E1:246 Natural Language Understanding - Assignment 3
Parsing With PCFG's

Rahul Bansal
M.tech Ist year
rahulbansal@iisc.ac.in

Abstract

Parsing uncovers the hidden structure of linguistic input. In many applications involving natural language, the underlying predicate-argument structure of sentences can be useful. The syntactic analysis of language provides a means to explicitly discover the various predicate-argument dependencies that may exist in a sentence. A Probabilistic Context-Free Grammar (PCFG) is simply a Context-Free Grammar with probabilities assigned to the rules such that the sum of all probabilities for all rules expanding the same non-terminal is equal to one. PCFG parsing takes advantage of probabilities by giving you the most probable parse for a sentence.

Introduction

The information required to parse a query sentence is called grammar. Grammar is a set of syntactic rules. One might write down the rules of syntax as a context-free grammar (CFG). The following CFG represents a simple grammar of transitive verbs in English, verbs (V) that have a subject and object noun phrase (NP), plus modifiers of verb phrases (VP) in the form of prepositional phrases (PP).

```
S -> NP VP
NP -> 'John' | 'pockets' | D N | NP PP
VP -> V NP | VP PP
V -> 'bought'
D -> 'a'
N -> 'shirt'
PP -> P NP
P -> 'with'
```

Figure 1: Context-Free Grammar

The above CFG can produce a syntax analysis of a sentence like **(John bought a shirt with pockets)** with S as the start symbol of the grammar. Parsing the sentence with the CFG rules gives us two possible derivations for this sentence. In one parse, pockets are a kind of currency which can be used to buy a shirt, and the other parse, which is the more plausible one, John is purchasing a kind of shirt that has pockets.

```
(S (NP John)
   (VP (VP (V bought)
           (NP (D a)
               (N shirt)))
       (PP (P with)
           (NP pockets))))
```

```
(S (NP John)
   (VP (V bought)
       (NP (NP (D a)
               (N shirt))
           (PP (P with)
               (NP pockets)))))
```

Figure 2: Example of CFG

However, writing down a CFG for the syntactic analysis of natural language is problematic. Rules interact with each other in combinatorially explosive ways. So not only do we need to know the syntactic rules for a particular language, but we also need to know which analysis is the most plausible for a given input sentence.

Probability Context Free Grammar

We want to provide a model that would match the intuition that the second tree in Figure 2 is preferred over the first. The parses themselves can be thought of as ambiguous (leftmost or rightmost) derivations of the following CFG.
We can add scores or probabilities to the rules in this CFG in order to provide a score or probability for each derivation. The probability of a derivation is simply the sum of scores or product of probabilities of all the CFG rules used in that derivation. In order to make sure that the probability of the set of trees generated by a PCFG is well-defined, we assign probabilities to the CFG rules such that for a rule $N \rightarrow \alpha$ the probability is $P\ ((N \rightarrow \alpha)\ /\ N)$ that is, each rule probability is conditioned on the left-hand side of the rule. This means that during the context-free expansion of a non-terminal, the probability is distributed among all the expansions of the non-terminals.

$$\sum_{\alpha} P(N \rightarrow \alpha) = 1$$

```
S -> NP VP (1.0)
NP -> 'John' (0.1) | 'pockets' (0.1) | D N (0.3) | NP PP (0.5)
VP -> V NP (0.9) | VP PP (0.1)
V -> 'bought' (1.0)
D -> 'a' (1.0)
N -> 'shirt' (1.0)
PP -> P NP (1.0)
P -> 'with' (1.0)
```

Figure 3: Probability Context-Free Grammar

Treebank

A treebank is simply a collection of sentences (also called a corpus of text), where each sentence is provided a complete syntax analysis. The syntactic analysis for each sentence has been judged by a human expert as the most plausible analysis for that sentence. Treebanks provide annotations of syntactic structure for a large sample of sentences. We can use supervised machine learning methods in order to train a parser to produce a syntactic analysis for input sentences by generalizing appropriately from the training data extracted from the treebank.

Treebanks solves the knowledge acquisition problem of finding the grammar underlying the syntax analysis since the syntactic analysis is directly given instead of a grammar. Since each sentence in a treebank has been given its most plausible syntactic analysis, supervised machine learning methods can be used to learn a scoring function over all possible syntax analyses.

NLTK treebank of 200 documents is used for this assignment.

Parsing Algorithm

Given an input sentence, a parser produces an output analysis of that sentence, which we now assume is the analysis that is consistent with a treebank that is used to train a parser.

Consider the following simple CFG G that can be used to derive strings such as a and b and c from the start symbol N.

An important concept for parsing is a derivation.

```
N -> N 'and' N
N -> N 'or' N
N -> 'a' | 'b' | 'c'
```

For the input string a and b or c the following sequence of actions separated by the arrow symbol represents a sequence of steps called a derivation: In this derivation each line is called a sentential

```
N
=> N 'or' N
=> N 'or c'
=> N 'and' N 'or c'
=> N 'and b or c'
=> 'a and b or c'
```

form. Furthermore, each line of the derivation applies a rule from the CFG in order to show that the input can, in fact, be derived from the start symbol N. In the above derivation, we restricted ourselves to only expand on the rightmost non-terminal in each sentential form. This method is called the rightmost derivation of the input using a CFG. An interesting property of a rightmost derivation is revealed if we arrange the derivation in reverse order:

```
'a and b or c'
=> N 'and b or c'       # use rule N -> a
=> N 'and' N 'or c'     # use rule N -> b
=> N 'or c'             # use rule N -> N and N
=> N 'or' N             # use rule N -> c
=> N                    # use rule N -> N or N
```

Model for Parsing

In order to find the most plausible parse tree, the parser has to choose between the possible derivations each of which can be represented as a sequence of decisions. Let each derivation $D = d_1, d_2.....d_n$ which is the sequence of decisions used to build the parse tree. Then for input sentence x the output parse tree y is defined by the sequence of steps in the derivation. We can introduce a probability for each derivation:

$$P(x, y) = P(d_1, d_2...d_n)$$
$$= \prod_{i=1}^{n} P(d_i|(d_1...d_{i-1}))$$

The conditioning context in the probability $P(d_i|(d_1...d_{i-1}))$ is called the history and corresponds to a partially build parse tree (as defined by the derivation sequence). A simplifying assumption is being made that keeps the conditioning context to a finite set by grouping the histories into equivalence classes using a function g.

$$P(x, y) = P(d_1, d_2...d_n)$$
$$= \prod_{i=1}^{n} P(d_i|g(d_1...d_{i-1}))$$

Results

Below table shows the evaluation metrics: Precision, Recall and F1 score.

Precision	51.2
Recall	53.8
F1 Score	52.4

Sentence 1 - I worked very hard to finish the assignments given in Natural Language Understanding course .

```
(ROOT
  (S
    (NP (PRP I))
    (VP (VBD worked)
      (ADJP (RB very) (JJ hard))
      (S
        (VP (TO to)
          (VP (VB finish)
            (NP
              (NP (DT the) (NNS assignments))
              (VP (VBN given)
                (PP (IN in)
                  (NP (NNP Natural) (NNP Language) (NNP Understanding) (NN course)))))))))
    (. .)))
```

Figure 4: Parsed tree from online parser

```
(S
  (NP-SBJ (PRP I))
  (S|<VP-.>
    (VP
      (VBD worked)
      (VP|<RB-ADVP>
        (RB very)
        (VP|<ADVP-VP>
          (ADVP (RB hard))
          (VP
            (TO to)
            (VP
              (VBG finish)
              (NP
                (Language (DT the) (NN assignments))
                (PP
                  (VBN given)
                  (PP
                    (IN in)
                    (NP
                      (NNP Natural)
                      (NP|<NNUnderstandingNP>
                        (NNP UNK)
                        (NP|<NNP-NN> (NNP UNK) (NN course)))))))))))
    (. .)))
```

Figure 5: Parsed tree from implemented parser

Sentence 2 - Polling will be held on 23 April for the third and largest phase of Lok Sabha election in 116 seats.

```
(ROOT
  (S
    (NP (NN Polling))
    (VP (MD will)
      (VP (VB be)
        (VP (VBN held)
          (PP (IN on)
            (NP
              (NP (CD 23))
              (NP (NNP April))))
          (PP (IN for)
            (NP
              (NP (DT the) (JJ third)
                (CC and)
                (JJS largest) (NN phase))
              (PP (IN of)
                (NP (NNP Lok) (NNP Sabha) (NN election)))
              (PP (IN in)
                (NP (CD 116) (NNS seats))))))))
    (. .)))
```

Figure 6: Parsed tree from online parser

```
(S
  (NP-SBJ SabhaS Polling))
  (VP
    (MD will)
    (VP
      (VB be)
      (VP
        (VBN held)
        (VP|<PP-CLR-S-CLR>
          (PP-CLR (IN onseats.NP (CD 23)))
          (VP|<S-CLR-PP>
            (S-CLR
              (NP-SBJ
                (NP 116P April))
                (PP
                  (IN for)
                  (NP
                    (NP (DT the) (JJ third))
                    (NP|<CC-NP>
                      (CC and)
                      (NP (JJS largest))))))
              (VP
                (VB phase)
                (PP-CLR
                  (IN of)
                  (NP
                    (JJ Lok)
                    (NP|<NN-NN> (NN UNK) (NN election))))))
            (PP (IN in) (NP (NN UNK) (NN UNK)))))))))
```

Figure 7: Parsed tree from implemented parser

Sentence 3 - dhoni is the coolest captain .

```
(ROOT
  (S
    (NP (NN dhoni))
    (VP (VBZ is)
      (NP (DT the) (JJS coolest) (NN captain)))
    (. .)))
```

Figure 8: Parsed tree from online parser

```
(S
  (NP-SBJ (NNS dhoni))
  (S|<VP-.>
    (VP
      (VBZ is)
      (NP (DT the) (NP|<NN-NN> (NN coolest) (NcaptainNK))))
    (. .)))
```

Figure 9: Parsed tree from implemented parser

Sentence 4 - He is the most powerful person .

```
(ROOT
  (S
    (NP (PRP He))
    (VP (VBZ is)
      (NP (DT the)
        (ADJP (RBS most) (JJ powerful))
        (NN person)))
    (. .)))
```

Figure 10: Parsed tree from online parser

```
(S
  (NP-SBJ (PRP He))
  (S|<VP-.>
    (VP
      (VBZ is)
      (NP-PRD
        (DT the)
        (NP-PRD|<ADJP-NN>
          (ADJP (RBS most) (JJ powerful))
          (NN person))))
    (. .)))
```

Figure 11: Parsed tree from implemented parser