MVA - Algorithms for Speech and NLP TD 2

Yayun Dai

2019/03/13

1 Implementation

1.1 Probabilistic Context-free Grammar

To read trees from the training corpus, I make use of *nltk.Tree* class from NLTK library. The trees are binarized to Chomsky Normal Form before entering the PCFG module. The extracted rules contain two parts :

— a dictionary named 'grammar', consisting of items of the form *tuple(lhs, rhs)* : *probability*. The keys are tuples containing only grammar symbols and the values are the corresponding probabilities. *lhs* is a string that represents the grammar symbol of the left-hand side of a rule and *rhs* is a string that can be split by spaces into the grammar symbols of the right-hand side of the rule.

— a dictionary named 'lexicon', consisting of items of the form *tuple(lhs, rhs)* : *probability*. Here *lhs* represents a token and *rhs* is a corresponding part-of-speech tag. The sum of the probabilities of all items for a given token is 1.

1.2 Out-of-vocabulary words

For a token not included in the lexicon extracted from the training corpus, I replace it by a "similar" word in the extracted vocabulary. To make use of both formal similarity (to handle spelling errors) and embedding similarity (measured by cosine similarity), my procedure is :

— If the token exists in word embeddings, then I use the embedding similarity to find the most similar token in the extracted vocabulary.

— If the token does not appear in word embeddings, it is probable because the token is not correctly spelled, then I use Levenshtein Distance to find a corrected token.

1.3 CYK algorithm

For the implementation CYK algorithm, I make use of the pseudo code from the website of USNA https ://www.usna.edu/Users/cs/nchamber/courses/nlp/f12/labs/cky-pseudo.html :

2 Error Analysis

To evaluate my parser, I used the standard tool for constituency parse evaluation evalb on 10% of the treebank. The results are shown in Table 1. My Tagging accuracy is 93.48% and the complete match is 26.67%. Among the 310 sentences, there are 10 that my algorithm failed to parse.

	All	len<=40
Number of sentence	310	288
Number of Error sentence	0	0
Number of Skip sentence	10	9
Number of Valid sentence	300	279
Bracketing Recall	72.95	76.01
Bracketing Precision	71.47	74.31
Bracketing FMeasure	72.20	75.15
Complete Match	26.67	28.67
Average crossing	1.92	1.42
No crossing	52.33	55.91
2 or less crossing	74.00	78.49
Tagging accuracy	93.48	93.79

TABLE 1: Summary of evaluation results

By analyzing these sentences, I find that some of them are because the OOV assigned a token of completely different nature from the original token.
For example :
Nous nous efforçons depuis des années de mieux défendre les intérêts financiers de la Communauté .
My OOV replaced *efforçons* by *efforts*, which may make the parsing procedure difficult as a verb is recognized as a noun.
Other of them may due to the grammar complexity of the sentences themselves. They may contain some structure that is rarely seen in our training corpus.
For example :
Le deuxième élément est un surcroît de conception démocratique , ce qui signifie que si un ou plusieurs pays , quels qu' ils soient , pour une raison quelconque , souhaitent , du_moins dans un premier temps , ne_pas participer , cela ne peut être tenu pour une exclusion ni pour une limitation de leur présence au_sein_de l' Union .
This sentence is indeed very grammarly difficult to parse.

For the first problem, we can solve it by using a larger word embeddings. There are many words that are not contained in the polyglot library such as *efforcer*. For the second problem, we can improve it by training a PCFG on a larger corpus. With a larger dataset, PCFG may be able to learn more complicate grammar rules.