Algorithms for Speech and NLP TP2: Robust Probabilistic Parser

Pierre-Louis Guhur

This report relates the development of a robust probabilistic constituency parser for French based on CYK algorithm and the PCFG model. Dataset was extracted from the SEQUOIA corpus.

I. Parsing sentences with PCFG model and CYK algorithm

In formal language theory, a grammar is a set of production rules that describe how to form strings from the language's alphabet according to the language's syntax. In the Chomsky hierarchy, context-free grammar (CFG) relates to a certain type of grammar, in which the left-hand side of each production rule consists of only a single nonterminal symbol. A probabilistic context-free grammar (PCFG) assigns to each production rule a probability, such that the probability of a derivation is the product of the probabilities of the productions used in that derivation.

The first task was to parse the SEQUOIA corpus. For example, for the sentence:

```
( (SENT (PP–MOD (P En) (NP (NC 1996)))
(PONCT ,) (NP–SUJ (DET la) (NC municipalite))
(VN (V etudie)) (NP–OBJ (DET la)(NC
possibilite) (PP (P d’) (NP (DET une)(NC
construction) (AP (ADJ neuve)))))
(PONCT .)))
```

The functional labels are first ignored, such that the sentence becomes:

```
( (SENT (PP (P En) (NP (NC 1996))) (PONCT
,) (NP (DET la) (NC municipalite)) (VN (
V etudie)) (NP (DET la)(NC possibilit )
(PP (P d’) (NP (DET une) (NC construction)
(AP (ADJ neuve))))) (PONCT .)))
```

Then, the sentence is recursively transformed as a tree structure. I used here the function *ntlk.Tree.fromstring*.

The SEQUOIA corpus was divided into three parts to constitute the training set, the validation set and the testing set. In order to parse sentences in the testing set, we need first to train a parser on the training set.

The first task consists in building a grammar, transform them in the normal Chomsky form and finally estimate the associated probabilities to each rule. The estimation is done statistically by taking the frequency of each rule. We assume thus that the training set is large enough to provide significant statistics.

After that, we apply the probabilistic version of CYK algorithm (Cocke-Younger-Kasami) on sentences in the testing set.

II. Dealing with out-of-vocabulary (OOV) words

The corpus contains several challenges: some words are not frequent enough to appear both on the training set and the testing set, whereas other words have spelling errors (such as ”barisienne” instead of ”parisienne”). This disproves the previous hypothesis to compute statictics of a PCFG.

To deal with OOV words, one needs to replace them by terminal symbols learned during the training phase. Then, the leaves are replaced to their original content. I used the ”Polyglot” library, which provides an embedding for a large corpus of French words. Unfortunately, more than 40% of words in the testing set were not present in this corpus. That is why, I extended the search of OOV words by several manners:

- words are tested with different upper and lower cases
- numbers are transformed with a unique Unicode symbol
- each combination of 1-Levenshtein distance is tested

If the word is finally found on the Polyglot's corpus, I searched its K nearest neighbors with the L2-distance. I manually set $K = 5$, as it produces reasonable words; however, K could also be chosen as the one providing the best overall result. Because most of these neighbors are not in the training set, I had to generate of all possibilities in the training set of these neighbors.

For example, if the testing set contains the word ”markting”, I first generate all words distant from ”markting” by one in the Levenshtein distance (such that ”merkting”, ”marktin”, ”marketing”...), then I keep only those which are present in Polyglot's corpus (here it is only ”marketing”). The KNN of ”marketing” are: ”marketing”, ”management”, ”média”, ”logistique”, ”design”. Then, I search in the training set whether these words, or some similar words close by a Levenshtein distance of 1, exist or not. This creates a pool of candidate word.

To select the best word among the pool of candidate, I used the context of the previous and next words. To do that, I built a transition matrix formed by bi-grams on the training set. Because most of the bi-grams do not exist, I use an interpolation defined as:

$$P_li(w_k|w_{k1}) = \lambda P_{uni}(w_k) + (1\lambda).P_{mle}(w_k|w_{k1}),$$

where P_{uni} is the uni-gram probability and $P_{mle}(w_k|w_{k1}) = \frac{(w_k|w_{k1})}{w_{k1}}$. The best candidate is the one that maximizes:

$$P(w_k|w_{k-1})P(w_{k+1}|w_k)$$

However, it happens that the word is completely unknown by Polyglot or that Polyglot's suggestions are too far from the learned word. In this case, I tried to find the word w that maximizes

$$P(w|w_{k-1})P(w_{k+1}|w)$$

. I thought that the context should be enough to provide a correct estimation of the grammatical class of a word. This was not working, since this method produces mainly punctuation symbols. Subsequently, I created a pool of candidates that share a same property with the OOV word: the number of letters. This works surprisingly well, because most of the OOV words are long and using used in specific context.

III. Evaluation

Qualitative evaluation is difficult to obtain for several reasons:

-

Evaluation was executed over 309 sentences with their ground truth. Because the grammar is large, with around 17000 rules, and the complexity to parse one sentence is $O(|G|.n^3)$ where G, the length of the grammar, the time elapsed to evaluate all those sentence is about 10 hours. Results are promising, 92 which give a rate of accuracy of 29.7%.

False predictions are sometimes due to errors in the rules productions predictions but to structure retrieval. For example, for the sentence "Wikipedia : ebauche droit.",I predicted: "((SENT (NP Wikipedia) (PONCT :)) (NP (NC ebauche) (AP droit))) (PONCT .)))", instead of: "((SENT (NP (NPP Wikipedia)) (PONCT :) (NP (NC ebauche) (NC droit)) (PONCT .)))"

Other type of errors happen during the structure retrieval, such that: "(PP *(P (D des))* (NP (NC HLM) (PP (P de) (NPP Paris)))))", instead of "(PP (P+D des) (NP (NC HLM) (PP (P de) (NP (NPP Paris)))))"

OOV correction is working surprisingly well, but in some cases, it induces absurd errors, for example when a punctuation symbol replace a word in the sentence. Additional coverage of such mistakes could improve performances. For example, each OOV word beginning with an upper case could be considered apart.

IV. Further work

This work was particularly interesting to execute for the OOV words, as it allows to find new ideas. It also opens new questions. For example, I only corrected OOV words during the testing phase, assuming that errors appearing in the training set would not matter. Indeed, if the training set contains "markting" instead of "marketing", all words "marketing" in the testing set will be transformed as "markting". However, this assumption is not always followed. For example, the word "merketing" would not be corrected as it has a Levenshtein distance of 2 to "markting", but it would have be corrected with a better training set; similarly, numbers should all be encoded as special characters during the training and the testing.

I should also consider words where a space is missing. For example, "thatmovie" could be split into two words, by testing each splitting possibility, until Polyglot testifies the existence of the two words.