Machine Learning for natural language processing

PCFG: Parameter estimation with EM

Laura Kallmeyer

Heinrich-Heine-Universität Düsseldorf

Summer 2016

Introduction

- Unsupervised parameter estimation for HMMs has been done with the EM algorithm, based on the forward and backward probabilities.
- EM can be used more generally for unsupervised parameter estimation with generative models (Dempster et al., 1977).
- Today: introduction of PCFG, inside and outside probabilities and unsupervised estimation of the probabilities of the PCFG using EM based on inside and outside probabilities.

Booth (1969); Pereira & Schabes (1992); Collins; Petrov et al. (2006)

Table of contents

1. Motivation
2. PCFG
3. Inside and outside computation
4. EM Training for probability estimation
5. Treebank refinement

Motivation

Probabilistic Context-Free Grammars (PCFG)

- are CFGs with probabilities attached to productions
- are widely used for data-driven constituency parsing

The probabilities can be estimated from unannotated data via the EM algorithm, based on **inside** and **outside** probabilities, similar to estimating HMM parameter with forward and backward probabilities.

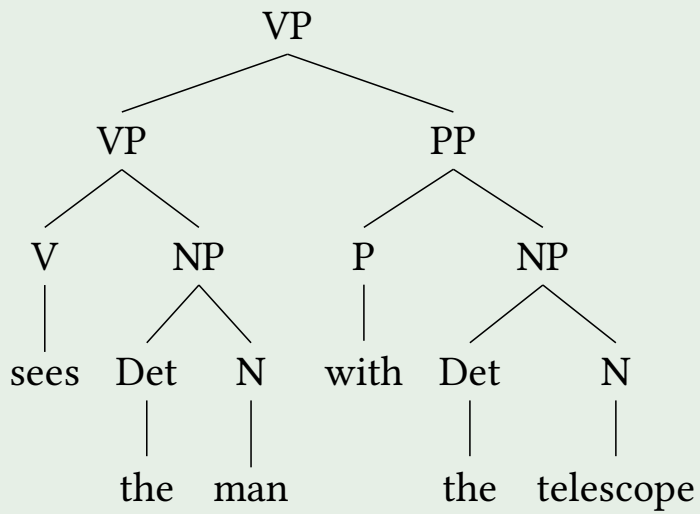
Motivation

PCFG

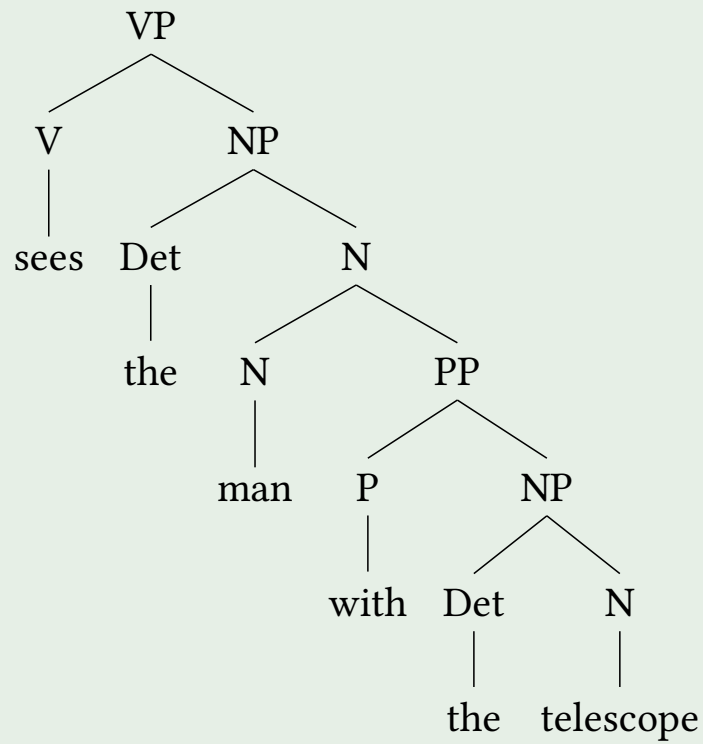
Start symbol VP

				1	Det → the
0.8	VP → V NP	1	PP → P NP	1	P → with
0.2	VP → VP PP	0.1	N → N PP	0.6	N → man
1	NP → Det N	1	V → sees	0.3	N → telescope

t_1



t_2



$$P(t_1) = 0.6 \cdot 0.8 \cdot 0.2 \cdot 0.3 = 0.0288$$

$$P(t_2) = 0.6 \cdot 0.8 \cdot 0.1 \cdot 0.3 = 0.0144$$

PCFG

We assume (without loss of generality) that our PCFGs are in Chomsky Normal Form, i.e., we adopt the following definition:

PCFG

A **probabilistic context-free grammar** is a context-free grammar $G = \langle N, T, P, S \rangle$ with an additional function $p : P \to \mathbb{R}$ such that

- N and T are alphabets of non-terminal and terminal symbols,
- $S \in N$ is a distinguished start symbol,
- P is a finite set of productions (rules) such that every rule $r \in P$ either is of the form $A \to a$ with $A \in N, a \in T$ or of the form $A \to BC$ with $A, B, C \in N$,
- p assigns probabilities to rules such that for any rule $A \to \alpha$, $0 \leq p(A \to \alpha) \leq 1$, and for every $A \in N$: $\sum_{A \to \alpha \in P} p(A \to \alpha) = 1$.

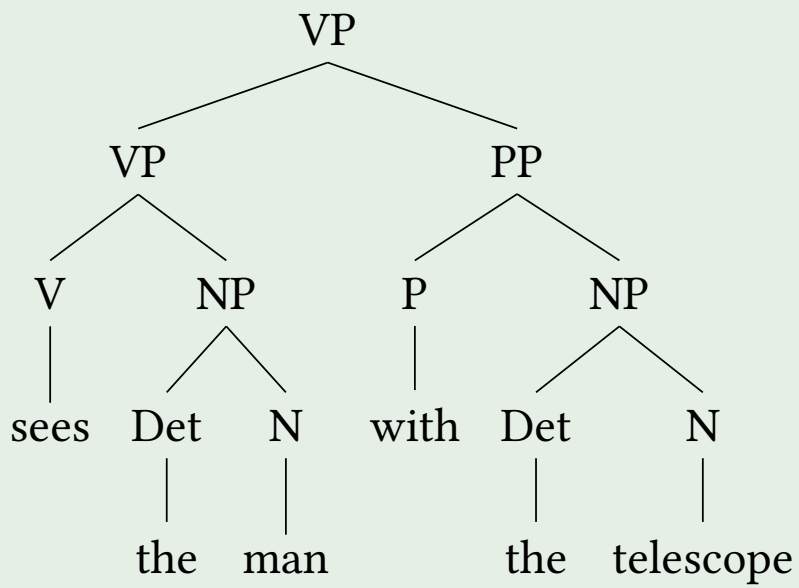
$p(A \to \alpha)$ is the conditional probability of expanding to α, given the non-terminal A, $p(A \to \alpha) = P(\alpha|A)$.

PCFG

Each internal node in a parse tree t for an input sentence w corresponds to a unique production with the label of the node being the lefthand side and the labels of its daughters (from left to right) being the righthand side. Furthermore, it spans a unique substring of the input, characterized by the indices of the first and last terminal.

PCFG

t_1



$[\text{VP}, 1, 6] \rightarrow [\text{VP}, 1, 3]\ [\text{PP}, 4, 6]$
$[\text{VP}, 1, 3] \rightarrow [\text{V}, 1, 1]\ [\text{NP}, 2, 3]$
$[\text{NP}, 2, 3] \rightarrow [\text{Det}, 2, 2]\ [\text{N}, 3, 3]$
$[\text{PP}, 3, 6] \rightarrow [\text{P}, 4, 4]\ [\text{NP}, 5, 6]$
$[\text{NP}, 5, 6] \rightarrow [\text{Det}, 5, 5]\ [\text{N}, 6, 6]$
$[\text{V}, 1, 1] \rightarrow \text{sees}$
$[\text{Det}, 2, 2] \rightarrow \text{the}$
$[\text{N}, 3, 3] \rightarrow \text{man}$
$[\text{P}, 4, 4] \rightarrow \text{with}$
$[\text{Det}, 5, 5] \rightarrow \text{the}$
$[\text{N}, 6, 6] \rightarrow \text{telescope}$

PCFG

We can define a parse tree as a set of productions with span indices.

The probability of a parse tree is the product of the probabilities of its rules with span indices:

$$P(t) = \prod_{[A,i,j] \to [B,i,k][C,k+1,j] \in t} p(A \to BC) \prod_{[A,i,j] \to a \in t} p(A \to a)$$

Let $T(w)$ be the set of all possible parse trees for w. Then we have for every $t \in T(w)$:

$$P(t|w) = \frac{P(t)}{\sum_{t' \in T(w)} P(t')}$$

and

$$P(w) = \sum_{t' \in T(w)} P(t')$$

PCFG

Problems one might want to solve:

1. **Parsing/Decoding**: Given a PCFG G and an input sentence w, determine the (k) best parse trees for w.
2. **Likelihood**: Given a PCFG G and an input sentence w, determine the likelihood of w.
3. **Supervised data-driven treebank grammar induction**: Given a treebank (= training data annotated with parse trees), read off a PCFG that maximizes the likelihood of the training data.
4. **Semi-supervised data-driven grammar induction**: Given a CFG and unannotated training data, estimate the probabilities of the CFG rules.

We will look at 2. and 4.

Inside and outside computation

Given a PCFG and an input $w = w_1 \ldots w_n$, determine the likelihood of w, i.e., compute $\sum_{t' \in T(w)} P(t')$.

We don't want to compute the probability of every parse tree separately and then sum over the results. This is too expensive.

Instead, similar to the idea of the forward and backward algorithms for HMM, we adopt a computation with tabulation, in order to share the results for common subtrees.

Inside and outside computation

Idea: We fill a $|N| \times |w| \times |w|$ matrix α where the first dimension is the id of a non-terminal, and the second and third are the start and end indices of a span. $\alpha_{A,i,j}$ gives the probability of deriving $w_i \ldots w_j$ from A or, put differently, of a parse tree with root label A and yield $w_i \ldots w_j$:

$$\alpha_{A,i,j} = P(A \overset{*}{\Rightarrow} w_i \ldots w_j | A)$$

Inside computation

1. for all $1 \leq i \leq |w|$ and $A \in N$:
   if $A \to w_i \in P$, then $\alpha_{A,i,i} = p(A \to w_i)$, else $\alpha_{A,i,i} = 0$
2. for all $1 \leq i < j \leq |w|$ and $A \in N$:
   $\alpha_{A,i,j} = \sum_{A \to BC \in P} \sum_{k=i}^{j-1} p(A \to BC) \alpha_{B,i,k} \alpha_{C,k+1,j}$

We have in particular $\alpha_{S,1,|w|} = P(w)$.

Inside and outside computation

Inside computation

0.3: S → AS 0.6: S → AX 0.1: S → a 1: X → SA 1: A → a

input $w = a^4$

j				
4	(3.87 · 10^{-2},S), (0.069,X)	(6.9 · 10^{-2},S), (0.03,X)	($3 \cdot 10^{-2}$,S), (0.1,X)	(1,A), (0.1,S)
3	(6.9 · 10^{-2},S), (0.03,X)	($3 \cdot 10^{-2}$,S), (0.1,X)	(1,A), (0.1,S)	
2	($3 \cdot 10^{-2}$,S), (0.1,X)	(1,A), (0.1,S)		
1	(1,A), (0.1,S)			
	1	2	3	4 i

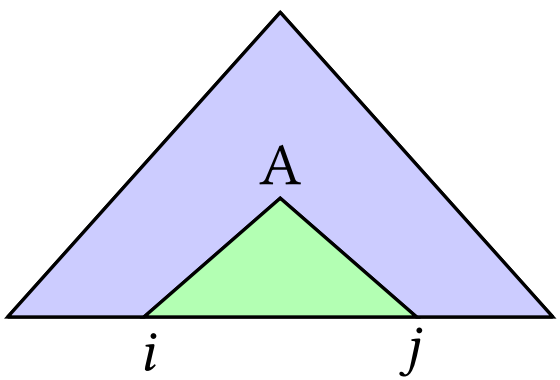
$P(aaaa) = \alpha_{S,1,4} = 0.0387$

Inside and outside computation

We can also compute the outside probability of a given non-terminal A with a span from i to j.

Inside: Sum over all possibilities for the tree below A (spanning from i to j).

Outside: Sum over all possibilities for the part of the parse tree outside the tree below A, i.e., over all possibilities to complete a A, i, j tree into a parse tree for the entire sentence.



Outside probability $\beta_{A,i,j}$

Inside probability $\alpha_{A,i,j}$

Inside and outside computation

We fill a $|N| \times |w| \times |w|$ matrix β such that $\beta_{A,i,j}$ gives the probability of deriving $w_1 \ldots w_{i-1} A w_{j+1} \ldots w_{|w|}$ from S or, put differently, of deriving a tree with root label S and yield $w_1 \ldots w_{i-1} A w_{j+1} \ldots w_{|w|}$:

$$\beta_{A,i,j} = P(S \overset{*}{\Rightarrow} w_1 \ldots w_{i-1} A w_{j+1} \ldots w_{|w|} | S)$$

We need the inside probabilities in order to compute the outside probabilities.

Outside computation

1. $\beta_{S,1,|w|} = 1$ and $\beta_{A,1,|w|} = 0$ for all $A \neq S$
2. for all $1 \leq i < j \leq |w|$ and $A \in N$:
$$\beta_{A,i,j} = \sum_{B \to AC \in P} \sum_{k=j+1}^{n} p(B \to AC) \beta_{B,i,k} \alpha_{C,j+1,k} + \sum_{B \to CA \in P} \sum_{k=1}^{i-1} p(B \to CA) \beta_{B,k,j} \alpha_{C,k,i-1}$$

Inside and outside computation

Outside computation

0.3: S → AS 0.6: S → AX 0.1: S → a 1: X → SA 1: A → a

input $w = a^3$

j				
3	(1,S), (0,A), (0,X)	(0.3,S), (0,A), (0.6,X)	$(9 \cdot 10^{-2}$,S), (0.18,X), $(3 \cdot 10^{-2}$,A)	
2	(0,S), (0,X), (0.03,A)	(0.6,S), (0,X), $(8.99 \cdot 10^{-3}$,A)		
1	(0,S), (0,X), $(6.9 \cdot 10^{-2}$,A)			
	1	2	3	i

Inside and outside computation

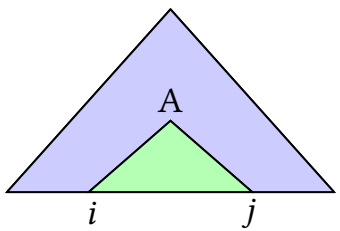
The following holds:

1. For every i, j with $1 \leq i < j \leq n$, we have

$$P(w) = \sum_{A \in N} \alpha_{A,i,j} \beta_{A,i,j}$$

2. The probability of a parse tree for w with a node labeled A that spans $w_i \ldots w_j$ is

$$P(S \overset{*}{\Rightarrow} w_1 \ldots w_{i-1} A w_{j+1} \ldots w_n \overset{*}{\Rightarrow} w_1 \ldots w_n) = \alpha_{A,i,j} \beta_{A,i,j}$$



3. In particular: $P(w) = \alpha_{S,1,|w|}$

Training

Supervised data-driven PCFG parsing: Given a treebank, read off the rules and estimate their probabilities based on the counts of the rules.

More challenging: Unsupervised parameter estimation: Given a CFG and unannotated training data, estimate the probabilities of the rules.

We use the EM algorithm, based on the inside and outside computations from the previous slides.

Training

Underlying ideas, as in the HMM parameter estimation:

- We estimate parameters iteratively: we start with some parameters and use the estimated probabilities to derive better and better parameters.
- We use our current parameters to estimate (fractional) counts of possible parse trees and possible rules. In other words, the probability mass assigned to the training corpus gets distributed among the possible parse trees.
- These fractional counts are then used to compute the parameters of the next model.

Training

For each rule $r = A \to \gamma \in P$, we start with some initial probabilities $p^{(0)}(r)$ that can be chosen randomly. In each iteration, based on the probabilities $p^{(i)}$, new probabilities $p^{(i+1)}$ are estimated.

Intuition:

$$p^{(i+1)}(A \to \gamma) = \frac{\text{expected count of } A \to \gamma}{\text{expected count of non-terminal } A}$$

more precisely

$$p^{(i+1)}(A \to \gamma) = \frac{f^{(i)}(A \to \gamma)}{\sum_{A \to \gamma' \in P} f^{(i)}(A \to \gamma')}$$

In the E-step of the algorithm, we compute the fractional counts $f^{(i)}(r)$ for all $r \in P$ and in the M-step, we re-estimate the probabilities according to these new counts.

Training

We can think of this as follows:

- Our training data are sentences $w(1), \ldots, w(N)$.
- In each iteration, based on the current probabilities, we create a treebank for training:

  For each of the sentences, the treebank contains all possible parse trees. But tree t does not occur once in the treebank, instead, it occurs $P(t)$ times.
- Consequently, when counting occurrences of rules in the treebank in order to estimate new probabilities, an occurrence of some rule r in a parse tree t does not add 1 to the count but it adds $P(t)$.
- The resulting count for rule r, summing up the probabilities of the parse trees for every occurrence of r, is then the exptected count of r.

Training

Computation of the fractional counts for a single sentence w: We distribute $P(w)$ among all the rules used in any of the parse trees of w, in accordance with the probability of these parse trees.

We have

$$P(w) \quad = \quad \alpha_{S,1,|w|}$$

and

$$\begin{aligned} P(S \overset{*}{\Rightarrow} & \; w_1 \dots w_{i-1} A w_{j+1} \dots w_{|w|} \\ \Rightarrow & \; w_1 \dots w_{i-1} B C w_{j+1} \dots w_{|w|} \\ \overset{*}{\Rightarrow} & \; w_1 \dots w_{k-1} C w_{j+1} \dots w_{|w|} \\ \overset{*}{\Rightarrow} & \; w_1 \dots w_{|w|}) \\ & = \beta_{A,i,j} \alpha_{B,i,k-1} \alpha_{C,k,j} p(A \to BC) \end{aligned}$$

Training

Computation of $C_w(A \to \gamma)$ for a sentence w

Let $G = \langle N, T, P, S \rangle$ be a PCFG with probabilities $p(r)$ for all rules $r \in P$ and let $w \in T^*$ be an input sentence.

1. Calculate the inside and outside probabilities $\alpha_{A,i,j}$ and $\beta_{A,i,j}$ for all $A \in N$ and $1 \leq i < j \leq |w|$.
2. For every rule of the form $A \to BC$:

$$C_w(A \to BC) = \sum_{1 \leq i < k \leq j \leq n} \frac{\beta_{A,i,j} \alpha_{B,i,k-1} \alpha_{C,k,j} p(A \to BC)}{\alpha_{S,1,|w|}}$$

3. For every rule of the form $A \to a$:

$$C_w(A \to a) = \sum_{1 \leq i \leq n, w_i = a} \frac{\beta_{A,i,i} p(A \to a)}{\alpha_{S,1,|w|}}$$

Training

In order to calculate the fractional count $f^{(i)}(A \to \gamma)$, sum over the counts $C_w(A \to \gamma)$ for all sentences in the training corpus:

E-step

Let our training corpus consist of sentences $w^{(1)} \ldots w^{(N)}$ and let the PCFG and its probability function p be as above.

$$f(A \to \gamma) = \sum_{1 \leq m \leq N} C_{w^{(m)}}(A \to \gamma)$$

This is the **E (expectation)** step for our parameters.

Training

From these frequencies (= fractional counts) $f(A \to \gamma)$, we can estimate new rule probabilities $\hat{p}$ towards maximizing the observed data:

M-step

For every $A \to \gamma \in P$:

$$\hat{p}(A \to \gamma) = \frac{f(A \to \gamma)}{\sum_{A \to \gamma' \in P} f(A \to \gamma')}$$

This is the **M (maximization)** step for the rule probabilities.

Training

EM algorithm for estimation of p for a PCFG; training corpus is a sequence of sentences $w^{(1)}, \ldots, w^{(N)}$

initialize p

iterate until convergence:

E-step

for every $1 \leq m \leq N$: compute $C_w(A \to \gamma)$ as above

for every $r \in P$: $f(A \to \gamma) = \sum_{1 \leq m \leq N} C_{w^{(m)}}(A \to \gamma)$

M-step

for every $A \to \gamma \in P$: $\hat{p}(A \to \gamma) = \frac{f(A \to \gamma)}{\sum_{A \to \gamma' \in P} f(A \to \gamma')}$

return p

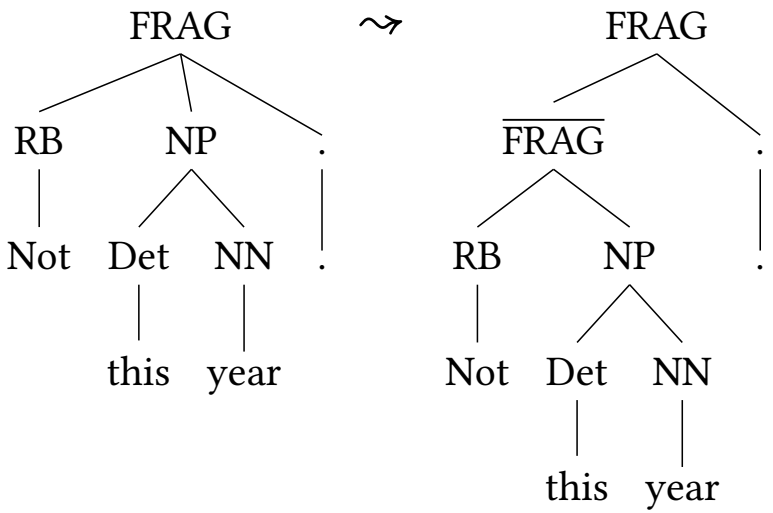
Treebank refinement

- So far, we have seen completely unsupervised training. Each iteration has a complexity of $\mathcal{O}(|N|^3|w|^3)$.
- The complexity decreases considerably if we know the parse trees of the sentences except for the node labels. I.e., we know about the bracketing of the trees.
- Pereira & Schabes (1992) show how this information can be integrated into the inside outside computation:
  1. Given a sentence w with its bracketing, we define $\overline{c}(i,j)$ as 1 if a subtree spanning $w_i \ldots w_j$ exists and otherwise it is 0.
  2. For every value $\alpha_{A,i,j}$ and $\beta_{A,i,j}$, we multiply with the factor $\overline{c}(i,j)$. Consequently all values where there is no corresponding bracketing are set to 0.
- Inside outside computation becomes linear in the size of the input.

Treebank refinement

There also have been a range of approaches to refining treebank grammars. One of the best performing approaches is the Berkeley parser Petrov et al. (2006).

- Starting point: Penn Treebank trees.
- Binarization of these trees: left-branching binarization with new intermediate non-terminals $\overline{A}$ where A is the root of the tree one wants to binarize.



- Start with the treebank grammar obtained from these trees.

Treebank refinement

Iteration for learning new labels and estimating new probabilities:

Given a current PCFG, repeat the following until no more successful splits can be found:

- Split non-terminals:
  - Split every non-terminal A into 2 new symbols A_1, A_2(e.g, NP $\rightsquigarrow$ NP_1, NP_2),
  - replace every rule $A \to \alpha$ with $A_1 \to \alpha$ and $A_2 \to \alpha$, both with the same probability as the original rule,
  - and for every occurrence of A in a righthand side of some $B \to \gamma$, replace $B \to \gamma$ with two new rules, one with A_1 instead of A and another one with A_2, dividing the probability of the original rule between these two new rules. This is done repeatedly until all old non-terminals have been removed.
  - Furthermore, add some small amount of randomness to the probabilities to break the symmetry.

Treebank refinement

- Then, probabilities are re-estimating using the inside-outside EM algorithm with the splitted grammar, based on the correct treebank bracketing and the coarser node labels in the tree-bank.
- Each split that has contributed to increasing the probability of the training data is kept and the other splits are reversed.

Result:

- POS tags get split, for instance VBZ is split into 11 different POS tags.
- Phrasal non-terminals get split as well, for example different VP categories for infinite VPs, passive VPs, intransitive VPs etc.
- The best evaluation was obtained with a resulting grammar with 1043 symbols, F_1 score of 90.2% on the Penn treebank.

References

Booth, T. 1969. Probabilistic representation of formal languages. In *Tenth annual ieee symposium on switching and automata theory*, .

Collins, Michael. ???? The inside-outside algorithm. `www.cs.columbia.edu/~mcollins/io.pdf`.

Dempster, A. P., N. M. Laird & D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* 39(1). 1–21.

Pereira, Fernando & Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th annual meeting of the association for computational linguistics*, 128–135. Newark, Delaware, USA: Association for Computational Linguistics. doi:10.3115/981967.981984. `http://www.aclweb.org/anthology/P92-1017`.

Petrov, Slav, Leon Barrett, Romain Thibaux & Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the 21st international conference on computational linguistics and 44th annual meeting of the acl*, 433–440. Sydney.