Fatma Moalla
fatma.moalla@student.ecp.fr

Probabilistic Context-Free Grammar parser for French: assignment 2

1 PCFG model description

The objective of this assignment is to construct a Probabilistic French parser using the Probabilistic Context-Free Grammar (PCFG) model and the CYK algorithm that should be robut to Out-Of-Vocabulary words (OOV).

First, we extract the corpus from French SEQUOIA treebank v6.0 and we split it into 80% train, 10% validation and 10% tests sets. The goal is to learn the Part-of-Speech tags from the training sets and to parse the sentences in the testing sets.

In order to do this task, we will use the word embeddings from French polyglot given as an input and we will use the PCFG model and the CYK algorithm to perform the parsing in addition to an OOV model used to replace the unknown words.

1.1 PCFG model

In this part, we extracted the lexicon and the PCFG model. This model can be defined as one that for each leaf of the parsed tree (leaf is a Part-Of-Speech tag or binary Part-Of-Speech tag) assigns an occurrence probability. The probability of a given sentence of length n given Part-of-Speech tags (PoS) is defined as following:

$$P(S, PoS) = \prod_{i=1}^{n} P(R_i, L_i)$$

with R_i and L_i are defined as the right-hand side rules and left-hand side rules : $R_i \rightarrow L_i$

In addition,

$$P(R_i|L_i) = \frac{\#((R_i \rightarrow L_i))}{\#(R_i)}$$

In order to extract the sentence from the PoS rules and calculate the previous probabilities, we transform the corpus to Chomsky Normal Form using the NLTK package.

1.2 CYK algorithm

Once we extracted the probabilities and the combination between rules from the PCFG model, we start the CYK algorithm for parsing.

This algorithm will produce the most probable tree recursively for an input sentence by using dynamic programming. In fact, the idea behind CYK is to keep track of all the unary and binary rules related to the most probable Part-of-Speech tags and more specifically the PoS of the leaves of the parsed tree obtained at the beginning of the parsing process. At the end of this algorithm, we reverse the normal chomsky form of the predicted tree which is our output.

1.3 OOV module

The role of this module is to handle missing words from the vocabulary, replace them with the closest word using the Levenstein distance and the cosine similarity distance with embeddings in

order to assign a unique POS tag to these words. These embeddings are extracted from the Polyglot French lexicon To do this task, we actually combine both distances that choose max_{lev} neighbors using the Levenstein distance and max_{embed} to get the most similar words using the cosine distance from embedding words.

After this step, we can get several candidates that can replace the 'oov' word. In order to select the best word and take advantage of the context in the sentence, we use a linear combination of unigram and bigrams models as following (we use the logarithm in our model to avoid numerical overflow) :

$$P(w_i|w_{i-1}) = (1-\alpha)\log(P_{bi}(w_i|w_{i-1})) + \alpha\log(P_{uni}(w_i))$$

with w_i the new word that can replace the 'oov' word.

2 Results analysis and discussion

2.1 Results

The algorithm that we computed can parse a unique sentence in a .txt file and the whole test corpus which containing 310 sentences.

First we started with a sentence that we designed and we looked for its parsed tree :

- **Original sentence:** – `ce fichier est un test`
- **Parsed sentence** : `(SENT (PONCT –)(NP (DET ce)(NC fichiers))(VN (V est))(NP (DET un)(NC tests)))`

We can see that the algorithm replaced `test` by `tests` using the Out-Of-Vocabulary module and the Part of Speech tags are qualitatively accurate.

2.2 Error analysis

Due to time limitations, unfortunatly, we were not able to parse the whole testset and we only parsed **25 sentences** even though we used multiprocessing with 4 CPUs. We also, would have computed the accuracy and F1-score between the parsed sentence and the original sentence from the test corpus and interpret the results.

In this part, we want to understand the errors that our algorithm encounters.

For the following example in the testset, our parser was unable to find the tags for each word and also unable to find a replacement for the `oov` words like `civiques` and `civils`:

- `Contre lui,le parquet a requis quatre ans d' emprisonnement avec sursis, 50\_000 euros d'amende et trois ans d'interdiction des droits civiques et civils .`
- `(SENT (NULL Contre)(NULL lui)(NULL ,)(NULL le)(NULL parquet)(NULL a)(NULL requis)(NULL quatre)(NULL ans)(NULL d')(NULL emprisonnement)(NULL avec)(NULL sursis)(NULL ,)(NULL 50_000)(NULL euros)(NULL d')(NULL amende)(NULL et)(NULL trois)(NULL ans)(NULL d')(NULL intention)(NULL des)(NULL droits)(NULL et)(NULL .))`

The parsing error can maybe due to the fact that the sentence contains `numbers` and the fact that it is difficult to find a replacement of these numerical words in the provided vocabulary. Therefore, our parser needs to be more robust to this type words, for example by augmenting and enriching the initial corpus.