Dependency Parsing for French

Alexandre Guignochau
Lund Institute of Technology
Lund, Sweden.
aguignoc@etu.enseeiht.fr

Abstract

This document presents a simple dependency parser using Nivre's parsing applied to the French language, using a CoNLL-annotated corpus, which was never done before. The goal of the parser to predict the part-of-speech tags for each word, the dependency relations between two words, and the function of each word in the sentence. We managed to obtain quite good results for the first two parts, but the function tagging is not completely correct. This document also presents some applications for this dependency parser.

1 Credits

This document is written according to the ACL-2010 style guidelines.

2 Introduction

The CoNLL conference is an important meeting in the natural language processing world. Every year, research teams try to achieve great performance in predicting part-of-speech tags or functions for every word in a sentence, or dependency relations between two words. However, it is quite disturbing to see that a language widely spoken throughout the world, such as French, is not represented in these yearly conferences.
After having acquired a French corpus annotated according to the current CoNLL standards, our goal was to create such a dependency parser.
The parser is designed to predict part-of-speech tags, functions, and dependency relations after learning them from a manually, and thus correctly, annotated corpus. Our goal is to achieve an honest first performance. As we cannot compare our results to others, it is difficult to evaluate our score. Therefore, we are not aiming at a top-level performance, but at a solid basis for future comparisons.
This document will be organized according to this plan. The next section will deal with the goals of our project. The corpus used will be presented in Section 4, the parser in Section 5. The results will be discussed in Section 6, and further improvements will be suggested afterwards. Finally, section 8 will deal about the possible applications of our parser.

3 Goals

The first and main objective to this project is to create a dependency parser capable of achieving good results for the French language. However, because that was never realized before, it is not possible to compare our work with other teams. That is why we want to obtain a first result, even if it is possibly not perfect. Many improvements will be suggested at the end of this document, which can significantly increase the parser's performance.
Firstly, our parser should be able to predict quite well the part-of-speech tags. That is, it should be able to decide whether a word is a noun, a verb, or an adjective, for instance. This part is not as easy as it sounds for the French language, because some words written in exactly the same way can have different tags according to the sentence. For example, the verb "avoir" can be considered as a verb or as an auxiliary verb used to form a tense of another verb. This is one example, but many more can be found.
Secondly, our parser should be able to predict the dependency relations between words in the same sentence. For example, it should be able to find the noun to which an adjective is referring, or the subject of a verb.
Finally, our parser should be able to tag a function on a word. In other words, it should determine if a word is the subject of the sentence, or if it is an object of the verb, for instance.

4 Corpus

The corpus used throughout our project is a compilation of articles taken from the French newspaper "Le Monde".
It was kindly given to us by its author Anne Abeillé.
"Le Monde" is a daily newspaper dealing mostly with politics and economy. Therefore, the vocabulary used in the corpus is quite formal and very different from the vocabulary any Frenchmen would use in an everyday conversation, and the writing style is completely opposed to the one of a great nineteenth century author.
This corpus consists of about 300 000 words, and is annotated with dependency relations according to the current CoNLL standards.

5 Parser

The parser used is directly derived from a common Nivre parser. It is divided in two steps: first we have to train the parser using an important part of the corpus, and then we have to use it on another part of the corpus, quite smaller.
For each Nivre parser, we have to choose which features we want to include, since our result will depend on them. It is therefore essential to experiment different sets of features, as there is no miracle solution.
We decided to use the classic features, which can give pretty good results, and are very basic. Our features are the two first elements in the queue, and the top two elements in the stack.

5.1 Training

The first part of the parsing is to read the annotated corpus and save all the actions done. We will then create a decision model using the J48 tree-model in Weka. This model will tell us which action to do and which function should be used when parsing the unknown part of the corpus.
The reference parser will extract the actions with the format "action.function", for example "ra.suj" will represent the right arc action, with the subject function.
Once all the actions and functions are saved and the decision model created, we can go on to the next phase: the parsing.

5.2 Parsing

Now, we want to predict the part-of-speech and function tags, and the dependency relations of an unknown text.
The parser will read the corpus word by word, and for each word it will use the features decided before to realize the most probable action and function according to the decision model previously created.

6 Results

Our parser returns a text file composed of the original text, and the annotations that were made during the parsing sequence, according to the CoNLL format.
In order to produce an evaluation of the correctness of the output produced, we will use the CoNLL evaluation script that is comparing our output with the original version of the text, manually annotated, and thus correct.
This evaluation program will give us three results : the "Labeled attachment score", the "Unlabeled attachment score", and the "Label accuracy score".
The "unlabeled attachment score" represents the percentage of words that received a correct part-of-speech tag. The "label accuracy score" corresponds to the percentage of the words that received the correct function tag. Finally, the "labeled attachment score" is the percentage of words that received both a correct part-of-speech and function tag.
Here are the results obtained with our corpus:

- Unlabeled attachment score : 80.83 %
- Label accuracy score : 54.90 %
- Labeled attachment score : 49.25 %

The unlabeled attachment score is quite good if we compare it to the state-of-art in other languages quite close to French, such as Spanish, Italian, or even English.
However, the label accuracy score, and thus the label attachment score is surprisingly low. Some functions were completely incorrectly tagged, and we can find very curious anomalies in our result output. For instance, some sentences consist of two roots, which is completely absurd, and some words were tagged as punctuation by the function tagger, even if they are not tagged as such by the part-of-speech tagger.

7 Improvements

As the results are quite satisfying but far from optimal, here are some improvements that could be made to our parser to make it more efficient.
Firstly, it could be quite interesting to use two classifiers instead of one. At this time, we only use one classifier which will select both the ac-

tion and the function to use. The idea is to use a first classifier which will only select the action, and then use a second classifier which will choose the correct function to tag, knowing which action has been done. By using this way, the dependency relations would be more accurate than actually, as the correct action would be selected most of the time, and the function tagging could also slightly improve.

A second way of improving our parser would be to select better features. As explained before, the features selected are very basic, in order to have a first solid result, and establish a basis for comparison. However, many other features could be selected to improve this result. Although selecting more words from the stack or the queue does not really improve the result, it can be interesting to run tests with new features such as the first word of a sentence for example. Improving the selection of these features, can make both unlabeled and labeled attachment scores rise.

Thirdly, it could be a good idea to change the selection method. Currently, the action and function chosen are the most probable ones. However, this choice will affect the future choices made for the next words of the sentence, and it can possibly lead to wrong tags for those words. A solution would be to select the second most probable action and function for the current word, which can lead to much more probable tags for the next words of the sentence. To generalize this idea, instead of selecting the most probable action and function for a word, we should rather select the most probable action and function for this word and the following ones. However, in order to use this method, it is not possible to use the tree-based decision model of Weka. By using this selection method, both attachment scores will increase, but there should not be any sentence with a lot of illogic and wrong tags, as it can currently happen.

Finally, we can create specific rules for the French language which could complete or replace the decision model for some words. For instance, we know that the preposition "dans" ("in" in English) is nearly always followed by a place indicator. Therefore, we can know that after this preposition it is very likely to encounter a noun group. Many similar grammatical rules can be created and applied, but should be used with precaution as there are surely some exceptions.

These improvements can surely significantly increase the performance of our parser.

8 Applications

Let us now see the different applications possible for our parser.

First of all, it can be used by the common applications to all parsers. For example, it can be used in speech synthesis, in order to understand the meaning of a spoken sentence, and possibly to construct a grammatically correct and meaningful answer. It can also be used for grammar checking in word processing programs, even if its performances should be improved in order to do that.

Another useful application for our parser is a tool for non-native speakers interested in learning French. A similar tool already exists: it consists of a grammar checking program which will check the non-native speakers' texts and classify them into one of six levels. However, this tool is not very accurate, as it is based on a small corpus of texts written by French students. There are sometimes mistakes in those training texts, and there are only three different types of texts. Therefore, it will not be possible to check the correctness of a random text.

Our parser is based on a relatively large corpus, and is therefore capable of checking random texts. However, it would still not be perfect, as the corpus used is a compilation of very well-written newspaper articles. The construction of the sentences and the vocabulary used in this corpus is very different from the basic construction and vocabulary a beginner will have, and our parser will probably not be able to tag correctly the wrongly-spelled words.

9 Conclusion

We managed to obtain a parser for the French language according to the CoNLL standard, which was never done before. This parser is able to predict part-of-speech and function tags for every word in a sentence, as well as dependency relations between two words. The results, which can be used as a baseline for future comparisons, are quite encouraging, though they are low for the function tagging rate. However, many improvements can be done to have better results, and this can be a very interesting subject for future research.

References

Anne Abeillé, Lionel Clément and François Toussenel. 2003. *Building a treebank for French*. Kluwer Academic Publishers, Dordrecht, NL.

Marie Candito, Benoît Crabbé, Pascal Denis and François Guérin. 2009. *Analyse syntaxique du français : des constituants aux dépendances.* Senlis.

Joakim Nivre. 2003. *An Efficient Algorithm for Projective Dependency Parsing.*