Syntactic parser combination for improved dependency analysis

Francis Brunet-Manquat
Laboratoire CLIPS-IMAG
CNRS – UJF - INPG
385, rue de la Bibliothèque
BP 53 – 38041 Grenoble Cedex 9, France
Francis.Brunet-Manquat@imag.fr

Abstract

The goal of this article is to present our work about a combination of several syntactic parsers to produce a more robust parser. We have built a platform which allows us to compare syntactic parsers for a given language by splitting their results in elementary pieces, normalizing them, and comparing them with reference results. The same platform is used to combine several parsers to produce a dependency parser that has larger coverage and is more robust than its component parsers. In the future, it should be possible to "compile" the knowledge extracted from several analyzers into an autonomous dependency parser.

1 Introduction

Our laboratory is involved in two international projects: C-STAR (Blanchon and Boitet 2000), with its associated European project NESPOLE!, for speech translation and UNL, Universal Networking Language (Sérasset and Boitet 2000), for written translation. These two projects are characterized by the use of a pivot representation of the utterance and by the fact that the utterance to be translated is likely to be "ill-formed", i.e. not conform to a academic language grammar. In a pivot system, an utterance in a source language is parsed to yield a pivot representation which generate into a target language is performed. To process ill-formed data, we need robust analysis tools capable of producing a partial analysis.

The goal is to specify, design and develop a multilingual platform, called DepAn (Dependency Analysis), which compares parsers for a given language by splitting their results in elementary pieces, normalizing them, and comparing them with reference results, and which combine these parsers to produce a dependency parser that has larger coverage and is more robust than its component parsers.

The platform combines several analyses of the same utterance, and then computes the best data to produce the best possible analysis. Our approach is based on the method called "vote by majority", *the more common to the different parsers one data will be, the stronger its weight will be*, and also based on a training method which adapts each *vote* according to the typologies of utterances (domain, style) and the abilities of the parsers.

The approach used, called *combination approach*, has known lots of success in speech recognition (Fiscus 1997, Schwenck and Gauvain 2000), part of speech tagging (Halteren and al. 1998, Brill and al. 1998, Marquez et Padro 1998), named entity recognition (Borthwick and al. 1998), word sense disambiguation (Pedersen, 2000) and recently in parsing (Henderson and Brill 1999), Inui and Inui 2000, Monceaux and Robba 2003). These works prove that combining different systems provides an improvement in comparison to the best system.

Our work in syntactic analysis are distinguished from our predecessors by the combination methods that we use. Our platform is made up of a statistical processing, a correspondence processing and a reconstruction processing. Furthermore, we base our platform on a dependency representation that describes the syntactic relations between words. A study realized within the framework of the international projects, CSTAR and UNL, suggests that this representation type is adapted to a robust and partial parsing.

2 Analysis platform design

The platform must not integrate the parsers, but it must be able to extract the linguistic data from their analyses, interpret them, combine them, and produce a dependency tree (or several) combining the best extracted data.

2.1 Processing steps

The platform process comprizes two stages: the *standardization* of the analysis results and the

construction of the dependency analysis (see Figure 1).

The standardization is made up of two steps:

- The *extraction step* permits to recover the linguistic data of analysis produced by the linguistic parsers. These parsers are shared out in three groups according to their analysis results (Monceaux and Robba 2002): *the parsers based on the chunks* which segment the sentence in syntagms (chunks), *the parsers based on the dependencies* which produce dependencies between words of a sentence, and *the parsers based on the chunks and the dependencies* which segment the sentence in syntagms and produce dependencies between syntagms and words.
- The *projection step* process the extracted data to produce a set of dependency structures, called *standardized structures*. a rate is associated at each data (pos, syntactic relations, etc.) according to the parser which produces it. These rates, called *confidence rates*, are pre-calculated during a training step (see 2.3 Confidence rate). A dependency structure is described by a matrix representation offering both handiness and efficiency (see 2.2 Dependency matrix).

The construction is made up of three steps:

- The *correspondence step* links the nodes of the different normalized structures provided by the previous step. So, we create a structure, called *segmentation network (SN)*, which represents the different segmentations of a sentence and the links between the nodes of the normalized structures. This network represents the "*pivot link*" between these structures (see 3.1 Correspondence of the dependency structures).
- The *combination step* according to established links produces a single dependency representation which contains all the extracted linguistic data. The resulting data can be such as inconsistancies i.e. a word can't be both a noun and a verb (contradictory part-of-speech). The confidence rate of these data are then recalculated (see 3.2 Combination of the linguistic data).
- The *production step* builds the new dependency structures according to the combined data, their new confidence rates, and some linguistic and structural constraints (see 3.3 Production of the dependency structures).

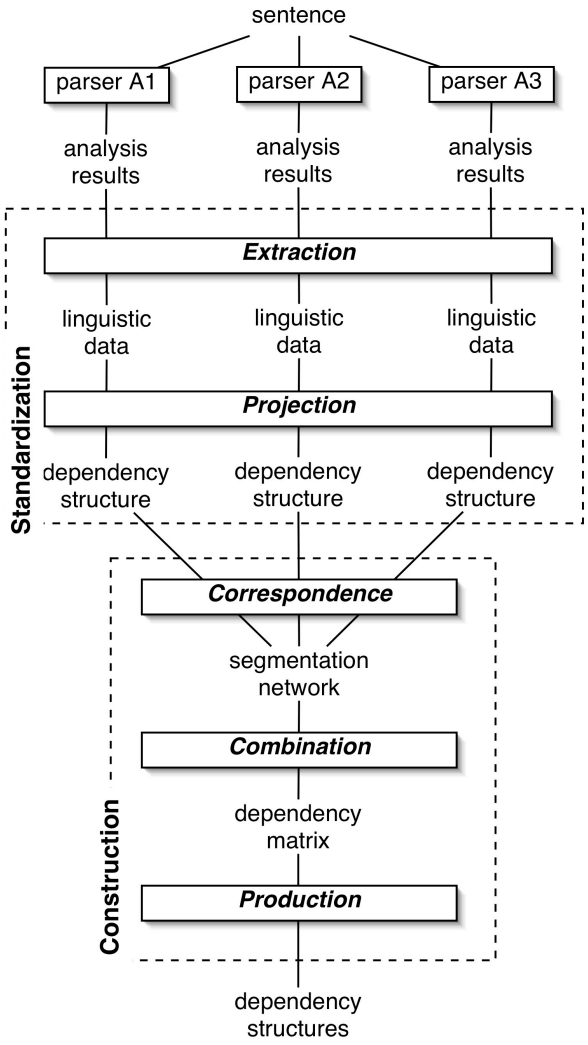


Figure 1: Functional architecture

2.2 Dependency matrix

Our analysis platform is based on the dependencies, i.e. it produces dependencies between the words of a sentence. In our platform, a dependency structure is described by a matrix representation. Our representation, called Dependency Matrix (DM), is made up of a couple <*L,M*>:

- *L* is a list of nodes. A node is made up of a set of linguistic data (part of speech and grammatical variable). Each node represents a word.
- *M* is a square matrix which describes the dependencies between the nodes of *L*. M(i,j) contains the set of syntactic dependencies between i node and j node of *L*.

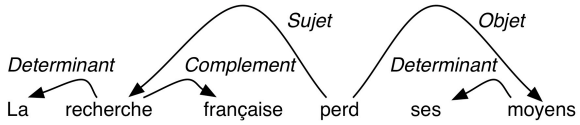


Figure 2: Syntatic dependency structure

The DM corresponding to the syntactic dependency structure above is:

L =

la	::	pos=determinant
recherche	::	pos=noun
française	::	pos=adjective
perd	::	pos=verb
ses	::	pos=determinant
moyens	::	pos=noun

M =

Dependency ▶

	la	recherche	française	perd	ses	moyens
la						
recherche	Determinant		Complement			
française						
perd		Subjet				Objet
ses						
moyens					Determinant	

A matrix representation has two advantages for the automatic process:

- *Handiness*: mathematic tools are associated to matrix: addition, deletion, comparison, etc. These tools permit a simple processing of the data contained in the matrix.
- *Effectiveness*: efficient methods are associated to matrix: pattern matching methods, combination methods, etc.

We choose also to use a matrix representation because the combination of different dependency structures provides a graph containing all the possible dependencies.

2.3 Confidence rate training

In (Brunet-Manquat 2003), we present projection rules to transform the extracted data into a set of normalized data. Each normalized data D is associated to a confidence rate of the data D according to the parser which produces it.

The rates are calculated according to the parser evaluations. For each parser A_i, we calculate the recall and the accuracy of each linguistic data D produced by A_i:

$$\text{Recall}_{A_i}(D) = \frac{\text{number of corrected data } D}{\text{number of reference data } D}$$

$$\text{Accuracy}_{A_i}(D) = \frac{\text{number of corrected data } D}{\text{number of nominated data } D}$$

The confidence rate corresponds to the F-measure which combines recall and accuracy in a single measure:

$$\text{F - measure}_{A_i}(D) = \frac{(\beta^2 + 1) \times \text{Accuracy}_{A_i}(D) \times \text{Recall}_{A_i}(D)}{\beta^2 \times \text{Accuracy}_{A_i}(D) + \text{Recall}_{A_i}(D)}$$

Where β is an accuracy coefficient, $1 \geq \beta \geq 0$.

Within the framework of our work, we want to be able to set our platform according to our needs in analysis: information retrieval, machine translation, parsing, etc. We introduce the *accuracy coefficient* β (between 0 and 1) into the F-measure. It permits to customize the platform and so the final analysis: if β is close to 0, the accuracy will be favoured in the calculation of the confidence rate. In the following, β will be equal to 1. It will be interesting to introduce a second coefficient for the recall (evaluation in progress).

3 Dependency structure construction

At the end of the normalization process, a set of dependency strutures is associated to each sentence. The next step consists in combining these structures to obtain a single dependency representation which contains all the linguistic data of these structures. To perform this combination, we must put in correspondence these dependency structures.

3.1 Correspondence of the dependency structures

The structure correspondence consists in regrouping the nodes representing the same word into a sentence (the shared minimal data). But it consists also in representing the word conflicts produced by the different segmentations of a sentence because of, for example, compound words (words *high energy* or word *high-energy*), dictionnary terms (words *United Kingdom* or word United_Kingdom), etc.

In order to represent the correspondences, we create a structure, called *segmentation network (SN)*, that represents the different segmentations of a sentence and links the nodes of the normalized structures. This network represents the "pivot link" between these structures.

A SN is a lattice; each node of this lattice represents a possible segment of a word and serves to link the nodes of the dependency structures. In practice, a node N_{sn} of a SN is made up of two data:

- SNODE: a sequence which represents a substring of a sentence. For example, the words of the sentence *"On avait dénombré cent vingt-neuf candidats"* have to SNODE: *On[1,2]*, *avait[3-7]*, *dénombré[8-15]*, etc. This

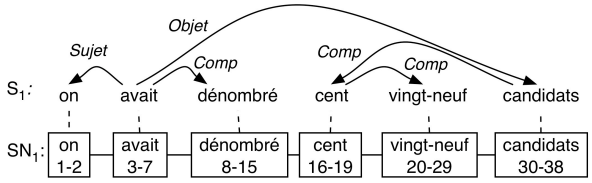
data is based on the proposal of (Boitet and Zaharin,1988) *Structured String-Tree Correspondences (SSTC).*

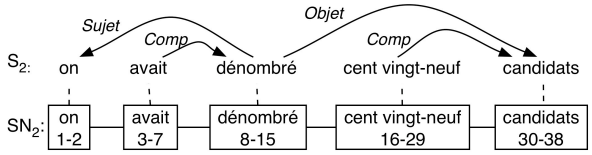
- L: a set which contains the nodes of the normalized structures linked to the node N_{rs}.

The first step consists in creating an initial SN for each dependency structure. Each initial node N_{sn} of an initial SN is created according to a node N_i of the dependency structure S_k:

$SNODE(N_{sn})=SNODE(S_k.N_i)$ & $L(N_{sn})=\{S_k.N_i\}$

Then the nodes of the initial SN are inserted into the lattice according to their appearance order into the sentence (according to their SNODE). In the following, we take two dependency structures S_1 and S_2, and their initial segmentation networks SN_1 and SN_2:

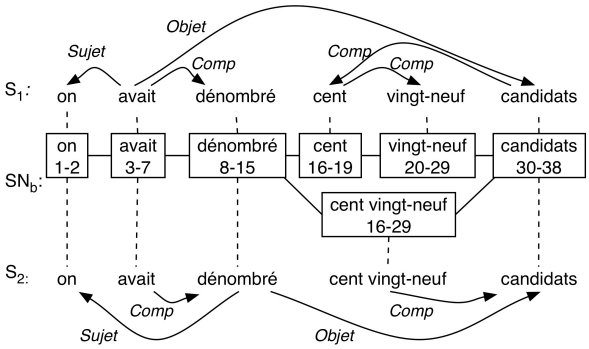




Let us do the correspondences between SN_1 and SN_2. First, the initial network SN_1 is chosen as the basic SN, called SN_{base}. We use two rules to introduce the nodes of the others SN into the basic SN:

- Rule 1) **Correspondence**: If a node N_i of SN_k is equal to a node N_{sn} of the SN_{base} (equal if $SNODE(N_i)==SNODE(N_{sn})$), N_{sn} will be linked to the node N_i: $L(N_{sn}) = L(N_{sn}) \cup L(N_i)$.
- Rule 2) **Insertion**: If a node N_i of SN_k is not equal to a node of SN_{base}, this node will be inserted in SN_{base} according to their SNODE.

The first nodes *on[1-2]*, *avait[3-7]*, *dénombré[8-15]* of SN_2 verify the first rule. They correspond to the nodes *on[1-2]*, *avait[3-7]*, *dénombré[8-15]* of SN_{base}. The fourth node *cent vingt-neuf[16-19]* of SN_2 verifies the second rule, so it is inserted into SN_{base}. The last node of SN_2 *candidats[30-38]* verifies the first rule. We obtain the following lattice:



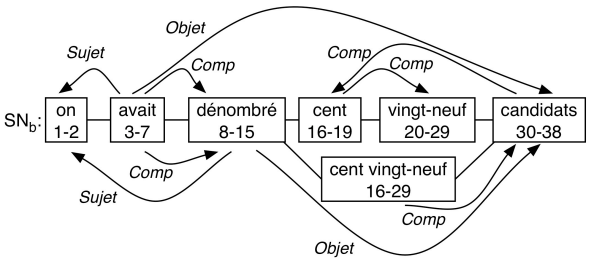
The final segmentation network represents the possible segmentations of the sentence and links the nodes of structures between them[1]. Now the correspondences between the nodes of the structures are established, we can combine these structures to provide a single dependency representation, which combine all linguistic data of these structures.

3.2 Combination of the linguistic data

The correspondences between the different structures being established, the combination step of linguistic data can begin. The method used here is based on the method known as "majority vote": *the more common to the different parsers one data will be, the stronger its weight will be.*

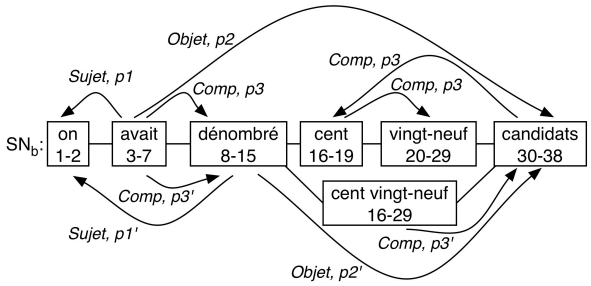
At the end of the correspondence phase, a set of dependency structures and a segmentation network SN are associated to every sentence. The first stage consists in creating a dependency structure, called *combined matrix CM* (the nodes of the SN will be used as nodes for this representation) for each segmentation network. This matrix is filled with the linguistic data contained in the associated dependency structures.

For example, we regroup dependencies for the previous SN_b:



[1] In (Brunet-Manquat 2004), we propose to improve the correspondence step by adding correspondence rules allowing processing the compound words or dictionary terms, for example, by establishing a relation between the node *United_Kingdom* and the nodes *United* and *Kingdom*.

Subsequently, we associate a confidence rate *p* for each data:



Each confidence rate could be seen as a *weighted vote* of a parser A_i for a data D. During a training phase, these rates (votes) will be adapted to the different abilities of the parser according to the typologies (domain, style) of the reference utterances.

Some linguistic data will be equivalent, some others will be contradictory (for example the dependency Subject(x, y) is contradictory with the dependency Object(x, y), the part-of-speech are mutually contradictory into a same word).

Finally, we group all the linguistic data D_i of a sentence S (D_i is the data D provided by the parser i), and we calculate the new associated confidence rate, called *combined rate*, for each data D. A combined rate of a data D is calculated according to the confidence rates of all the data D. We propose two calculations: *standardized* and *corrected* calculations.

Standardized calculation: the combined rate of a data D is equal to the sum of the confidence rates of the data D_i divided by the number n of parsers that can provide this data D:

$$R_{combined}(D) = \frac{(\sum_i R_{confidence}(D_i))}{n}$$

Where i = parser producing the data D;
n = number of parsers that can provide the data D.

For example, let us calculate the combined rate associated to the dependency OBJ(x, y) (the word y is the object of the word x) provided by the parsers A1 and A2. The combined rate associated to OBJ(x, y) is equal to the sum of the two confidence rates provided by A1 and A2: confidence(OBJ::A1)=0.5 and confidence (OBJ::A2)=0.7, divided by the number of parsers that can provide this type of information (three for the example), (0.5+0.7+0)/3 = 0.4. If the third parser provides an other dependency, for example SUBJ (x, y) (the word y is the subject of the word x), and if the confidence rate of this data is 0.8, the merged rate associated to SUBJ(x, y) is equal to (0+0+0.8)/3 = 0.26.

Corrected calculation: the combined rate of the data D is equal to the sum of the confidence rates of data D_i minus the sum (multiplied by a correction coefficient) of the confidence rates of the data contradictory to D, the whole divided by the number of parsers that can provide the data D:

$$R_{combined}(D) = \frac{(\sum_i R_{confidence}(D_i) - \alpha \times \sum_p R_{confidence}(D_p))}{n}$$

Where i = parser producing the data D;
n = number of parsers that can provide the data D;
p = parser producing a date contradictory to D.
α = an correction coefficient, $1 \geq \alpha \geq 0$.

For the previous example, the syntactic dependencies between words x and y are contradictories: either OBJ(x, y), or SUBJ(x, y). The combined rate associated to OBJ(x, y) is equal to ((0.5+0.7) - (0.4*0.8))/3 = 0.29 (with a correction coefficient at 0.4) and the combined rate associated to SUBJ(x, y) is equal to (0.8 – 0.4*(0.5+0.7))/3 = 0.1.

These two calculations favour the linguistic data provided by the greatest number of parsers. The corrected calculation permits to treat both the silence and the contradiction of others parsers.

3.3 Production of the dependency structures

This last step permits to build the new dependency structures according to the data combined in the previous step. These structures are produced according to the combined rate associated to these data, and linguistic and structural constraints. The production is based on a constraint satisfaction method made up of three rules:

Let D be a part-of-speech or a grammatical variable of a combined matrix CM:

- For each node N of CM, D is kept if its combined rate is higher than the combined rates of the contradictory data.

Let D be a syntactic relation of CM:

- Only one syntactic dependency between two nodes N_i et N_k is kept: The data with the best confidence rate on the case CM(i, k) is kept;
- A node N_i depends on only one node N_k: In the column CM(i), which represents the dependencies $N_k \rightarrow N_i$, only the data with the best confidence rate is kept.

Concerning the word conflicts resulting from the different possible segmentations of a sentence, we choose to keep only the nodes resulting from the "best" word chunker among our parsers, i.e. the parser with the word segmentation closer to the segmentation of the reference corpus. Soon we will introduce a process that associates a segmentation rate to each node, which represents the confidence

on the word segmentation according to the parsers, like the confidence rate on the linguistic data. This rate will permit us to introduce a segmentation constraint in our production step.

4 Experimentation and measures

4.1 Parsers and corpus

We experiment our platform on French. We have three parsers for this evaluation: IFSP (Incremental Finite-State Parser) (Aït-Mokhtar and Chanod 1997) which builds the syntactic groups (chunks) of a sentence, and then uses the structure built to extract the syntactic dependencies between words, the parser of the GREYC (Vergne 1998) which combines tagging methods to build not-recursive chunks and a dependency algorithm to calculate the dependency structure and XIP (Xerox Incremental Parser) (Haït-mokhtar and al. 2002) which has different linguistic processings organized in an incremental way (morphological tagging, chunk parsing, dependency extraction) to obtain an dependency analysis.

The corpus used is the corpus of the university Paris VII (Abeillé and Clément 1999). This corpus is made up of a million sentences extracted from "*Le Monde*", a French newspaper. The sentences are chunked and the words tag. A small part of this corpus was standardized to correspond to a dependency corpus. For this experimentation, we use a reference corpus made up of 400 sentences, arbitrarily selected, made up of long and complex sentences, 30 words on average per sentence (minimum 9 words, maximum 73 words). For example:

« *La cessation de paiement constatée, le tribunal de commerce nomme un administrateur judiciaire, qui doit évaluer les dettes - alors gelées - et proposer soit un plan de continuation, soit la liquidation judiciaire*. »

4.2 Training

The first 200 sentences of the reference corpus are used in the training step. Our experimentation is restricted with 10 linguistic data: 6 part-of-speech (noun, verb, adjective, pronoun, preposition and determinant) and 4 syntactic dependencies (subject, object, complement and determinant). Figure 3 represents the global measures for the part-of-speech. Figure 4 represents the global measures for the syntactic dependencies.

The syntactic dependency measures are bad because the word average per sentence of the reference corpus is high and also because the sentences are difficult to analyze. The confidence rates (F-measure) of each part-of-speech (for example, F-measure of noun pos: IFSP: 78,4%, GREYC: 77,8%, XIP: 79,9%) and each syntactic dependency (for example, F-measure of subject pos: IFSP: 50,0%, GREYC: 36,4%, XIP: 50,5%) enable us to produce our combination results on the 200 remaining sentences (see 3 Dependency structure construction).

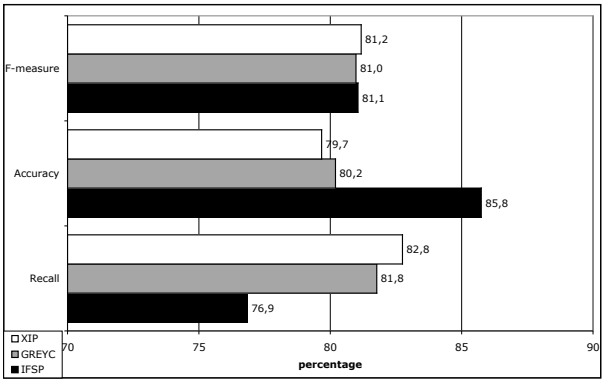


Figure 3: Part-of-speech measures

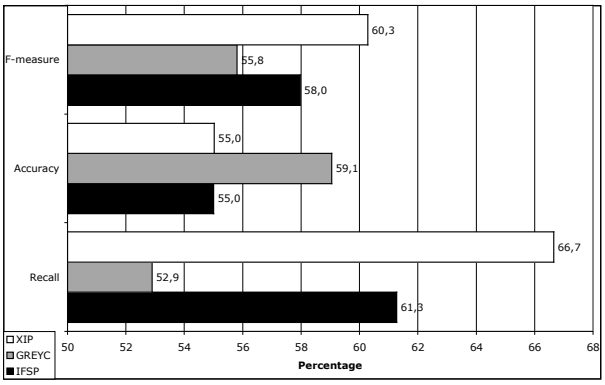


Figure 4: Syntactic dependency measures

4.3 Evaluation

We evaluate our platform analysis and the other parser analysis. The platform DepAn uses the standardized calculation to combine the linguistic data (see 3.2 Combination of the linguistic data).

The evaluation of part-of-speech tagging (see Figure 5) shows that our approach permits a significant gain of 2.6% in comparison to the best parser according to the F-measure (DepAn: 86,3% and GREYC: 84,1%). However, the gain is not significant for the evaluation of syntactic dependency (see Figure 6). It is equal to 1.1% in comparison to the best parser (DepAn: 62,9% and XIP: 62,2%).

This weak gain is understandable because the standardized calculation used to combine all the data *D* does not consider the data contradictory to *D*. To improve the analysis, we propose another calculation combining the linguistic data, the corrected calculation (see 3.2 Combination of the linguistic data). This calculation permits to treat both silences and contradictions of others parsers.

The evaluation of the platform with the corrected calculation is currently in progress.

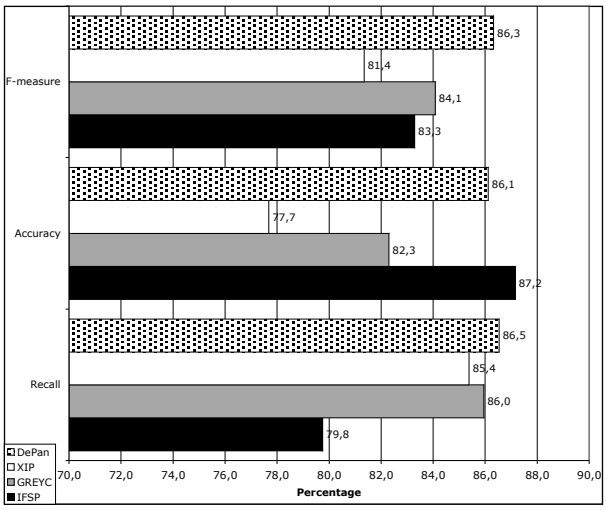


Figure 5: Part-of-speech measures

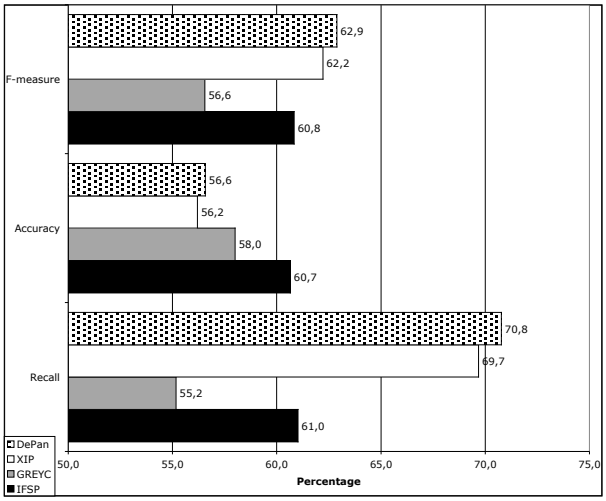


Figure 6: Syntactic dependency measures

5 Conclusion

The platform DepAn allows us to compare syntactic parsers for a given language by splitting their results in elementary pieces, normalizing them, and comparing them with reference results. The same platform is used to combine several parsers to produce a customized dependency parser, which combines the different abilities of these parsers and which is adapted to the style or the domain of the reference utterances.

The evaluations show that our approach, a combination processing associated with a statistical processing, improve the analysis in comparition to the used parsers. The gain is not significant for the moment but the future corrections will improve this gain.

Our platform is currently tested on English. We use the SUSANNE (http://www.grampson.net) corpus. The SUSANNE Corpus was created, with the sponsorship of the Economic and Social Research Council (UK), as part of the process of developing a comprehensive language-engineering-oriented taxonomy and annotation scheme for the logical and surface grammar of English. The SUSANNE Corpus itself comprises an approximately 130,000-word subset of the Brown Corpus of American English.

In the short term, we also hope to combine other parser types (semantic for example) to the syntactic parsers to produce multilevel dependency structures containing several linguistic levels: semantics, logic, syntactic, etc. In the future, we hope to learn from the combination of several parsers. For example, it should by possible to "compile" the knowledge extracted from these parsers into an autonomous dependency parser.

6 Acknowledgements

We would like to express our special thanks to all the creators of the parsers used here for enabling all of this research by providing their systems to us.

References

Abeillé A. and L. Clément (1999). *A tagged reference corpus for French*, LINC'99 Proceedings, EACL workshop, Bergen.

Aït-Mokhtar S. and Chanod JP. (1997), *Incremental finite-state parsing*, in Applied Natural Language Processing *1997*, April 1997, Washington.

Ait-Mokhtar S., Chanod JP. and Roux C. (2002), *Robustness beyond Shallowness: Incremental Deep Parsing*, in Natural Language Engineering, 8 (2/3), pp 121-144, Cambridge University Press.

Blanchon H. and Boitet C. (2000). *Speech Translation for French within the C-STAR II Consortium and Future Perspectives*. Proc. ICSLP 2000. Beijing, China, Oct. 16-20, 2000. vol 4/4: pp. 412-417.

Boitet Ch. and Zaharin Y. (1988), "*Representation trees and string-tree correspondences*", published in COLING-88, pp 59-64.

Brill E. and Wu J. (1998) *Classifier Combinaison for Improved Lexical Disambiguation*. In Proc. of the 17th COLING, pp. 191-195.

Brothwick A., Sterling J., Agichtein E. and Grishman R. (1998) *Exploiting diverse knowledge sources via maximum entropy in named entity recognition*. Proceedings of the sixth workshop on very large corpora, pages 152-160, Montreal.

Brunet-Manquat F. (2004), "*Description et conception d'une plate-forme robuste combinant*

des analyseurs d'énoncé", journal on line ISDM, vol. 13, février 2004, 12 pages.

Brunet-Manquat F. (2003), "*Fusionner pour mieux analyser: quelques idées et une première expérience*", In Proc. Of RECITAL-2003, Batz-sur-mer, France, 10-14 juin 2003. Vol 1/2, pp 429-438.

Fiscus J.G. (1997), "*A post-processing system to yield reduced error word rates: Recognizer output voting error reduction (ROVER)*", published in IEEE Workshop on Automatic Speech Recognizer and Understanding, pp 347-354.

Halteren H., J. Zavrel and W. Daelemans (1998). *Improving data driven wordclass tagging by system combination*. In Proc. of the 17th COLING.

Henderson, J. C. and Bril E. (1999). *Exploiting Diversity in Natural Language Processing: Combining Parsers*. In Proc. of the 1999 SIGDAT Conference on EMNLP and VLC, pp. 187-194.

Illouz G. (1999), "*Méta-étiqueteur adaptatif: vers une utilisation pragmatique des resources linguistiques*", published in TALN'99.

Inui T. and Inui K. (2000), *Committee-based Decision Making in Probabilistic Partiel Parsing*, In Proc. of COLING-2000.

Marquez and Padro (1998). *On the evaluation and comparaison of taggers : the effect of noise in test corpora*. Actes COLING/ACL'98, Montreal, Canada.

Monceaux L. and Isabelle Robba I. (2002), "*Les analyseurs syntaxiques : atouts pour une analyse des questions dans un système de question-réponse ?* ", Actes de *TALN'2003*, pp.195-204.

Pedersen T. (2000), *A Simple Approach to Building Ensembles of Naive Bayesian Classifiers for Word Sense Disambigusation* In Proc. of the NAACL, pp. 63-69, 2000.

Sérasset G. and Boitet C. (2000). *On UNL as the future "html of the linguistic content" & the reuse of existing NLP components in UNL-related applications with the example of a UNL-French deconverter*, Proc. of COLING-2000, Saarbrücken, 31 July – 3 August 2000

Schwenk H. and Gauvain J.L. (2000), "*Combining multiple speech recognizers using voting and language model information*", published in IEEE International Conference on Speech and Language Processing (ICSLP), pp. II:915-918.

Vergne J. and Giguet E. (1998), *Regards théorique sur le « Tagging »,* Actes de TALN'1998, pp 24-33.