Houssam Zenati
CentraleSupélec

# NLP Homework - PCFG - CYK

NLP
March 18, 2019

In this practical work we develop a basic probabilistic parser for French that is based on the CYK algorithm and the PCFG model and that is robust to unknown words.

## 1 Implementation of a PCFG Parser

### 1.1 Extracting a PCFG from the Training Corpus

A PCFG [1, 2] assigns a probability to each parse tree $T$ of a sentence $S$ which is defined as the product of the probabilities of all the $n$ rules used to expand each of the $n$ non-terminal nodes in the parse tree $T$, where each rule $i$ can be expressed as $LHS_i \rightarrow RHS_i$:

$$P(T, S) = \prod_{i=1}^{n} P(RHS_i | LHS_i) \tag{1}$$

In the code, to exctract a PCFG from the training corpus, we count rules $\alpha \rightarrow \beta$ and count non-terminals $\alpha$ in two dictionaries so as to compute normalized probabilities:

$$P(\alpha \rightarrow \beta | \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)} \tag{2}$$

Those probabilities are stored in dictionaries of unary and binary rule probabilities (see (1.2)).

### 1.2 The CYK Algorithm

Once the PCFG is extracted, to produce the the most-likely parse $\hat{T}$ for a given sentence $S$, the CYK parsing algorithm for PCFGs solves [1]:

$$\hat{T} = \underset{T \text{ s.t } T \text{ yields } T}{\arg\max} P(T) \tag{3}$$

Now following the CYK presented in [2] we assume a Chomsky Normal Form for every tree (and thus preprocess every tree accordingly when learning rules, see code) which justifies learning unary and binary rules only. Note that for sparsity reasons, we also remove functional labels in preprocessing. The CYK is implemented following the pseudo-code in [2]. The input are tokenised sentences and outputs are in the same bracketed format as the training data.

### 1.3 Out-of-Vocabulary Module

An OOV module is implemented to assign a unique part-of-speech to which is not included in the lexicon extracted from the training corpus. The module retrieves a ”similar” word from the extracted lexicon using a combination of formal and embedding similarity.

For the formal similarity, we generate all words within edit Levenshtein distance ($k = 1$ for faster inference) and then intersect with the extracted lexicon. For the embedding similarity, the cosine similarity is used to retrieve $n$ best embedding candidates ($n$ is finetuned on the validation set). Moreover, the Polyglot embedding lexicon for French [3] is used to retrieve the embeddings of the extracted lexicon words only.

Concretely, to combine those similarities, since embedding similarities are sorted (while not for Levenshtein ones), we choose to pick embedding candidates in their order and select the first which is also a Levenshtein candidate. If there is no overlap, we choose to take a random candidate from Levenshtein candidates and otherwise any embedding candidate.

## 2 Evaluation of the parser

### 2.1 Evaluation of the Performances

We use the SEQUOIA treebank v6.0 and split it into 3 parts (80% / 10% / 10%) respectively for training, development and evaluation of the parser. We then report in Table (1) the part-of-speech (POS) mean accuracy,mean precision, F1 and recall using pyevalb.

| | Precision | Recall | F1 Score | Accuracy |
|---|---|---|---|---|
| Test set | 78.39% | 61.29% | 67.85% | 61.29% |

Table 1: Results on the test set (averaged)

### 2.2 Example of Parsings, Error Analysis

We evaluate the parser on a small set of sentences to qualitatively evaluate disambiguation.

- 'Les enfants et les voitures' becomes 'Les enfants et les voiture', the parser does not know the plural and replaces 'voitures' to singular
- 'Cette insuffisanse rénale' becomes 'Cette insuffisante rénale'. Unfortunately 'insuffisante' does represent an excellent candidate for both Levenshtein and embedding similarities however it does not grammatically makes sense here in the context.
- 'Il y a une fautte' becomes 'Il y a une faute'. good
- 'L'Algérie est un beau pays' becomes 'sinistrés est un eau pays'. It looks like both 'Algérie' and 'beau' are unknown words. First proposed candidate does not make sense

Note that all retrieved trees were grammaticaly meaningful (except for 'insuffisanse' (AJD)).

To improve the parser, we could include singular/plural and other grammatical rules to augment the lexicon. We could also consider contextual features to learn most probable fit. Also, more easily, we could think of enriching the treebank dataset to have more lexicon and more robust parsing. More generally, PCFG solutions exist for missing structural dependencies between rules, lack of sensitivity to lexical sependencies, etc. Such more elaborated PCFGs were out of the scope of this introduction work.

## References

[1] D. Jurafsky and J. H. Martin, *Speech and Language Processing.* Stanford University, 2018.

[2] M. Collins, *Probabilistic Context-Free Grammars (PCFGs).* Columbia University, 2011.

[3] R. Al-Rfou, B. Perozzi, and S. Skiena, "Polyglot: Distributed word representations for multilingual nlp," in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, (Sofia, Bulgaria), pp. 183–192, Association for Computational Linguistics, August 2013.