The Inside-Outside Algorithm

Michael Collins

1 Introduction

This note describes the *inside-outside* algorithm. The inside-outside algorithm has very important applications to statistical models based on context-free grammars. In particular, it is used in EM estimation of probabilistic context-free grammars, and it is used in estimation of discriminative models for context-free parsing.

As we will see, the inside-outside algorithm has many similarities to the forward-backward algorithm for hidden Markov models. It computes analogous quantities to the forward and backward terms, for context-free trees.

2 Basic Definitions

This section gives some basic definitions. We first give definitions for context-free grammars, and for a representation of parse trees. We then describe *potential functions* over parse trees. The next section describes the quantities computed by the inside-outside algorithm, and the algorithm itself.

2.1 Context-Free Grammars, and Parse Trees

The set-up is as follows. Assume that we have some input sentence $x_1 \ldots x_n$, where n is the length of the sentence. Assume in addition we have a context-free grammar (CFG) $G = (N, \Sigma, R, S)$ in Chomsky normal form, where:

- N is a finite set of non-terminal symbols.
- Σ is a finite set of terminal symbols.
- R is a finite set of rules. The grammar is in Chomsky normal form, so each rule takes one of two forms: 1) $A \rightarrow BC$ where A, B, C are all non-terminal symbols; 2) $A \rightarrow x$ where A is a non-terminal, and x is a terminal symbol.
- $S \in N$ is a distinguished start symbol.

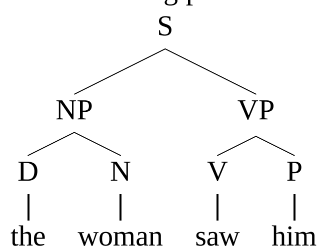
The previous class note on PCFGs (posted on the webpage) has full details of context-free grammars. For the input sentence $x_1 \ldots x_n$, the CFG defines a set of possible parse trees, which we will denote as $\mathcal{T}$.

Any parse tree $t \in \mathcal{T}$ can be represented as a set of of *rule productions*. Each rule production can take one of two forms:

- $\langle A \rightarrow B\ C, i, k, j \rangle$ where $A \rightarrow B\ C$ is a rule in the grammar, and i, k, j are indices such that $1 \leq i \leq k < j \leq n$. A rule production of this form specifies that the rule $A \rightarrow B\ C$ is seen with non-terminal A spanning words $x_i \ldots x_j$ in the input string; non-terminal B spanning words $x_i \ldots x_k$ in the input string; and non-terminal C spanning words $x_{k+1} \ldots x_j$ in the input string.

- $\langle A, i \rangle$ where A is a non-terminal, and i is an index with $i \in \{1, 2, \ldots n\}$. A rule production of this form specifies that the rule $A \rightarrow x_i$ is seen in a parse tree, with A above the i'th word in the input string.

As an example, consider the following parse tree:



This tree contains the following rule productions:

$$
\begin{array}{l}
\texttt{S} \rightarrow \texttt{NP VP}, 1, 2, 4 \\
\texttt{NP} \rightarrow \texttt{D N}, 1, 1, 2 \\
\texttt{VP} \rightarrow \texttt{V P}, 3, 3, 4 \\
\texttt{D}, 1 \\
\texttt{N}, 2 \\
\texttt{V}, 3 \\
\texttt{P}, 4
\end{array}
$$

2.2 Potential Functions

We now define potential functions over parse trees. For any rule production r (of the form $\langle A \rightarrow B\ C, i, k, j \rangle$ or $\langle A, i \rangle$) we will use $\psi(r)$ to denote the *potential* for that rule. The potential function $\psi(r)$ has the property that $\psi(r) \geq 0$ for all rule productions r.

The potential for an entire tree t is defined as follows:

$$\begin{aligned}\psi(t) &= \prod_{r\in t}\psi(r)\\ &= \left(\prod_{\langle A\rightarrow B\ C,i,k,j\rangle\in t}\psi(A\rightarrow B\ C,i,k,j)\right)\times\left(\prod_{\langle A,i\rangle\in t}\psi(A,i)\right)\end{aligned}$$

Here we write $r \in t$ if the parse tree t contains the rule production r. As an example, consider the parse tree we gave earlier. The potential for that parse tree would be

$$\psi(\mathtt{S}\rightarrow\mathtt{NP}\ \mathtt{VP},1,2,4)\times\psi(\mathtt{NP}\rightarrow\mathtt{D}\ \mathtt{N},1,1,2)\times\psi(\mathtt{VP}\rightarrow\mathtt{V}\ \mathtt{P},3,3,4)$$
$$\times\psi(\mathtt{D},1)\times\psi(\mathtt{N},2)\times\psi(\mathtt{V},3)\times\psi(\mathtt{P},4)$$

Hence to calculate the potential for a parse tree we simply read off the rule productions in the parse tree, and multiply the individual rule potentials.

Note that the potential for an entire parse tree satisfies $\psi(t)\geq 0$, because each of the individual rule potentials $\psi(r)$ satisfies $\psi(r)\geq 0$.

In practice, the rule potentials might be defined in a number of ways. In one setting, we might have a probabilistic CFG (PCFG), where each rule $\alpha\rightarrow\beta$ in the grammar has an associated parameter $q(\alpha\rightarrow\beta)$. This parameter can be interpreted as the conditional probability of seeing the rule $\alpha\rightarrow\beta$, given that the non-terminal α is being expanded. We would then define

$$\psi(A\rightarrow B\ C,i,k,j)=q(A\rightarrow B\ C)$$

and

$$\psi(A,i)=q(A\rightarrow x_i)$$

Under these definitions, for any tree t, the potential $\psi(t)=\prod_{r\in t}\psi(r)$ is simply the probability for that parse tree under the PCFG.

As a second example, consider a conditional random field (CRF) style model for parsing with CFGs (see the lecture slides from earlier in the course). In this case each rule production r has a feature vector $\underline{\phi}(r)\in\mathbb{R}^d$, and in addition we assume a parameter vector $\underline{v}\in\mathbb{R}^d$. We can then define the potential functions as

$$\psi(r)=\exp\{\underline{v}\cdot\underline{\phi}(r)\}$$

The potential function for an entire tree is then

$$\psi(t)=\prod_{r\in t}\psi(r)=\prod_{r\in t}\exp\{\underline{v}\cdot\underline{\phi}(r)\}=\exp\{\sum_{r\in t}\underline{v}\cdot\underline{\phi}(r)\}$$

Note that this is closely related to the distribution defined by a CRF-style model: in particular, under the CRF we have for any tree t

$$p(t|x_1 \dots x_n) = \frac{\psi(t)}{\sum_{t \in \mathcal{T}} \psi(t)}$$

where $\mathcal{T}$ again denotes the set of all parse trees for $x_1 \dots x_n$ under the CFG.

3 The Inside-Outside Algorithm

3.1 Quantities Computed by the Inside-Outside Algorithm

Given the definitions in the previous section, we now describe the quantities calculated by the inside-outside algorithm. The inputs to the algorithm are the following:

- A sentence $x_1 \dots x_n$, where each x_i is a word.
- A CFG (N, Σ, R, S) in Chomsky normal form.
- A potential function $\psi(r)$ that maps any rule production r of the form $\langle A \rightarrow B\ C, i, k, j \rangle$ or $\langle A, i \rangle$ to a value $\psi(r) \geq 0$.

As before, we define $\mathcal{T}$ to be the set of all possible parse trees for $x_1 \dots x_n$ under the CFG, and we define $\psi(t) = \prod_{r \in t} \psi(r)$ to be the potential function for any tree.

Given these inputs, the inside-outside algorithm computes the following quantities:

1. $Z = \sum_{t \in \mathcal{T}} \psi(t)$.
2. For all rule productions r,

$$\mu(r) = \sum_{t \in \mathcal{T} : r \in t} \psi(t)$$

3. For all non-terminals $A \in N$, for all indicies i, j such that $1 \leq i \leq j \leq n$,

$$\mu(A, i, j) = \sum_{t \in \mathcal{T} : (A, i, j) \in t} \psi(t)$$

   Here we write $(A, i, j) \in t$ if the parse tree t contains the terminal A spanning words $x_i \dots x_j$ in the input. For example, in the example parse tree given before, the following (A, i, j) triples are seen in the tree: $\langle \texttt{S}, 1, 4 \rangle$; $\langle \texttt{NP}, 1, 2 \rangle$; $\langle \texttt{VP}, 3, 4 \rangle$; $\langle \texttt{D}, 1, 1 \rangle$; $\langle \texttt{N}, 2, 2 \rangle$; $\langle \texttt{V}, 3, 3 \rangle$; $\langle \texttt{P}, 4, 4 \rangle$.

Note that there is a close correspondence between these terms, and the terms computed by the forward-backward algorithm (see the previous notes).

In words, the quantity Z is the sum of potentials for all possible parse trees for the input $x_1 \ldots x_n$. The quantity $\mu(r)$ for any rule production r is the sum of potentials for all parse trees that contain the rule production r. Finally, the quantity $\mu(A, i, j)$ is the sum of potentials for all parse trees containing non-terminal A spanning words $x_i \ldots x_j$ in the input.

We will soon see how these calculations can be applied within a particularly context, namely EM-based estimation of the parameters of a PCFG. First, however, we give the algorithm.

3.2 The Inside-Outside Algorithm

Figure 3.2 shows the inside-outside algorithm. The algorithm takes as its input a sentence, a CFG, and a potential function $\psi(r)$ that maps any rule production r to a value $\psi(r) \geq 0$. As output, it returns values for Z, $\mu(A, i, j)$ and $\mu(r)$, where r can be any rule production.

The algorithm makes use of inside terms $\alpha(A, i, j)$ and outside terms $\beta(A, i, j)$. In the first stage of the algorithm, the $\alpha(A, i, j)$ terms are calculated using a simple recursive definition. In the second stage, the $\beta(A, i, j)$ terms are calculated, again using a relatively simple recursive definition. Note that the definition of the $\beta(A, i, j)$ terms depends on the α terms computed in the first stage of the algorithm.

The α and β terms are analogous to backward and forward terms in the forward-backward algorithm. The next section gives a full explanation of the inside and outside terms, together with the justification for the algorithm.

3.3 Justification for the Inside-Outside Algorithm

We now give justification for the algorithm. We first give a precise definition of the quantities that the α and β terms correspond to, and describe how this leads to the definitions of the Z and μ terms. We then show that the recursive definitions of the α and β terms are correct.

3.3.1 Interpretation of the α Terms

Again, take $x_1 \ldots x_n$ to be the input to the inside-outside algorithm. Before we had defined $\mathcal{T}$ to be the set of all possible parse trees under the CFG for the input sentence. In addition, define

$$\mathcal{T}(A, i, j)$$

Inputs: A sentence $x_1 \dots x_n$, where each x_i is a word. A CFG (N, Σ, R, S) in Chomsky normal form. A potential function $\psi(r)$ that maps any rule production r of the form $\langle A \rightarrow B\ C, i, k, j \rangle$ or $\langle A, i \rangle$ to a value $\psi(r) \geq 0$.

Data structures:

- $\alpha(A, i, j)$ for any $A \in N$, for any (i, j) such that $1 \leq i \leq j \leq n$ is the inside term for (A, i, j).
- $\beta(A, i, j)$ for any $A \in N$, for any (i, j) such that $1 \leq i \leq j \leq n$ is the outside term for (A, i, j).

Inside terms, base case:

- For all $i \in \{1 \dots n\}$, for all $A \in N$, set $\alpha(A, i, i) = \psi(A, i)$ if the rule $A \rightarrow x_i$ is in the CFG, set $\alpha(A, i, i) = 0$ otherwise.

Inside terms, recursive case:

- For all $A \in N$, for all (i, j) such that $1 \leq i < j \leq n$,

$$\alpha(A, i, j) = \sum_{A \rightarrow B\ C \in R} \sum_{k=i}^{j-1} (\psi(A \rightarrow B\ C, i, k, j) \times \alpha(B, i, k) \times \alpha(C, k+1, j))$$

Outside terms, base case:

- Set $\beta(S, 1, n) = 1$. Set $\beta(A, 1, n) = 0$ for all $A \in N$ such that $A \neq S$.

Outside terms, recursive case:

- For all $A \in N$, for all (i, j) such that $1 \leq i \leq j \leq n$ and $(i, j) \neq (1, n)$,

$$\begin{aligned} \beta(A, i, j) &= \sum_{B \rightarrow C\ A \in R} \sum_{k=1}^{i-1} (\psi(B \rightarrow C\ A, k, i-1, j) \times \alpha(C, k, i-1) \times \beta(B, k, j)) \\ &\quad + \sum_{B \rightarrow A\ C \in R} \sum_{k=j+1}^{n} (\psi(B \rightarrow A\ C, i, j, k) \times \alpha(C, j+1, k) \times \beta(B, i, k)) \end{aligned}$$

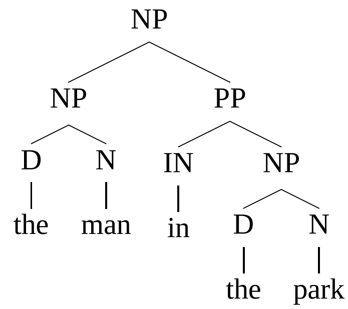
Outputs: Return

$$\begin{aligned} Z &= \alpha(S, 1, n) \\ \mu(A, i, j) &= \alpha(A, i, j) \times \beta(A, i, j) \\ \mu(A, i) &= \mu(A, i, i) \\ \mu(A \rightarrow B\ C, i, k, j) &= \beta(A, i, j) \times \psi(A \rightarrow B\ C, i, k, j) \times \alpha(B, i, k) \times \alpha(C, k+1, j) \end{aligned}$$

Figure 1: The inside-outside algorithm.

to be the set of all possible trees rooted in non-terminal A, and spanning words $x_i \ldots x_j$ in the sentence. Note that under this definition, $\mathcal{T} = \mathcal{T}(\mathtt{S}, 1, n)$ (the full set of parse trees for the input sentence is equal to the full set of trees rooted in the symbol S, spanning words $x_1 \ldots x_n$).

As an example, for the input sentence *the dog saw the man in the park*, under an appropriate CFG, one member of $\mathcal{T}(\mathtt{NP}, 4, 8)$ would be



The set $\mathcal{T}(\mathtt{NP}, 4, 8)$ would be the set of all possible parse trees rooted in NP, spanning words $x_4 \ldots x_8 =$ *the man in the park*.

Each $t \in \mathcal{T}(A, i, j)$ has an associated potential, defined in the same way as before as

$$\psi(t) = \prod_{r \in t} \psi(r)$$

We now claim the following: consider the $\alpha(A, i, j)$ terms calculated in the inside-outside algorithm. Then

$$\alpha(A, i, j) = \sum_{t \in \mathcal{T}(A,i,j)} \psi(t)$$

Thus the inside term $\alpha(A, i, j)$ is simply the sum of potentials for all trees spanning words $x_i \ldots x_j$, rooted in the symbol A.

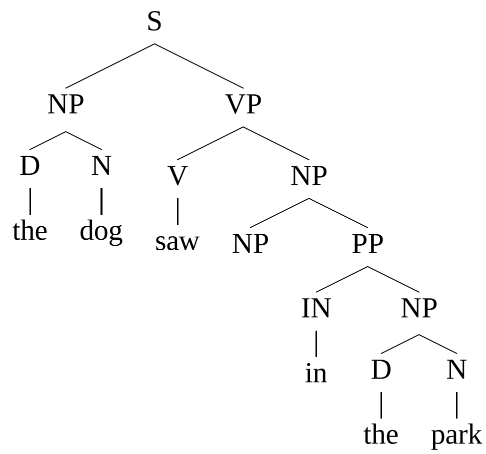
3.3.2 Interpretation of the β Terms

Again, take $x_1 \ldots x_n$ to be the input to the inside-outside algorithm. Now, for any non-terminal A, for any (i, j) such that $1 \leq i \leq j \leq n$, define

$$\mathcal{O}(A, i, j)$$

to be the set of all *outside trees* with non-terminal A, and span $x_i \ldots x_j$.

To illustrate the idea of an outside tree, again consider an example where the input sentence is *the dog saw the man in the park*. Under an appropriate CFG, one member of $\mathcal{T}(\mathtt{NP}, 4, 5)$ would be



This tree is rooted in the symbol `S`. The leafs of the tree form the sequence $x_1 \ldots x_3$ `NP` $x_6 \ldots x_n$.

More generally, an outside tree for non-terminal A, with span $x_i \ldots x_j$, is a tree with the following properties:

- The tree is rooted in the symbol `S`.
- Each rule in the tree is a valid rule in the underlying CFG (e.g., `S -> NP VP`, `NP -> D N`, `D -> the`, etc.)
- The leaves of the tree form the sequence $x_1 \ldots x_{i-1}\ A\ x_{j+1} \ldots x_n$.

Each outside tree t again has an associated potential, equal to

$$\psi(t) = \prod_{r \in t} \psi(r)$$

We simply read off the rule productions in the outside tree, and take their product.

Again, recall that we defined $\mathcal{O}(A, i, j)$ to be the set of all possible outside trees with non-terminal A and span $x_i \ldots x_j$. We now make the following claim. Consider the $\beta(A, i, j)$ terms calculated by the inside-outside algorithm. Then

$$\beta(A, i, j) = \sum_{t \in \mathcal{O}(A,i,j)} \psi(t)$$

In words, the outside term for (A, i, j) is the sum of potentials for all outside trees in the set $\mathcal{O}(A, i, j)$.

3.3.3 Putting the α and β Terms Together

We now give justification for the Z and μ terms calculated by the algorithm. First, consider Z. Recall that we would like to compute

$$Z = \sum_{t \in \mathcal{T}} \psi(t)$$

and that the algorithm has the definition

$$Z = \alpha(S, 1, n)$$

By definition, $\alpha(S, 1, n)$ is the sum of potentials for all trees rooted in S, spanning words $x_1 \ldots x_n$—i.e., the sum of potentials for all parse trees of the input sentence—so this definition is correct.

Next, recall that we would also like to compute

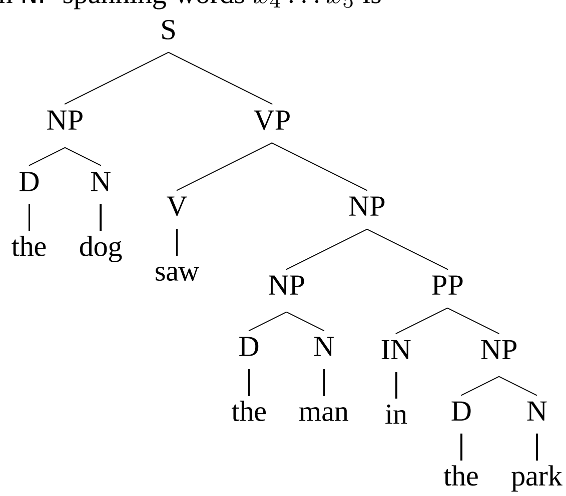
$$\mu(A, i, j) = \sum_{t \in \mathcal{T} : (A,i,j) \in t} \psi(t)$$

and that the algorithm computes this as

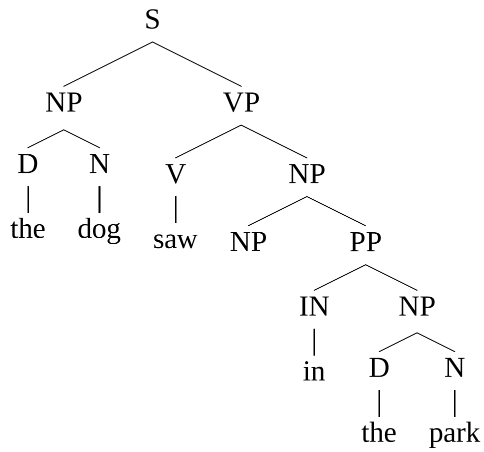
$$\mu(A, i, j) = \alpha(A, i, j) \times \beta(A, i, j)$$

How is this latter expression justified?

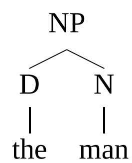
First, note that any tree with non-terminal A spanning words $x_i \ldots x_j$ can be decomposed into an outside tree in $\mathcal{O}(A, i, j)$ and an inside tree in $\mathcal{T}(A, i, j)$. For example, consider the example used above, with the triple $(\mathtt{NP}, 4, 5)$. One parse tree that contains an NP spanning words $x_4 \ldots x_5$ is



This can be decomposed into the outside tree



together with the inside tree



It follows that if we denote the outside tree by t_1, the inside tree by t_2, and the full tree by t, we have

$$\psi(t) = \psi(t_1) \times \psi(t_2)$$

More generally, we have

$$\begin{aligned}
\mu(A, i, j) &= \sum_{t \in \mathcal{T} : (A,i,j) \in t} \psi(t) && (1) \\
&= \sum_{t_1 \in \mathcal{O}(A,i,j)} \sum_{t_2 \in \mathcal{T}(A,i,j)} (\psi(t_1) \times \psi(t_2)) && (2) \\
&= \left(\sum_{t_1 \in \mathcal{O}(A,i,j)} \psi(t_1) \right) \times \left(\sum_{t_2 \in \mathcal{T}(A,i,j)} \psi(t_2) \right) && (3) \\
&= \alpha(A, i, j) \times \beta(A, i, j) && (4)
\end{aligned}$$

Eq. 1 follows by definition. Eq. 2 follows because any tree t with non-terminal A spanning $x_i \ldots x_j$ can be decomposed into a pair (t_1, t_2) where $t_1 \in \mathcal{O}(A, i, j)$, and $t_2 \in \mathcal{T}(A, i, j)$. Eq. 3 follows by simple algebra. Finally, Eq. 4 follows by the definitions of $\alpha(A, i, j)$ and $\beta(A, i, j)$.

A similar argument can be used to justify computing

$$\mu(r) = \sum_{t \in \mathcal{T} : r \in t} \psi(t)$$

as

$$\begin{aligned}
\mu(A, i) &= \mu(A, i, i) \\
\mu(A \to B\ C, i, k, j) &= \beta(A, i, j) \times \psi(A \to B\ C, i, k, j) \times \alpha(B, i, k) \times \alpha(C, k+1, j)
\end{aligned}$$

For brevity the details are omitted.

4 The EM Algorithm for PCFGs

We now describe a very important application of the inside-outside algorithm: EM estimation of PCFGs. The algorithm is given in figure 2.

The input to the algorithm is a set of training examples $x^{(i)}$ for $i = 1 \dots n$, and a CFG. Each training example is a sentence $x_1^{(i)} \dots x_{l_i}^{(i)}$, where l_i is the length of the sentence, and each $x_j^{(i)}$ is a word. The output from the algorithm is a parameter $q(r)$ for each rule r in the CFG.

The algorithm starts with initial parameters $q^0(r)$ for each rule r (for example these parameters could be chosen to be random values). As is usual in EM-based algorithms, the algorithm defines a sequence of parameter settings $q^1, q^2, \dots q^T$, where T is the number of iterations of the algorithm.

The parameters q^t at the t'th iteration are calculated as follows. In a first step, the inside-outside algorithm is used to calculate *expected counts* $f(r)$ for each rule r in the PCFG, under the parameter values q^{t-1}. Once the expected counts are calculated, the new estimates are

$$q^t(A \to \gamma) = \frac{f(A \to \gamma)}{\sum_{A \to \gamma \in R} f(A \to \gamma)}$$

4.1 Calculation of the Expected Counts

The calculation of the expected counts $f(r)$ for each rule r is the critical step: we now describe this in more detail. First, some definitions are needed. We define $\mathcal{T}_i$ to be the set of all possible parse trees for the sentence $x^{(i)}$ under the CFG. We define

$$p(x, t; \underline{\theta})$$

to be the probability of sentence x paired with parse tree t under the PCFG with parameters $\underline{\theta}$ (the parameter vector $\underline{\theta}$ contains a parameter $q(r)$ for each rule r in the CFG). For any parse tree t, for any context-free rule r, we define count(t, r) to be the number of times rule r is seen in the tree t. As is usual in PCFGs, we have

$$p(x, t; \underline{\theta}) = \prod_{r \in R} q(r)^{\text{count}(t,r)}$$

Given a PCFG, and a sentence x, we can also calculate the conditional probablity

$$p(t|x;\underline{\theta}) = \frac{p(x,t;\underline{\theta})}{\sum_{t \in \mathcal{T}_i} p(x,t;\underline{\theta})}$$

of any $t \in \mathcal{T}_i$.

Given these definitions, we will show that the expected count $f^{t-1}(r)$ for any rule r, as calculated in the t'th iteration of the EM algorithm, is

$$f^{t-1}(r) = \sum_{i=1}^{n} \sum_{t \in \mathcal{T}_i} p(t|x^{(i)};\underline{\theta}^{t-1})\text{count}(t,r)$$

Thus we sum over all training examples ($i = 1 \ldots n$), and for each training example, we sum over all parse trees $t \in \mathcal{T}_i$ for that training example. For each parse tree t, we multiply the conditional probability $p(t|x^{(i)};\underline{\theta}^{t-1})$ by the count count(t,r), which is the number of times rule r is seen in the tree t.

Consider calculating the expected count of any rule on a single training example; that is, calculating

$$\text{count}(r) = \sum_{t \in \mathcal{T}_i} p(t|x^{(i)};\underline{\theta}^{t-1})\text{count}(t,r) \tag{5}$$

Clearly, calculating this quantity by brute force (by explicitly enumerating all trees $t \in \mathcal{T}_i$) is not tractable. However, the count(r) quantities can be calculated efficiently, using the inside-outside algorithm. Figure 3 shows the algorithm. The algorithm takes as input a sentence $x_1 \ldots x_n$, a CFG, and a parameter $q^{t-1}(r)$ for each rule r in the grammar. In a first step the μ and Z terms are calculated using the inside-outside algorithm. In a second step the counts are calculated based on the μ and Z terms. For example, for any rule of the form $A \rightarrow B\ C$, we have

$$\text{count}(A \rightarrow B\ C) = \sum_{i,k,j} \frac{\mu(A \rightarrow B\ C, i, k, j)}{Z} \tag{6}$$

where μ and Z are terms calculated by the inside-outside algorithm, and the sum is over all i, k, j such that $1 \leq i \leq k < j \leq n$.

The equivalence between the definitions in Eqs. 5 and 6 can be justified as follows. First, note that

$$\text{count}(t, A \rightarrow B\ C) = \sum_{i,k,j} [[\langle A \rightarrow B\ C, i, k, j\rangle \in t]]$$

where $[[\langle A \rightarrow B\ C, i, k, j\rangle \in t]]$ is equal to 1 if the rule production $\langle A \rightarrow B\ C, i, k, j\rangle$ is seen in the tree, 0 otherwise.

Hence

$$
\begin{aligned}
& \sum_{t \in \mathcal{T}_i} p(t|x^{(i)}; \underline{\theta}^{t-1}) \text{count}(t, A \rightarrow B\ C) \\
= & \sum_{t \in \mathcal{T}_i} p(t|x^{(i)}; \underline{\theta}^{t-1}) \sum_{i,k,j} [[\langle A \rightarrow B\ C, i, k, j \rangle \in t]] \\
= & \sum_{i,k,j} \sum_{t \in \mathcal{T}_i} p(t|x^{(i)}; \underline{\theta}^{t-1}) [[\langle A \rightarrow B\ C, i, k, j \rangle \in t]] \\
= & \sum_{i,k,j} \frac{\mu(A \rightarrow B\ C, i, k, j)}{Z}
\end{aligned}
$$

The final equality follows because if we define the potential functions in the inside-outside algorithm as

$$\psi(A \rightarrow B\ C, i, k, j) = q^{t-1}(A \rightarrow B\ C)$$

$$\psi(A \rightarrow i) = q^{t-1}(A \rightarrow x_i)$$

then it can be verified that

$$\sum_{t \in \mathcal{T}_i} p(t|x^{(i)}; \underline{\theta}^{t-1}) [[\langle A \rightarrow B\ C, i, k, j \rangle \in t]] = \frac{\mu(A \rightarrow B\ C, i, k, j)}{Z}$$

Inputs: Training examples $x^{(i)}$ for $i = 1 \ldots n$, where each $x^{(i)}$ is a sentence with words $x_j^{(i)}$ for $j \in \{1 \ldots l_i\}$ (l_i is the length of the i'th sentence). A CFG (N, Σ, R, S).

Initialization: Choose some initial PCFG parameters $q^0(r)$ for each $r \in R$. (e.g., initialize the parameters to random values.) The initial parameters must satisfy the usual constraints that $q(r) \geq 0$, and for any $A \in N$, $\sum_{A \rightarrow \gamma \in R} q(A \rightarrow \gamma) = 1$.

Algorithm:
For $t = 1 \ldots T$

- For all r, set $f^{t-1}(r) = 0$
- For $i = 1 \ldots n$
  - Use the algorithm in figure 3 with inputs equal to the sentence $x^{(i)}$, the CFG (N, Σ, R, S), and parameters q^{t-1}, to calculate count(r) for each $r \in R$. Set

    $$f^{t-1}(r) = f^{t-1}(r) + \text{count}(r)$$

    for all $r \in R$.
- Re-estimate the parameters as

  $$q^t(A \rightarrow \gamma) = \frac{f^{t-1}(A \rightarrow \gamma)}{\sum_{A \rightarrow \gamma \in R} f^{t-1}(A \rightarrow \gamma)}$$

  for each rule $A \rightarrow \gamma \in R$.

Output: parameters $q^T(r)$ for all $r \in R$ of the PCFG.

Figure 2: The EM algorithm for PCFGs.

Inputs: A sentence $x_1 \ldots x_n$, where each x_i is a word. A CFG (N, Σ, R, S) in Chomsky normal form. A parameter $q(r)$ for each rule $r \in R$ in the CFG.

Algorithm:

- Run the inside-outside algorithm with inputs as follows: 1) the sentence $x_1 \ldots x_n$; 2) the CFG (N, Σ, R, S); 3) potential functions

$$\psi(A \rightarrow B\ C, i, k, j) = q(A \rightarrow B\ C)$$

$$\psi(A \rightarrow i) = q(A \rightarrow x_i)$$

  where $q(A \rightarrow x_i)$ is defined to be 0 if the rule $A \rightarrow x_i$ is not in the grammar

- For each rule of the form $A \rightarrow B\ C$,

$$\text{count}(A \rightarrow B\ C) = \sum_{i,k,j} \frac{\mu(A \rightarrow B\ C, i, k, j)}{Z}$$

  where μ and Z are terms calculated by the inside-outside algorithm, and the sum is over all i, k, j such that $1 \leq i \leq k < j \leq n$.

- For each rule of the form $A \rightarrow x$,

$$\text{count}(A \rightarrow x) = \sum_{i: x_i = x} \frac{\mu(A, i)}{Z}$$

Outputs: a count count(r) for each rule $r \in R$.

Figure 3: Calculating expected counts using the inside-outside algorithm.