Algorithms for speech and language processing
TP2 Report: *Building a probabilistic parser for French*

Wilson Jallet

March 4, 2020

1 Implementation

Building the PCFG and lexicon

We use the Python Natural Language ToolKit (NLTK) [1] package to parse the treebank data as constituency trees. These trees are then shuffled together before splitting between training, development and testing data. This is needed because lines in the treebank seem to be grouped together thematically[1]. For reproducibility, we fixed the value of the random seed.

We then strip the lexical rules, leaving the part-of-speech (PoS) as terminals, and build the PCFG from these productions using NLTK's PCFG data structure and a helper which computes the empirical probabilities of individual productions.

The lexical rules are transformed into a lexicon using our own ProbabilisticLexicon Python class, which uses the NLTK probabilistic production data structure to represent the $(\text{token}, \text{PoS}, \text{probability})$ triple.

Out-of-Vocabulary module

There are two complementary strategies to propose surrogates for out-of-vocabulary (OOV) words: computing **spelling** nearest neighbors in the corpus according to the Levenshtein Edit distance, and computing **semantic** nearest neighbors according to the cosine distance of some embeddings (and intersecting with the corpus vocabulary).

For the Levenshtein-nearest neighbors, we run through the corpus and compute all the distances, and get the k elements with the lowest distance (without sorting). For the embedding nearest neighbors, we use Scikit-Learn's nearest neighbors implementation [2] (which is very efficient), which we fit using the cosine distance to measure semantic similarity[2].

In order to score the combined list of proposals, we use a language model trained on the corpus. We use NLTK's language modeling API to extract unigrams and bigrams from the corpus and assign appropriate weighted scores (averaging between bigram and unigram scores).

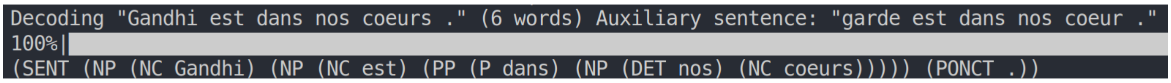
2 Results

Handling OOV proper nouns is a difficult problem, especially when they are the first token in a sentence and thus have no context for the OOV module to use: it happens that common

[1] For instance l. 800–1000 discuss health, and l. 2700–2800 seem to discuss French politics.

[2] We actually use the Euclidean distance on normalized embedding vectors, which is equivalent to the cosine distance because $\|\frac{x}{\|x\|} - \frac{y}{\|y\|}\|^2 = 2 - 2\langle \frac{x}{\|x\|}, \frac{y}{\|y\|} \rangle$.

nouns or even verbs are used as replacements when they are picked up by edit distance, see fig. 1.



```
Decoding "Gandhi est dans nos coeurs ." (6 words) Auxiliary sentence: "garde est dans nos coeur ."
100%|
(SENT (NP (NC Gandhi) (NP (NC est) (PP (P dans) (NP (DET nos) (NC coeurs))))) (PONCT .))
```

Figure 1: Failure of parsing a proper noun which is out-of-vocabulary.

References

[1] Edward Loper and Steven Bird. “NLTK: The Natural Language Toolkit”. In: *CoRR* cs.CL/0205028 (2002). URL: `http://dblp.uni-trier.de/db/journals/corr/corr0205.html#cs-CL-0205028`.

[2] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.