# NLP - TP 2

Marouen CHAFROUDA
marouen.chafrouda@student.ecp.fr

March 8, 2020

The goal of this homework is to implement a probabilistic parser for French that is based on the CYK algorithm and a Probabilistic context-free grammar model that is robust to OOVs.

## 1 PCFG:

The PCFG Model is composed of 2 elements:

- a grammar : stores the probability of each rule $X_0 \rightarrow X_1....X_n$ encountered:

$$P(X_0 \rightarrow X_1....X_n) = \frac{C(X_0 \rightarrow X_1....X_n)}{\sum_X C(X_0 \rightarrow X)}$$

- a lexicon : stores for each part-of-speech tag $X$, the frequencies of the words annotated with such tag:

$$P(X \rightarrow word) = \frac{C(X \rightarrow word)}{\sum_{words} C(X \rightarrow word)}$$

An auxiliary package called **nltk** was used as it provides a convenient framework to manipulate Trees.

## 2 Handling OOV words

As it is very likely that we will encounter unknown words or typos and our parser needs to be robust for OOVs, the strategy I used for handling such words :

1. If the word is in the vocabulary we simply return the word.

2. **Normalization** We check if simple changes in the word correspond to a word in vocabulary : numbers replaced by "#", checking Cases combination

3. **Embedding similarity** We check if the word or a Normalisation of the word is in the polyglot vocabulary. If it's the case, We use the embeddings to find the closest word in the vocabulary which also has a polyglot embedding and The POS tags for this word is used. The Closest word is found using Euclidian distance ($L_2$ norm ).

4. **Spelling correction** we try to correct the spelling of the word by computing the Damereau Levenstein Distance with each word of the lexicon. Different approaches were tried here as for the maximal distance to look for we tried a distance $d \leq 2$ and $d \leq 3$, and if we have several words achieving the minimal distance we tried using a bigram Matrix to keep the word that is closer in context of the words before and after. So finally among the possible condidates at Levenstein distance $\leq 2$ we use $score = distance(W[i]) + \lambda p(W[i]|W[i-1]) * p(W[i+1]|W[i])$ . with $\lambda = 1000$

5. If none of the above was works, the word couldn't not be recognized and we use two tags to solve with this:

   - NPP : token for proper noun and for which we'll assign the most frequent POS taken by a proper noun( 'NPP' in our Case ), this is assigned if the first letter of the unknown word is a Capital letter. **I ended up not using this as on the validation set it yielded bad results with a lot of words being assigned NPP when they're not actual proper nouns**

   - UNK the word cannot be recognized and we assigned the possible POS seen in train corpus along with their corresponding frequencies

## 3 Parsing with CYK algorithm

The CYK algorithm was implemented as described in **Daniel Jurafsky** and **James H. Martin** 's Speech and Language Processing book.

the algorithm seems to work fine as when testing either on validation set or on train set we get **93%** of the POS right, but it is extremely slow (1.5 hours to parse 310 sentences on test ). The complexity of the CYK algorithm is $\mathcal{O}(n^3 \cdot L)$, where $n$ is the length of the words in question and $L$ is the size of the grammar. To reduce this I tried reducing the number of rules in grammar for example by getting rid of the ones seen only once as these can be deemed coming from error in the train corpus or just insignificant/extremely rare to encounter.

Thanks to the CYK algorithm, we end with the most likely parsing tree in CNF form which we need to unnormalize (this was done using nltk).

## 4 Results and error Analysis

| Grammar | Leven-stein | Not Parsed | POS acc |
|---|---|---|---|
| Full | 3 | 96 | 0.8835 |
| Full | 2 | 96 | 0.88349 |
| Filtered | 2 | 132 | 0.8791 |
| Full+ Bigram | 3 | 104 | 0.8344 |
| Full+ Bigram | 2 | 71 | 0.8568 |

Table 1: Accuracy On the test set

The parsing system was built using the first 80% of SEQUOIA as training corpus. the next 10% were used to test different OOV strategies and the last 10% (the last 310 last sentences.) were used to test the parser.

Out of the 310 test sentences, 239 were succesfully parsed. Tagging accuracy was 85.68% . The parsing errors can be seperated to two categories:

- Grammar related : These errors are due to rules in test sentences that we did not see in the training corpus, unfortunately, these errors cannot be fixed without providing a bigger training corpus. An example of this is : the sentence : *"Le Procès en première instance"* where we don't Have in Our Grammar a Rule where the Sentence starts with a DET ('Le') followed by an NC ('Procès').

- OOV related : The other errors are related to our OOV strategy. When analysing the sentences we were unable to parse, we can see that these sentences are typically longer than the mean sentence length in test set (20 words against 26 words in average ). So we can see that the longer sentences the more they are affected by errors in OOVs which is expected. examples of this are : *"- Trois entreprises sont retenues : la Somatem , filiale de la Lyonnaise des eaux , la CG2A , filiale de l' ex- Compagnie générale des eaux -LRB- CGE , devenue Vivendi -RRB- , et le groupe américain Otis ."* , where we have 8 OOVs but also OOVs like **"Somatem"** are tricky to handle since these are Proper Nouns, and will be assigned a **<UNK>** Token (before it used to be assigned **<NPP>** token but this yielded worse results overall ). this sentence has many Proper Nouns and could not be parsed.

Finally, the OOV module does not make weird choices, and the correct POS tags of most of the OOVs are in the Tags of words assigned by our OOV strategy ( 96% )

## 5 Paths for improvements

Below are the three paths I would have followed to improve the parser with more time :

1) First of all, The parser as a whole is extremely slow (1.5 hours to parse 310 test sentences) This Was a big issue as it provides less time to try and test how good our model work and to observe the impact of parameters, so this can definitely be sped up especially that the parsing task is asynchronous so we can run the computations in parallel.

2) The OOV strategy can also be more extensive with the introduction of more advanced ways of spelling correction especially with the usage of Transformer Netsworks now, **BERT** can be used for example as a language model to help guessing the right correction of the OOV word.

## References

[1] Tutorial on Polyglot Embeddings : link

[2] Daniel Jurafsky & James H. Martin, Speech and Language Processing, 2018