Einführung in die Computerlinguistik

Probabilistic CFG

Laura Kallmeyer
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2013

Overview

1. Data-Driven Parsing
2. PCFG
3. Inside and Outside Probability
4. Parsing

[Jurafsky and Martin, 2009, Manning and Schütze, 1999]

Some of the slides are due to Wolfgang Maier.

Data-Driven Parsing

- Linguistic grammars can not only be created manually. Another way to obtain grammars is to interpret the syntactic structures in a treebank as the derivations of a latent grammar and to use an appopriate algorithm for grammar extraction.
- One can also estimate occurrence probabilities for the rules of a grammar. These can be used to determine the best parse, resp. parses of a sentence.
- Furthermore, rule probabilities can serve to speed up parsing.
- Parsing with a probabilistic grammar obtained from a treebank is called data-driven parsing.

PCFG (1)

In most cases, probabilistic CFGs are used for data-driven parsing.

A Probabilistic Context-Free Grammar (PCFG) is a tuple $G_P = (N, T, P, S, p)$ where (N, T, P, S) is a CFG and $p : P \to [0, 1]$[a] is a function such that for all $A \in N$,

$$\sum_{A \to \alpha \in P} p(A \to \alpha) = 1$$

$p(A \to \alpha)$ is the conditional probability $p(A \to \alpha \mid A)$

[a] $[0, 1]$ denotes $\{i \in \mathbb{R} \mid 0 \leq i \leq 1\}$.

PCFG (2)

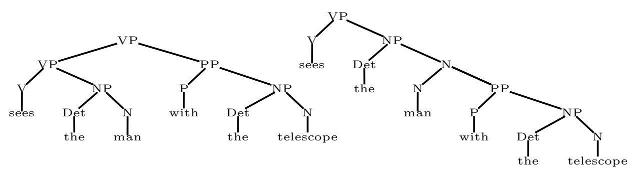
Example:

.8	VP → V NP	1	V → sees
.2	VP → VP PP	1	Det → the
1	NP → Det N	1	P → with
1	PP → P NP	.6	N → man
.1	N → N PP	.3	N → telescope

Start symbol VP.

PCFG (3)

- Probability of a parse tree: product of the probabilities of the rules used to generate the parse tree.
- Probability of a category A spanning a string w: sum of the probabilities of all parse trees with root label A and yield w.



$p = 0.6 \cdot 0.8 \cdot 0.2 \cdot 0.3 = 0.0288$ $\qquad$ $p = 0.6 \cdot 0.8 \cdot 0.1 \cdot 0.3 = 0.0144$

p(VP,sees the man with the telescope) $= 0.0288 + 0.0144$

PCFG (4)

Probabilities of leftmost derivations:

Let $G = (N, T, P, S, p)$ be a PCFG, and let $\alpha, \gamma \in (N \cup T)^*$.

- Let $A \to \beta \in P$. The probability of a leftmost derivation $\alpha \overset{A\to\beta}{\Rightarrow_l} \gamma$ is
$$p(\alpha \overset{A\to\beta}{\Rightarrow_l} \gamma) = p(A \to \beta)$$
- Let $A_1 \to \beta_1, \dots, A_m \to \beta_m \in P$, $m \in \mathbb{N}$. The probability of a leftmost derivation $\alpha \overset{A_1\to\beta_1}{\Rightarrow_l} \cdots \overset{A_m\to\beta_m}{\Rightarrow_l} \gamma$ is
$$p(\alpha \overset{A_1\to\beta_1}{\Rightarrow_l} \cdots \overset{A_m\to\beta_m}{\Rightarrow_l} \gamma) = \prod_{i=1}^{m} p(A_i \to \beta_i)$$

PCFG (5)

- The probability of leftmost deriving γ from α, $\alpha \overset{*}{\Rightarrow}_l \gamma$ is defined as the sum over the probabilities of all leftmost derivations of γ from α:
$$p(\alpha \overset{*}{\Rightarrow}_l \gamma) = \sum_{i=1}^{k} \prod_{j=1}^{m} p(A^i_j \to \beta^i_j)$$
where $k \in \mathbb{N}$ is the number of leftmost derivations of γ from α and $m \in \mathbb{N}$ is the derivation length of the ith derivation and $A^i_j \to \beta^i_j$ is the jth derivation step of the ith leftmost derivation.

In the following, the subscript l is omitted assuming that derivations are identified with the corresponding leftmost derivation for probabilities.

PCFG (6)

A PCFG is consistent if the sum of the probabilities of all sentences in the language equals 1.

Example of an inconsistent PCFG G:

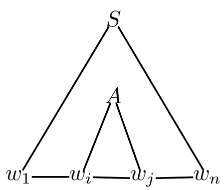
$.4\ S \to A \quad .6\ S \to B \quad 1\ A \to a \quad 1\ B \to B$

Problem: probability mass disappears into infinite derivations.

$\sum_{w \in L(G)} p(w) = p(a) = 0.4$

Inside and Outside Probability (1)

Idea: given a word $w = w_1 \cdots w_n$ and a category A, we consider the case that A is part of a derivation tree for w such that A spans $w_i \cdots w_j$.



- Inside probability of $\langle A, w_i \cdots w_j \rangle$: probability of a tree with root A and leaves $w_i \cdots w_j$.
- Outside probability of $\langle A, w_i \cdots w_j \rangle$: probability of a tree with root S and leaves $w_1 \cdots w_{i-1} A w_{j+1} \cdots w_n$.

Inside and Outside Probability (2)

Let G be a PCFG and let $w = w_1 \cdots w_n$, $n \in \mathbb{N}$, $w_i \in \Sigma$ for some alphabet Σ, $1 \le i \le n$, be an input string. Let $1 \le i \le j \le n$ and $A \in N$.

1. The probability of deriving $w_i \cdots w_j$ from A is called inside probability and defined as
$$p(A \stackrel{*}{\Rightarrow} w_i \cdots w_j)$$
2. The probability of a deriving A, preceded by $w_1 \cdots w_{i-1}$ and followed by $w_{j+1} \cdots w_n$ in a parse tree rooted with S is called outside probability and defined as
$$p(S \stackrel{*}{\Rightarrow} w_1 \cdots w_{i-1} A w_{j+1} \cdots w_n)$$

The product of inside and outside probability gives the probability of a parse tree for w containing a non-terminal A that spans $w_i \cdots w_j$.

Inside and Outside Probability (3)

Inside algorithm for computing the inside probabilities of a PCFG $G = (N, T, P, S, p)$ given an input string w:

- We assume all non-terminals $A \in N$ to be continuously numbered from 1 to $|N|$.
- We use a three-dimensional matrix chart β, where the first dimension contains an index denoting a non-terminal, and the second and third dimension contain indices denoting the start and the end of a part of the input string.
- Each cell $[A, i, j]$ in β, written as $\beta_A(i, j)$ contains the sum of probabilities of all derivations $A \stackrel{*}{\Rightarrow}_l w_i \cdots w_j$.
- We assume our grammar to be in Chomsky Normal Form. I.e., all productions have either the form $A \to a$ with $a \in T$ or $A \to BC$ with $B, C \in N$.

Inside and Outside Probability (4)

Computation of the inside probabilities (initialize all probabilities with 0):

1. For $1 \leq i \leq n$ and $p : A \rightarrow w_i$: $\beta_A(i,i) = \beta_A(i,i) + p$.
2. For all l with $2 \leq l \leq n$, all i with $1 \leq i \leq n-l+1$ and all $A \in N$: Let $j = i+l-1$. Then

$$\beta_A(i,j) = \sum_{p:A \rightarrow BC} \sum_{i \leq k < j} p \cdot \beta_B(i,k)\beta_C(k+1,j)$$

Inside and Outside Probability (5)

Outside algorithm for computing the outside probabilities of a PCFG: We use a three-dimensional matrix α with dimensions as in β (nonterminal index and start and end index of span). I.e., $\alpha_A(i,j)$ gives $p(S \stackrel{*}{\Rightarrow}_l w_1 \cdots w_{i-1} A w_{j+1} \cdots w_n)$.

1. Length n: $\alpha_S(1,n) = 1$ and $\alpha_A(1,n) = 0$ for all $A \neq S, A \in N$.
2. Length $l = n-1$ to $l = 1$:
   For all l with $n > l \geq 1$ and for all i with $1 \leq i \leq n-l+1$: $j = i+l-1$.

$$\begin{aligned}\alpha_A(i,j) \quad = \quad & \textstyle\sum_{p:B \rightarrow AC} \sum_{k=j+1}^{n} \beta_C(j+1,k) \cdot p \cdot \alpha_B(i,k) \\ & + \textstyle\sum_{p:B \rightarrow CA} \sum_{k=1}^{i-1} \beta_C(k,i-1) \cdot p \cdot \alpha_B(k,j)\end{aligned}$$

Inside and Outside Probability (6)

Probability of a sentence:

- $p(w_1 \cdots w_n) = \beta_S(1,n)$
- $p(w_1 \cdots w_n) = \sum_A \alpha_A(k,k) p(A \rightarrow w_k)$ for any k, $1 \leq k \leq n$
- $p(w_1 \cdots w_n | A \stackrel{*}{\Rightarrow} w_i \cdots w_j) = \beta_A(i,j)\alpha_A(i,j)$

- Inside probability: calculated bottom-up (CYK-style)
- Outside probability: calculated top-down.
- Sentence probability can be calculated in many ways.

Parsing (1)

- In PCFG parsing, we want to compute the most probable parse tree (= most probable derivation) given an input sentence w.
- This means that we are disambiguating: Among several readings, we search for the best.
- Sometimes, the k best are searched for ($k > 1$).
- During parsing, we must make sure that updates on probabilities (because a better derivation has been found for a non-terminal) do not require updates on other parts of the chart. $\Rightarrow$ the order should be such that an item is used within a derivation only when its final probability is reached.

Parsing (2)

We can extend the symbolic CYK parser to a probabilistic one. Instead of summing over all derivations (as in the computation of the inside probability), we keep the best one.

Assume a three-dimensional chart C (non-terminal, start index, length).

$C_{A,i,l} := 0$ for all A, i, l
$C_{A,i,1} := p$ if $p : A \to w_i \in P$ scan
for all $l \in [1..n]$:
  for all $i \in [1..n-l+1]$:
    for every $p : A \to B\ C$:
      for every $l_1 \in [1..l-1]$:
        $C_{A,i,l} = max\{C_{A,i,l}, p \cdot C_{B,i,l_1} \cdot C_{C,i+l_1,l-l_1}\}$ complete

Parsing (3)

We extend this to a parser.

- The parser can also deal with unary productions $A \to B$.
- Every chart field has three components, the probability, the rule that has been used and, if the rule is binary, the length l_1 of the first righthand side element.
- We assume that the grammar does not contain any loops $A \overset{+}{\Rightarrow} A$.

Parsing (4)

$C_{A,i,1} = \langle p, A \to w_i, - \rangle$ if $p : A \to w_i \in P$ scan
for all $l \in [1..n]$ and for all $i \in [1..n-l]$:
  for all $p : A \to B\ C$ and for all $l_1 \in [1..l-1]$:
    for all $l_1 \in [1..l-1]$:
      if $C_{B,i,l_1} \neq \emptyset$ and $C_{C,i+l_1,l-l_1} \neq \emptyset$ then:
        $p_{new} = p \cdot C_{B,i,l_1}[1] \cdot C_{C,i+l_1,l-l_1}[1]$
        if $C_{A,i,l} == \emptyset$ or $C_{A,i,l}[1] < p_{new}$ then:
          $C_{A,i,l} = \langle p_{new}, A \to BC, l_1 \rangle$ binary complete
  repeat until C does not change any more:
    for every $p : A \to B$:
      if $C_{B,i,l} \neq \emptyset$ then:
        $p_{new} = p \cdot C_{B,i,l}[1]$
        if $C_{A,i,l} == \emptyset$ or $C_{A,i,l}[1] < p_{new}$ then:
          $C_{A,i,l} = \langle p_{new}, A \to B, - \rangle$ unary complete
return build_tree(S,1,n)

Parsing (5)

.1	VP → VP NP	1	NP → Det N	.3	V → eats
.6	VP → V NP	.3	V → sees	1	Det → this
.3	VP → V	.4	V → comes	.5	N → morning
.5	N → apple				

Start symbol VP, input w = *eats this morning*

l				
3	.0045, VP → VP AP, 1			
2		.5, NP → Det N, 1		
1	.09, VP → V .3, V → eats	1, Det → this	.5, N → morning	
	1	2	3	i

Parsing (6)

.1	VP → VP NP	1	NP → Det N	.3	V → eats
.6	VP → V NP	.3	V → sees	1	Det → this
.3	VP → V	.4	V → comes	.5	N → morning
.5	N → apple				

Start symbol VP, input w = *eats this morning*

l				
3	.09, VP → V NP, 1			
2		.5, NP → Det N, 1		
1	.09, VP → V .3, V → eats	1, Det → this	.5, N → morning	
	1	2	3	i

(The analysis of the VP gets revised since a better parse tree has been found.)

References

[Jurafsky and Martin, 2009] Jurafsky, D. and Martin, J. H., editors (2009). *Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall Series in Articial Intelligence. Pearson Education International. Second Edition.

[Manning and Schütze, 1999] Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing.* The MIT Press, Cambridge, Massachusetts, London, England.