Return to Main

Objectives

Training:
Independence
Inside Probability
Outside Probability
Accumulation
Reestimation Equation

Recognition:
Chart Parsing
Example

On-Line Resources:
Manning: Probabilistic Models of Language Structure
Manning: Statistical NLP
Gazdar: NLP
Stolcke: CFG Parsing

LECTURE 30: PROBABILISTIC CONTEXT FREE GRAMMARS

- Objectives:
  - Training probabilistic context free grammars
  - Inside outside algorithm
  - Recognition using chart parsing

This lecture combines material from the course textbook:

> X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

LECTURE 30: PROBABILISTIC CONTEXT FREE GRAMMARS

- Objectives:
  - Training probabilistic context free grammars
  - Inside outside algorithm
  - Recognition using chart parsing

This lecture combines material from the course textbook:

X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

TRAINING PROBABILISTIC CONTEXT FREE GRAMMARS (PCFG)

The *training problem* involves estimating the probability of each rule in a probabilistic context free grammar (PCFG). Let define the probability of a rule, $A \rightarrow \alpha$ by $P(A \rightarrow \alpha | G)$.

The simplest approach to learning these probabilities would be to use a Viterbi-style training algorithm and approximate these probabilities by the number of times a rule was used divided by the total number of times a rule could have been used. This requires a hard decision. We seek a method analogous to Baum-Welch training in which we can make soft decisions:

$$P(A \rightarrow \alpha_j | G) = \frac{C(A \rightarrow \alpha_j)}{\sum_{i=1}^{m} C(A \rightarrow \alpha_i)}$$

where $C(\bullet)$ denotes the number of times each rule is used.

If you have hand-annotated data (parse trees), you can estimate these probabilities directly. However, such data is expensive to develop, and hence is in short supply (especially since language model training requires enormous amounts of text for LVCSR applications).

To estimate these probabilities from data, we can use EM techniques to derive the *inside-outside* algorithm. To do this, we must make one important indepdence assumption:

> **The probability of a constituent being derived by a rule is independent of how the constituent is being used as a subconstituent.**

For example, we assume the probability of a noun phrase rule (NP) is the same no matter where this NP rule is used (e.g, whether it is used as a subject of object of a verb).

INSIDE PROBABILITY CALCULATION

Let the word sequence $\boldsymbol{W} = w_1 w_2 \ldots w_T$ be generated by the PCFG, G with rules on the Chomsy normal form:

$$A_i \rightarrow A_m A_n$$
$$A_i \rightarrow w_i$$

where w_i is a terminal symbol (it cannot be further expanded) and A_m,A_n are non-terminal symbols.

The probability for these rules must satisfy the following constraint:

$$\sum_{m,\,n} P(A_i \rightarrow A_m A_n \,|\, G) + \sum_{l} P(A_i \rightarrow w_i \,|\, G) = 1 \qquad \forall i$$

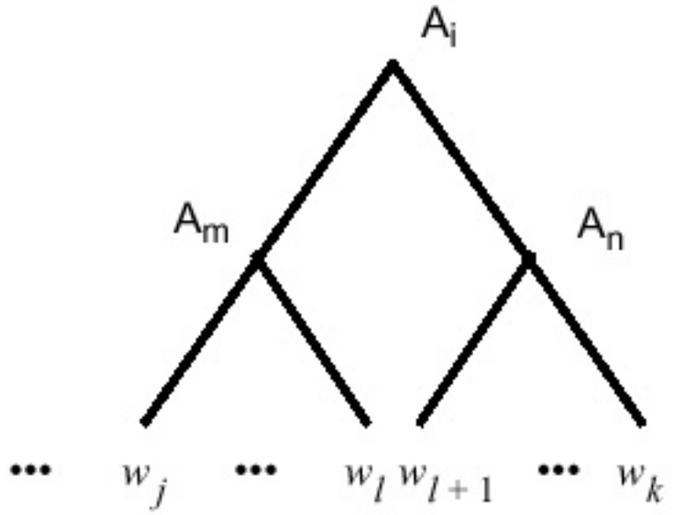
A non-terminal symbok, A_k, can generate a sequence of words $w_j w_{j+1} \ldots w_k$. We define the constituent probability:

$$Inside(j, A_i, k) = P(A_i \Rightarrow w_j w_{j+1} \ldots w_k \,|\, G)$$

This can be computed recursively as:

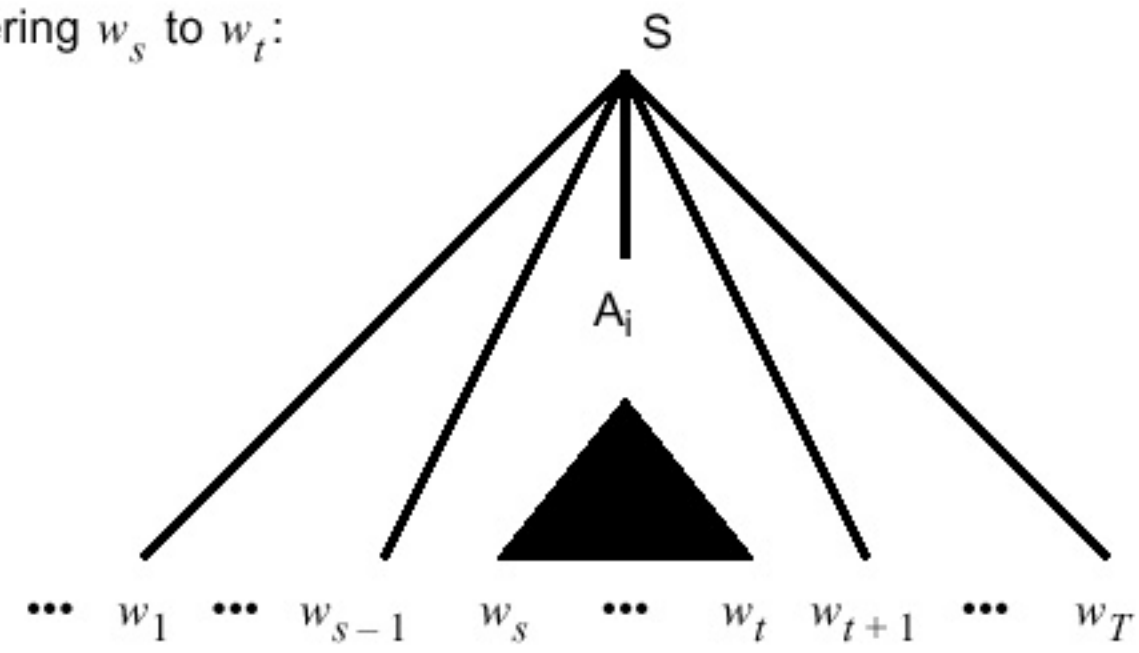
$$Inside(j, A_i, k) = \sum_{m,\,n} \sum_{l=j}^{k-1} P(A_i \rightarrow A_m A_n) P(A_m \Rightarrow w_j \ldots w_l) P(A_n \Rightarrow w_{l+1} \ldots w_k)$$
$$= \sum_{m,\,n} \sum_{l=j}^{k-1} P(A_i \rightarrow A_m A_n) Inside(j, A_m, k) Inside(l+1, A_n, k)$$

The inside probability is the sum of the probabilities of all derivations for the section over the span from j to k:



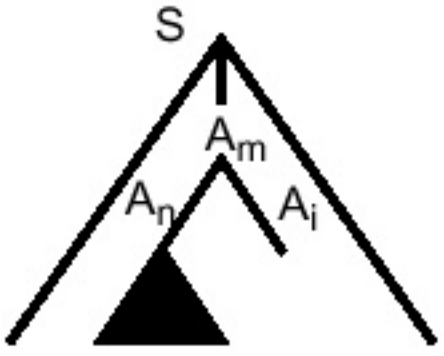
OUTSIDE PROBABILITY CALCULATION

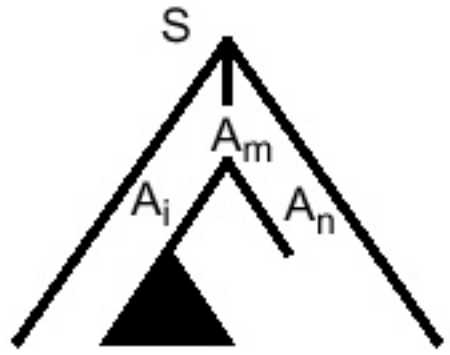
We can also define an outside probability for a non-terminal node, A_i, covering w_s to w_t:



$$Outside(s, A_t, t) = P(S \Rightarrow w_1 \ldots w_{s-1} A_i w_t \ldots w_T)$$

After the inside probabilities are computed bottom-up, we can compute the outside probabilities top-down. For each non-terminal symbol, A_i, there are one of two possible configurations $A_m \rightarrow A_n A_i$ or $A_m \rightarrow A_i A_n$:





We must consider both possibilities:

$$Outside(s, A_t, t) = P(S \Rightarrow w_1 \ldots w_{s-1} A_i w_t \ldots w_T)$$

$$= \sum_{m,\,n} \left\{ \begin{array}{l} \displaystyle\sum_{l=1}^{s-1} P(A_m \rightarrow A_n A_i) P(A_n \Rightarrow w_1 \ldots w_{s-1}) P(S \Rightarrow w_1 \ldots w_{s-1} A_i w_t \ldots w_T) \\ + \displaystyle\sum_{l=t+1}^{T} P(A_m \rightarrow A_i A_n) P(A_n \Rightarrow w_{t+1} \ldots w_l) P(S \Rightarrow w_1 \ldots w_{s-1} A_m w_{l+1} \ldots w_T) \end{array} \right\}$$

$$= \sum_{m,\,n} \left\{ \begin{array}{l} \displaystyle\sum_{l=1}^{s-1} P(A_m \rightarrow A_n A_i) Inside(l, A_n, s-1) Outside(l, A_m, t) \\ + \displaystyle\sum_{l=t+1}^{T} P(A_m \rightarrow A_i A_n) Inside(t+1, A_n, l) Outside(s, A_m, l) \end{array} \right\}$$

ACCUMULATING COUNTS FOR RULE PROBABILITIES

The inside and outside probabilities are used to compute the sentence probability as:

$$P(S \Rightarrow w_1 \ldots w_T) = \sum_i Inside(s, A_i, t) Outside(s, A_t, t) \qquad \forall s \leq t$$

The probability for the entire sentence can be computed using the forward probability:

$$P(S \Rightarrow \boldsymbol{W} | G) = Inside(1, S, T)$$

The probability that a particular rule, $A_i \rightarrow A_m A_n$, is used to cover the span $w_s \ldots w_t$ given the sentence and grammar is:

$$\xi(i, m, n, s, t) = P(A_i \rightarrow w_s \ldots w_t, A_i \rightarrow A_m A_n | (S \Rightarrow \boldsymbol{W}, G))$$

$$= \frac{1}{P(S \Rightarrow \boldsymbol{W}, G)} \sum_{k=s}^{t-1} P(A_i \rightarrow A_m A_n) Inside(s, A_m, k) Inside(k+1, A_n, t) Outside(s, A_i, t)$$

REESTIMATION EQUATION

Summing these counts across all sentences gives us an estimate of the number of times a rule has been used. Dividing by the total counts of productions used for each non-terminal gives:

$$P(A_i \rightarrow A_m A_n | G) = \frac{\sum_{s=1}^{T-1} \sum_{t=s+1}^{T} \xi(i, m, n, s, t)}{\sum_{m,n} \sum_{s=1}^{T-1} \sum_{t=s+1}^{T} \xi(i, m, n, s, t)}$$

In a similar manner, we can estimate $P(A_i \rightarrow w_m | G)$, the probability a terminal symbol rule was used.

This algorithm can also be used to select rules (learn rules), dynamical prune rules, etc.

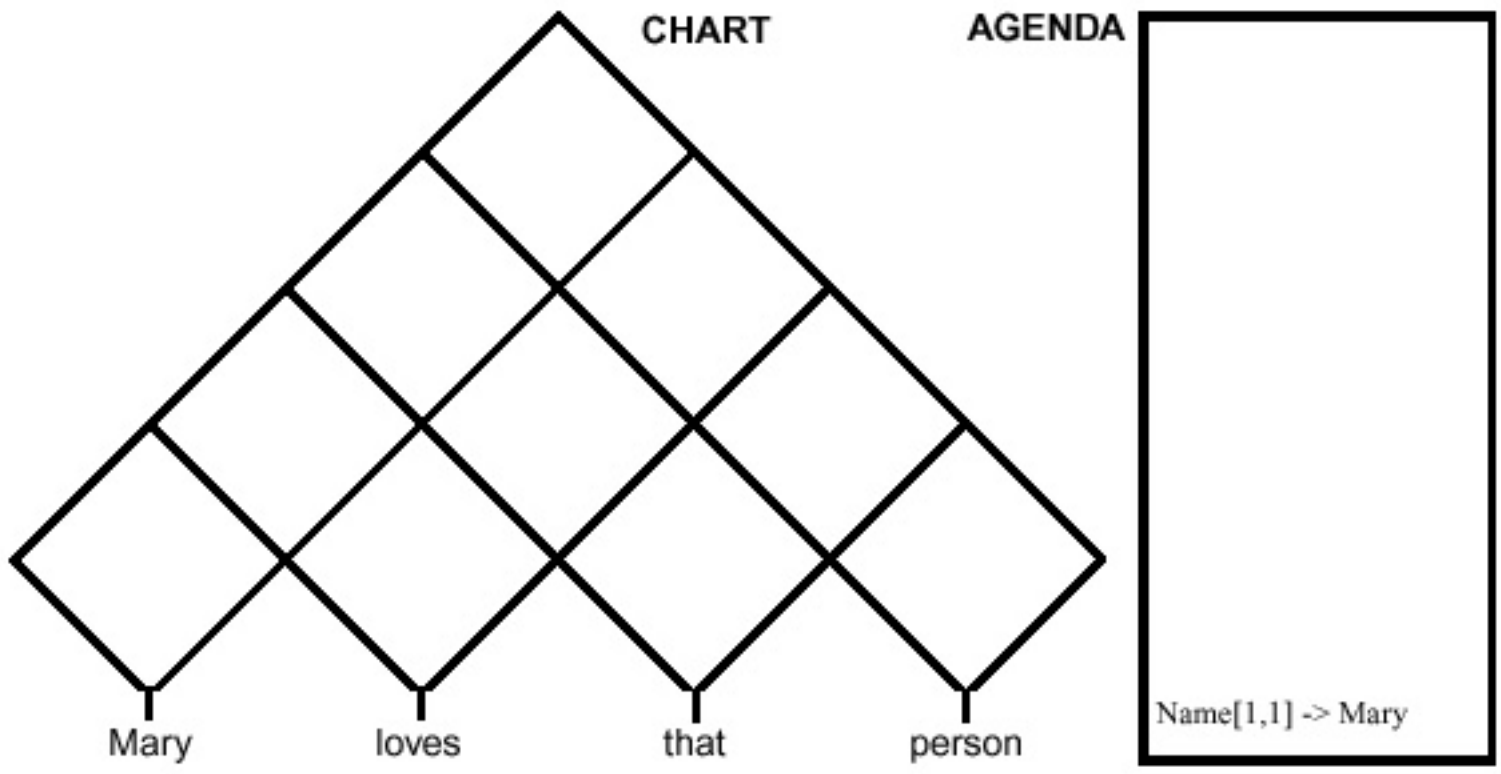
RECOGNITION: DATA-DIRECTED SEARCH VIA CHART PARSING

The recognition problem requires computation of $P(S \Rightarrow W | G)$. This can be performed using a probabilistic chart parser operating in breadth-first mode (and using some notion of beam pruning):

Step	Description
1	**Initialization**: Define a list called *chart* to store active arcs, and a list called an *agenda* to store active constituents until they are added to the chart.
2	**Repeat**: Repeat steps 2 to 7 until there is no input left.
3	**Push and pop the agenda**: If the agenda is empty, look up the interpretations of the next word in the input and push them to the agenda. Pop a constituent C from the agenda. If C corresponds to position from w_i to w_j of the input sentence, we denote it C[i,j].
4	**Add C to the chart**: Insert C[i,j] into the chart.
5	**Add active arcs**: For each rule, add to the chart an active arc of the form $X[i, j] \rightarrow \circ CY$ where $\circ$ denotes the point (key) after which things are not matched.
6	**Move $\circ$ forward**: For any active arc, add a new active arc of the form $X[1, j] \rightarrow Y \ldots C \circ \ldots Z$.
7	**Add new constituents**: For any active arc of the form $X[1, l] \rightarrow Y \ldots \circ C$, add a new constituent of type X[1,j] to the agenda..
8	**Exit**: If S[1,n] is in the chart, where n is the lenght of the input, terminate. (We can also recover all possible interpretations.)

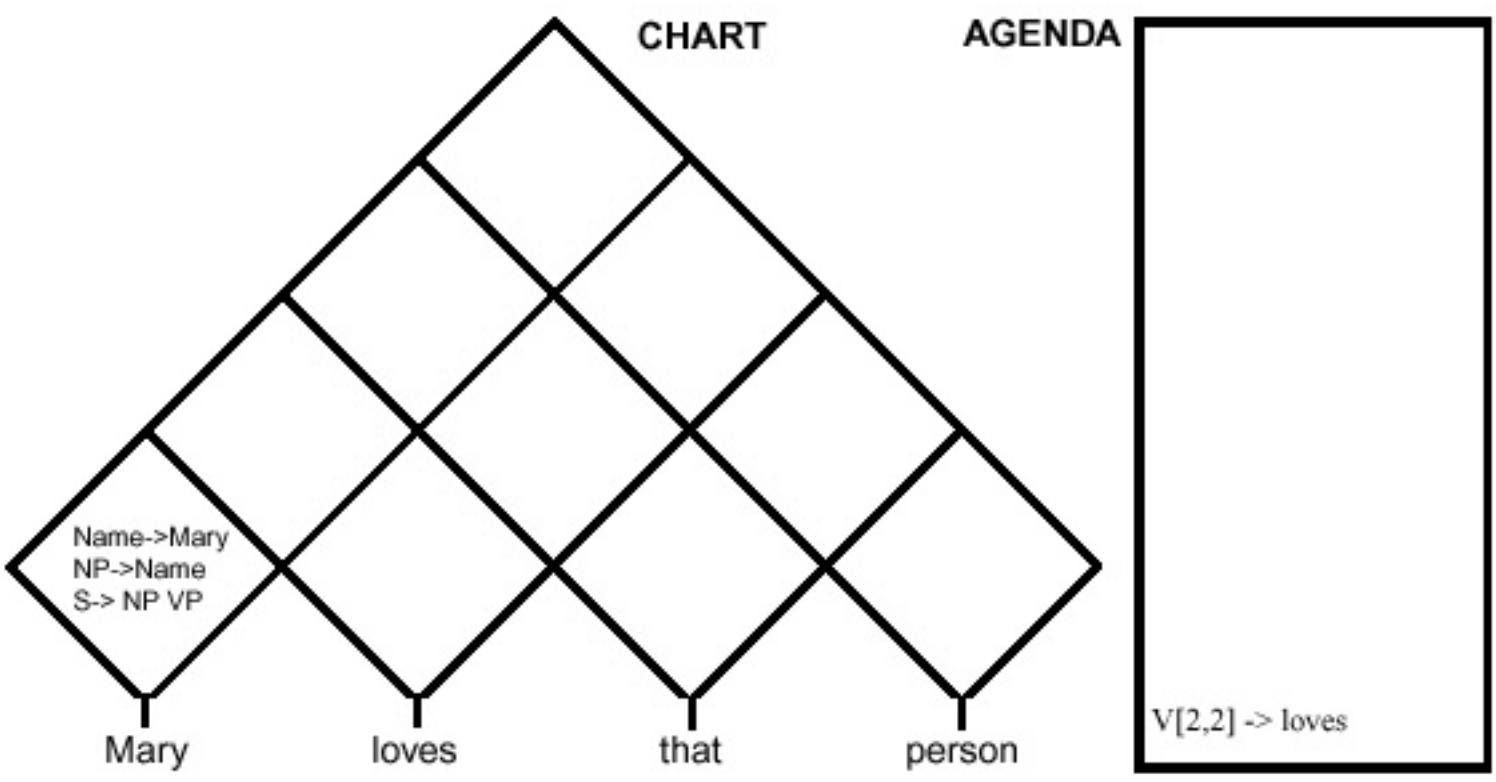
A CHART PARSING EXAMPLE

Initialization:



After "Mary", the chart now has rules:

- "Name -> Mary"
- "NP -> Name"
- S -> NP º VP



The chart after the entire sentence is parsed: