguments of verbs. The main source of information that is used in this process is a description of the the subcategorization frames (subcat frame from now on) of verbs (whether the verb is intransitive, transitive, ditransitive, ask for a locative argument ...).

The module must simultaneously offer a good coverage, robustness and efficiency.

Coverage has two aspects, lexical coverage (the number of verbs for which we describe their subcat frames) and syntactic coverage (the different possible realizations of a subcat frame (active, passive, dative shift ...)).

Robustness is the ability to recover syntactic arguments in a noisy input. Noise comes from the specificity of syntax of spontaneous speech (disfluences, hesitations) and from the errors produced by the ASR system.

The need for efficiency comes from the fact that the lattices that are successively processed by the different modules can be rather large.

Coverage is provided by two existing resources, the syntactic lexicon Dicovalence [11] and the French Tree Adjoining Grammar (TAG) FXMG [4]. Robustness and efficiency are realized through an underspecified representation of the subcat frames by means of finite state transducers.

These aspects are successively described in the two following sections.

Lexical and Syntactic Coverage

Dicovalence is a syntactic lexicon of French verbs. It describes the subcat frames of more than 3700 verbs. Dicovalence relies on the pronominal approach [10]: the syntactic arguments of a verb are given in their pronominal realizations. For every verb, the list of pronouns is given for every syntactic argument of the verb.

Dicovalence offers a good lexical coverage (it covers $89,3\%$ of the verbs of the French TreeBank) but its syntactic coverage is by nature restricted, since only the pronominal realization of syntactic arguments are described. In order to augment the syntactic coverage, one must associate to every subcat frame its non pronominal realization. This step is realized by the use of the large coverage grammar FXMG.

FXMG is a TAG automatically produced from a metagrammar [4]. We will not describe here the TAG formalism but only mention some key features. A TAG associates to every lexical entry a set of *elementary tree* that describe, among other syntactic characteristics, a possible realization of a subcat frame of the lexical entry. Elementary trees are grouped together in *families*. A family groups all the syntactic variation of a given subcat frame. FXMG defines 7600 types of elementary trees, grouped into 92 families. Families are of various sizes, as an example, the family of the transitive verbs is made of 159 elementary trees. Every family is associated to a tag that describes in a concise way the number and nature of the syntactic arguments. The tag corresponding to the transitive family, for example, is `n0Vn1`, the family of di-transitive verbs which indirect object is introduced by the preposition *à* is `n0Vn1an2`, and so on.

[5] reports a syntactic coverage of 75% of the TSNLP corpus [7]. This can be considered as a good result given the characteristics of this corpus: the frequencies of the syntactic constructions do not correspond to their usage frequencies, in other words, rare constructions are over represented.

FXMG defines a set of elementary trees, grouped into families but does not map lexical entries to families. It can clearly be seen at this point that Dicovalence and FXMG are complementary. The former provides a good lexical coverage while the latter offers good syntactic coverage.

In order to map Dicovalence subcat frames to FXMG tree families, correspondence rules are defined that map a couple *(syntactic function, pronoun)* to a subpart of a family tag. These rules are applied to a Dicovalence subcat frame and produce family tag subparts that are concatenated together[1].

Tree families as automata

We have described in the preceding section how the two resources Dicovalence and FXMG are mapped together. Once this mapping is done, we have at our disposal the tree families for the 3700 verbs of Dicovalence. These families describe all the realization of their possible subcat frames. In order to efficiently detect the occurrence of a subcat frame in the chunker output, every elementary tree is represented as a finite state automaton and a family as the union of its elementary trees automata.

The transformation of an elementary tree into an automaton does not preserve all the syntactic constraints that are encoded in the elementary tree. More precisely, the only information that is kept in the automaton is the number of syntactic arguments, their nature and their relative order. This relaxation of syntac-

[1] The process is actually more complex but we will omit technical details.

tic constraint increases robustness but can lead to erroneous attachment. In order to limit this effect, weights are added to the transition of the automata. These weights allow to implement a simple heuristic: the higher the number of syntactic arguments of a subcat frame and the closer these arguments are to the verb, the more this frame is favoured.

The process of instanciating a subcat frame in a chunk lattice is realized by means of automata composition. More precisely, given the chunk automaton T_C and an automaton for each elementary tree family, noted $T_{F_1}, \ldots, T_{F_n}$, these n automata are successively composed with T_C and the result of these compositions are then combined by means of the union operator.

The result is therefore $\cup_{i=1}^{n} T_C \circ T_{F_i}$[2]. This model can be contrasted with a transducer cascade ($T_C \circ T_{F_1} \circ \ldots \circ T_{F_n}$). The difference between both is important for sentences containing several verbs. In our case, the verbs are processed independently and the subcat frames found for each verb can be incompatible: a chunk can be the argument of several verbs. This choice is motivated by efficiency. Indeed, the syntactic constraints relaxation that was performed can lead to an unreasonable number of solutions, and therefore to an unreasonable size of the automata when implementing a cascade.

3. Dependency parse reranking

The dependency parsing process presented in the previous section is based only on generic linguistic resources available for French; no training on domain specific data is required. The output of this parsing process is a set of dependency hypotheses for each verb in the lattice. Each hypothesis corresponds to an instantiated subcat frame. Until now, the process does not include any semantic or pragrmatic constraint, which is often mandatory in order to select correct analyses. We believe that such information is specific to the task at hand, and cannot rely on generic resources but rather on domain specific ones.

Such domain adaptation is done in this study by means of a reranking process trained on domain specific data. This process is based on a classifier trained to select the relevant dependency analyses among all the hypotheses produced for a set of *target verbs*. These target verbs are selected as being crucial in obtaining a semantic representation for a spoken utterance in the context of a speech service (for example query verbs). We use a large margin classifier, called **IcsiBoost**[3], dedicated to process textual data. This classifier is based on a boosting algorithm of weak classifiers (one-level decision trees on the occurrence of ngrams of tokens). The training of this reranking classifier is done as follows:

1 - A corpus of utterance corresponding to the targeted application is collected, manually transcribed and processed by our parser as presented in section 2.

2 - For each utterance, and for each target verb the n-best list of dependency hypotheses is given to a human judge who accepts the correct analyses and discard the erroneous ones.

3 - All the dependency hypotheses produced for each utterance of the training corpus are grouped and labelled by a tag indicating their correctness. Each hypothesis is represented by the string of words supporting it with a label attached to each word corresponding to its function according to the syntactic dependency hypothesised. The classifier is then trained on such a corpus in order to separate the examples considered as correct by the human judges from the incorrect ones.

At runtime, when processing a new utterance, this classifier processes the dependency hypotheses produced by our parser on the ASR word lattice and gives a score to each of them of being correct. This score is used to rerank the hypotheses and those having a score above a given threshold are kept.

Although this domain adaptation requires collecting and transcribing speech samples characteristic of the application targeted, the human cost of manually validating dependency hypotheses is much smaller than the one needed to manually annotate a corpus or to adapt a given grammar to a particular application.

4. Evaluation

Our parsing method has been evaluated on the French Spoken Dialogue corpus MEDIA. The MEDIA corpus [3] has been recorded using a *Wizard of Oz* system simulating a telephone server for tourist information and hotel booking. Eight scenario categories were defined with different levels of complexity. The corpus contains 1257 dialogs from 250 speakers (about 70 hours of speech).

Our dependency parser has been applied to the reference and the ASR transcriptions of MEDIA. The ASR system used is the SPEERAL system developed at the University of Avignon. A word lattice has been obtained for each utterance of the MEDIA corpus. In this first evaluation we have restricted the evaluation of the syntactic dependencies to the target verb *réserver (to book)* as it is the most relevant one in the hotel booking framework of MEDIA.

The MEDIA corpus has been split in three corpora: a *training corpus* on which the ASR models and the reranking classifier have been trained; a *development corpus* on which the different parameters of the ASR module and the reranking classifier have been tuned; a *test corpus* on which we have evaluated our method. These three corpora are described in the table below.

Corpus	**Train**	**Dev.**	**Test**
# dialogue	727	79	208
# turn	12988	1265	3524
# target verb	659	73	187
WER	14.5	25.3	27.4

These three corpora have been processed by our parser. Every occurrence of the target verb is associated to a (possibly empty) n-best list of instantiated subcat frame hypotheses. Each dependency hypothesis has been manually validated, as presented in section 3, in order to train the reranking classifier on the training corpus, tune its parameter and choose the acceptance threshold on the development corpus and evaluate the overall performance on the test corpus. Let's recall that the dependency parser hasn't been trained at all on the MEDIA corpus, only the reranker is trained.

Three results are compared:

- The *Oracle* results consists in always choosing the correct dependency hypotheses among the n-best list output by the parser, if any of them are correct. This represent the upper bound that would reach a perfect reranker on these n-best lists.
- The *Baseline* results are obtained by always choosing the first dependency hypothesis output by the parser. The

[2] In practice, we only keep the n best paths of every automaton produced by a composition. The formula is therefore: $\cup_{i=1}^{n} \mathrm{nbest}(T_C \circ T_{F_i}, n)$ where $\mathrm{nbest}(A, n)$ is the operation that produces the automaton made of the n minimum weight paths of automaton A.

[3] http://code.google.com/p/icsiboost/

score of the different hypotheses is the linear combination of the ASR, the Part-Of-Speech tagger and the parser scores.

- The *Reranking* results are obtained thanks to our reranking classifier trained on the MEDIA corpus.

The first results given in the table below are the precision (**P**), recall (**R**) and F-measure (**F**) on the detection of *full* dependency analyses for each target verb. This means that a dependency hypothesis is considered as correct if the subcat frame chosen for the target verb is correct and if *all* its dependencies are correct at the word, chunk and functional levels. Experiments have been conducted on human transcription and ASR transciption.

Corpus	**ref. trans.**			**ASR trans.**		
results	**P**	**R**	**F**	**P**	**R**	**F**
Oracle	90.0	90.0	90.0	63.2	63.2	63.2
Baseline	42.8	38.3	40.4	34.7	29.2	31.6
Rerank.	59.9	70.8	64.9	45.9	53.1	49.2

The following table presents the evaluation of the *Baseline* and *Reranking* systems at the constituent level. In this table a constituent is correct if the word, chunk and functional levels are correct.

Corpus	**ref. trans.**			**ASR trans.**		
results	**P**	**R**	**F**	**P**	**R**	**F**
Baseline	88.8	76.6	82.4	78.2	63.5	70.1
Rerank.	91.1	90.2	90.6	80.4	76.0	78.2

From these results several conclusions can be drawn: firstly the coverage of the generic parser is satisfactory as the Oracle accuracy on the reference transcriptions of the corpus is 90% for the target verbs. This means that using generic linguistic resources is efficient for parsing spontaneous speech if some linguistic constraints are loosen (use of multiple ASR output and local parsing for finding the constituent dependencies). Secondly it is crucial to integrate domain specific knowledge in the decision process in charge of choosing the correct analyses among the n-best list provided by the parser. The results of the *Baseline* system illustrate this point and clearly demonstrate that ASR and semantic ambiguities can only be solved by means of domain-dependent models. Thirdly the impact of the ASR errors is important, especially at the full dependency parse level, as the evaluation presented here is very strict: any ASR error in one the constituent of a dependency hypothesis will lead to consider the whole hypothesis as erroneous. Nevertheless half of the dependencies are correctly retrieved for the target verbs, and the evaluation at the constituent level clearly shows that this method is efficient enough to be integrated in a SLU system as a F-measure of 78% is achieved at the constituent level (word+chunk+function). The next step will be to integrate in the reranking classifier more domain dependent data, as for example a concept ontology that can be used to filter the constituent hypotheses and add some semantic constraints on the different analyses produced.

5. Conclusions

We describe in this paper a syntactic parser for spontaneous speech geared towards the identification of verbal subcategorization frames. The parser proceeds in two stages. The first stage is based on generic syntactic resources for French. The second stage is a reranker which is specially trained for a given application. The parser is evaluated on the MEDIA corpus. The two main features of the method proposed is firstly to integrate closely ASR and syntactic parsing by processing ASR word lattices and secondly to use rich generic linguistic resources and limited in-domain adaptation data in order to reduce very significantly the amount of corpus and expertise needed to build a robust parser specific to a given application.

6. References

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

[2] Steven Abney. Partial parsing via finite-state cascades. In Workshop on Robust Parsing, 8th European Summer School in Logic, Language and Information, Prague, Czech Republic, pages 8–15., 1996.

[3] Helene Bonneau-Maynard, Sophie Rosset, Christelle Ayache, Anne Kuhn, and Djamel Mostefa. Semantic annotation of the french media dialog corpus. In *European Conference on Speech Communication and Technology (Eurospeech)*, Lisboa, Portugal, 2005.

[4] Benoit Crabbé. Grammatical development with xmg. In *Logical Aspects of Computational Linguistics*, 2005.

[5] Benoît Crabbé. *Représentation informatique de grammaires fortement lexicalisées, application à la grammaire d'arbres adjoints*. PhD thesis, Université Nancy 2, 2005.

[6] R. De Mori, F. Bechet, D. Hakkani-Tur, M. McTear, G. Riccardi, and G. Tur. Spoken language understanding. *Signal Processing Magazine, IEEE*, 25(3):50–58, May 2008.

[7] Sabine *et al* Lehmann. TSNLP: Test suites for natural language processing. In *Proceedings of the 16th conference on Computational linguistics*, pages 711–716, 1996.

[8] A. Moschitti, G. Riccardi, and C. Raymond. Spoken language understanding with kernels for syntactic/semantic structures. In *ASRU. IEEE Workshop*, pages 183–188, Dec. 2007.

[9] Benoît Sagot, Lionel Clément, Érice Villemonte de la Clergerie, and Pierre Boullier. The lefff 2 syntactic lexicon for french: architecture, acquisition, use. In *International Conference on Language Resources and Evaluation*, Genoa, 2006.

[10] Karel van den Eynde and Claire Blanche-Benveniste. Syntaxe et mécanismes descriptifs : présentation de l'approche pronominale. *Cahiers de lexicologie*, 32:63–104, 1978.

[11] Karel van den Eynde and Piet Mertens. La valence: l'approche pronominale et son application au lexique verbal. *Journal of French Language Studies*, 13:63–104, 2003.