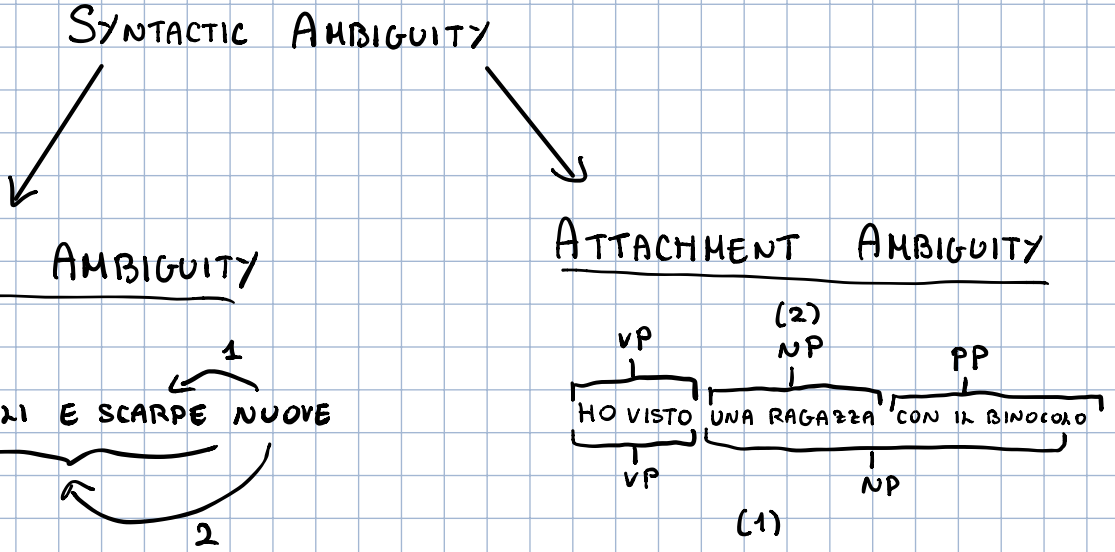
NLP - LEZIONE 11

DEL 07/11/2019

PCFG (MICHELE & SAMIR)

AMBIGUITÁ

Abbiamo due tipologie di AMBIGUITÁ SINTATTICHE, che sono



Non sappiamo se nie gli stivali che le scarpe sono nuove (1) oppure se solo le scarpe sono nuove.

Notiamo che secondo Chomsky l'ambiguità sintattica PREVALE su quella semantica in quanto la semantica è definita sul PARSE TREE.

PROBABILISTIC CF-GRAMMARS (PCFG)

Una PCFG è una quadrupla $\langle N, \Sigma, R, S \rangle$ simile a una tipica CF grammar ma con la differenza che le regole hanno la seguente forma

$$A \longrightarrow \beta \, [P]$$

con,

- A non-terminale;
- β stringa di terminali e non-terminali;
- $P \in [0,1]$ t.c. $P := P(\beta | A)$, ovvero la prob. che dato un non-terminale A, questo rarte e β.

Notiamo che vale la seguente EQUIVALENZA

$$\forall A \in NT: \sum_{\beta : A \to \beta \in P \in R} P = 1$$

Le PCFG sono state introdotte al fine di avere uno strumento che possiamo utilizzare per gestire l'ambiguità sintattica: la PROBABILITÀ.

Utilizzando le PCFG è infatti possibile CALCOLARE la probabilità di ogni PARSE TREE. Una volta che siamo in grado di fare questo, per rimuovere l'ambiguità sintattica ci basta considerare, data una frase S, solamente il parse tree che MASSIMIZZA la sua probabilità, o se vogliamo anche i N parse tree più probabili.

Al fine di calcolare la prob. di un PARSE TREE, utilizziamo le seguenti IPOTESI DI INDIPENDENZA:

1) <u>INDIPENDENZA DAL CONTESTO</u>

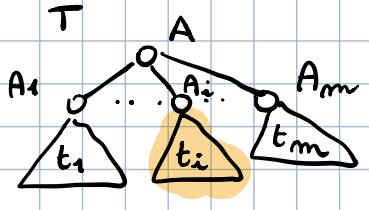
Dato un parse tree t con radice A, siano $t_1, \dots, t_m$ i sub parse-tree e siano $A_1, \dots, A_m$ i nodi radice corrispondenti.

L'ipotesi di INDIPENDENZA DAL CONTESTO ci dice che

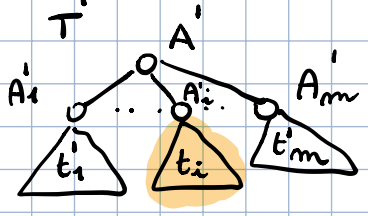
$$P(t_1 \mid A, A \to A_1 \cdots A_m, t_2, \dots, t_m) = P(t_1)$$

ovvero non importa chi sono i FRATELLI di t_1, la sua probabilità non cambia.

Graficamente abbiamo,



La prob. di t_1 è la stessa in entrambi gli alberi



2) INDIPENDENZA DALLA POSIZIONE

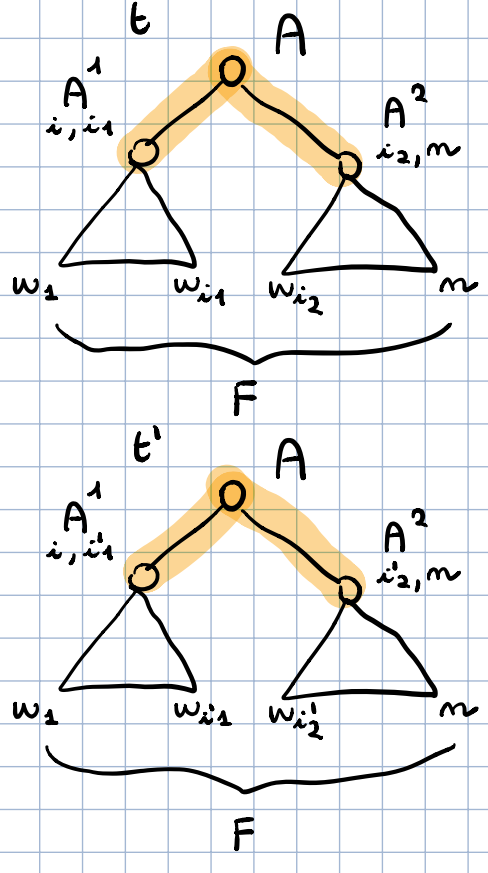
Dato un rose tree t con radice A, siano $t_1, \dots, t_m$ i sub rose-tree e siano $A^1_{1,j_1} \dots A^m_{j_m,m}$ i nodi radice corrispondenti. Con la notazione $A^1_{1\,j_1}$ vogliamo indicare che il sotto-albero t_1 SPAZIA nella sottoparola $F[1 \dots j_1]$.

L'ipotesi di INDIPENDENZA DALLA POSIZIONE ci dice che

$$P(A_{1,m} \to A^1_{1,j_1} \dots A^m_{j_m,m}) = P(A_{1,m} \to A^1 \dots A^m)$$

ovvero che non ci interessa la posizione relativa del non terminale nella produzione.

Graficamente abbiamo,



Anche se li interni non terminali A^1 e A^2 mariano la stringa F in due modi diversi in t e t', la lunghezza di ramo da A e A^1A^2 è la STESSA.

Supponiamo di avere due alberi t_1 e t_2 per una data SENTENCE S. Vogliamo trovare l'albero t^* tale che

$$t^* = \text{ARGMAX} \{ P(t_1|S), P(t_2|S) \}$$

Come facciamo?

Notiamo inizialmente che non è possibile STIMARE $P(t|S)$ in quanto, anche utilizzando BAYES, è quasi IMPROBABILE trovare la STESSA FRASE con tanti ALBERI DIFFERENTI all'interno di un corpus.

Fortunatamente è possibile dim. il seguente fatto: la probabilità di un PARSE TREE T è pari al PRODOTTO DELLE PROBABILITÀ ASSOCIATE ALLE n REGOLE UTILIZZATE PER ESPANDERE OGNUNO DEGLI n NODI NON-TERMINALI.

Formalmente,

$$P(t|G) = \prod_{(A \to \gamma) \in t} P(A \to \gamma) \qquad (1)$$

Inoltre, dato che $P(S|T) = 1$, abbiamo

$$P(T, S) = P(S|T) \cdot P(T) = P(T) \qquad (2)$$

Andiamo adesso a dimostrare (1)...

DIM.: Procediamo per induzione su $n :=$ # di produzioni utilizzate per ottenere t.

- CASO BASE, $n = 1$

  Se t è formato da una sola produzione $A_{11} \to a$, allora abbiamo che

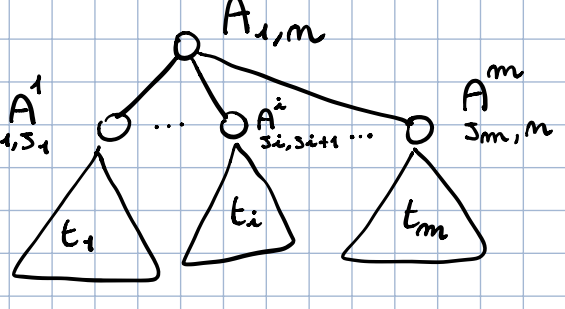
  $$P(t \mid G) = P(A_{11} \to a) \overset{\text{(IND. DALLA POSIZIONE)}}{=} P(A \to a)$$

- PASSO INDUTTIVO, $n > 1$

  Sia $A_{1,n}$ la radice di t, siano $t_1, \dots, t_m$ i sottoalberi di t, e siano $A^1_{1,s_1}, \dots, A^m_{s_m,n}$ le radici dei rispettivi sottoalberi.

  Con A^1_{1,s_1} stiamo nuovamente indicando il fatto che il non-terminale A^1 mappa nella SOTTOFRASE $F[1 \dots s_1]$.

  Graficamente, t



La probabilità di t è quindi data da

$$P(t|G) = P(A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}, t_1, t_2, \dots, t_m)$$

Applicando la CHAIN RULE, che ci dice che $P(B, C) = P(B|C) \cdot P(C)$, otteniamo la seguente

$$P(A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}, t_1, t_2, \dots, t_m) =$$

$$P(t_1 | A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}, t_2, \dots, t_m) \cdot$$

$$P(t_2 | A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}, t_3, \dots, t_m) \cdot$$

$$\vdots$$

$$P(t_m | A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}) \cdot$$

$$P(A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n} | A_{1,n}) \cdot P(A_{1,n})$$

Notiamo che dalle ipotesi di indipendenza enunciate prima otteniamo che $\forall i = 1, \dots, m$:

i) $P(t_1 | A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}, t_2, \dots, t_m) = P(t_i)$

ii) $P(A_{1,n}, A_{1,n} \to A^1_{1,s_1} \cdots A^m_{s_m,n}) = P(A, A \to A^1 \cdots A^m)$

Infine notiamo che, dato che la radice di t, A, è FISSA, abbiamo che

$$P(A, A \to A^1 \cdots A^m) = P(A \to A^1 \cdots A^m)$$

Mettendo tutto insieme e applicando l'IPOTESI INDUTTIVA, che ci dice che $P(t_i) = \prod_{A \to \gamma \in t_i} P(A \to \alpha)$, troviamo

$$\begin{aligned} P(t|G) &= P(A_{1,m}, A_{1,m} \to A^1_{1,s_1} \cdots A^m_{s_m,m}, t_1, t_2, \dots, t_m) \\ &= \prod_{i=1}^{m} P(t_i) \cdot P(A \to A^1 \cdots A^m) \\ &= \prod_{i=1}^{m} \prod_{A \to \gamma \in t_i} P(A \to \gamma) \cdot P(A \to A^1 \cdots A^m) \\ &= \prod_{A \to \gamma \in t} P(A \to \gamma) \end{aligned}$$

■

LANGUAGE MODELS ~ (1:06:00 min)

Il LANGUAGE MODELING viene utilizzato per calcolare la probabilità di una FRASE in una LINGUA, e viene utilizzato come tool di GENERAZIONE, come ad esempio durante la TRADUZIONE.

La probabilità di una frase NON AMBIGUA è la probabilità del suo unico parse tree. Per le frasi AMBIGUE invece abbiamo

$$P(S) = \sum_{T \in \text{TREES}(S)} P(T)$$

"Insieme dei parse trees di S.

CYK PER PCFG's

È possibile estendere l'algoritmo CYK per gestire "CFG's messe in forma CNF.

Per fare questo si MEMORIZZA, per ogni non-term. nella matrice un valore di prob. Quando confrontiamo due non-term A, B e troviamo un match con la regola $C \to AB$ [x] la prob. del nuovo non-term C sarà data da

$$P(C) = P(A) \cdot P(B) \cdot P(C \to AB)$$

dove P(A) e P(B) sono i valori di prob. associati ai simboli A e B.

Per diminuire la COMPLESSITÀ dell'algoritmo vogliamo scegliere di mantenere solamente le N prob. più alte. Questo ci permette di confrontare

al più $N \cdot N$ coppie di non-terminali ad ogni passo dell'algoritmo.

STIMARE PROBABILITÀ NELLE PCFGS ~ (1:24:00 min)

Per stimare le prob. in una PCFGs abbiamo due approcci:

1) Dato un TREEBANK possiamo utilizzare la MASSIMA VEROSOMIGLIANZA per stimare le probs

$$P(\alpha \to \beta \mid \alpha) = \frac{\text{COUNT}(\alpha \to \beta)}{\sum_{\gamma} \text{COUNT}(\alpha \to \gamma)} = \frac{\text{COUNT}(\alpha \to \beta)}{\text{COUNT}(\alpha)}$$

2) Utilizzando l'algoritmo INSIDE-OUTSIDE.

ALGORITHO INSIDE-OUTSIDE

L'idea è quella di ASSUMERE dei parametri iniziali Φ e utilizzare un CORPUS al fine di aggiustare in modo ITERATIVO i valori delle varie stime. Ad esempio è possibile iniziare con una DISTRIBUZIONE UNIFORME su tutte le regole della grammatica con lo stesso non-terminale $A \to \bullet$.

La probabilità di una frase W è data da

$$P_\phi(W) = \sum_T P_\phi(W, T)$$

dove $\sum_T$ è la sommatoria su tutti i parse tree della frase W, mentre ϕ è l'insieme dei PARAMETRI utilizzati per calcolare le probabilità.

Dato un CORPUS formato dalle frasi $W_1, \dots, W_n$, la LIKELIHOOD $L(\phi)$ del corpus è data da

$$L(\phi) = P_\phi(W_1) \cdot P_\phi(W_2) \cdots P_\phi(W_n)$$

L'inside-outside algorithm parte da un insieme di parametri ϕ per OTTENERE un nuovo insieme di parametri ϕ' t.c. $L(\phi') \geq L(\phi)$. Questo ricalcolo di UPDATE dei parametri viene ripetuto fino a quando la LIKELIHOOD CONVERGE.

OSS: L'INSIDE-OUTSIDE ALGORITHM è un caso peculiare dell'EXPECTATION MAXIMIZATION algorithm e viene utilizzato per MASSIMIZZARE LOCALMENTE la LIKELIHOOD dei dati di TRAINING.

Assumiamo SENZA PERDITÀ DI GENERALITÀ che la nostra PCFG sia in CNF. Fissati i parametri ϕ l'algoritmo procede nel seguente modo per aggiornarli:

- $\phi'(A \to BC) := \dfrac{\text{count}(A \to BC)}{\sum_{\alpha} \text{count}(A \to \alpha)}$
- $\text{count}(A \to BC) := \sum_{i=1}^{N} C_{\phi}(A \to BC, W_i)$
- $C_{\phi}(A \to BC, W_i) :=$ NUMERO ATTESO di volte in cui la regola $A \to BC$ viene utilizzata per generare la stringa W_1.

- $\phi'(A \to w) := \dfrac{\text{count}(A \to w)}{\sum_{\alpha} \text{count}(A \to \alpha)}$
- $\text{count}(A \to w) := \sum_{i=1}^{N} C_{\phi}(A \to w, W_i)$
- $C_{\phi}(A \to w, W_i) :=$ NUMERO ATTESO di volte in cui la regola $A \to w$ viene utilizzata per generare la stringa W_1.

Al fine di calcolare il valore di $P_\phi(A \to \alpha, w_i)$, ovvero il # medio di volte in cui viene utilizzata la regola $A \to \alpha$ per generare la parola w_i, introduciamo le seguenti probabilità

- **INSIDE PROBABILITY**

  Denotata con $\alpha_{i,j}(A)$, rappresenta la probabilità che il non terminale A può essere utilizzato per DERIVARE la SOTTOSTRINGA $w_i \dots w_j$ nella FRASE $w = w_1 \dots w_n$ dati i parametri specificati in ϕ.

  In formule,

$$\alpha_{i,j}(A) := P_\phi(A \overset{*}{\Rightarrow} w_i \dots w_j)$$

- **OUTSIDE PROBABILITY**

  Denotata con $\beta_{i,j}(A)$, rappresenta la prob. di ottenere la stringa $w_1 \dots w_{i-1} A w_{j+1} \dots w_n$ partendo dal non-terminale S e utilizzando i parametri specificati in ϕ.

  In formule,

$$\beta_{i,j}(A) := P_\phi(S \overset{*}{\Rightarrow} w_1 \dots w_{i-1} A w_{j+1} \dots w_n)$$

Al fine di calcolare queste probabilità possiamo utilizzare le seguenti RELAZIONI RICORSIVE, che funzionano se la grammatica è in CNF

- $\alpha_{i,j}(A) = \sum_{B,C} \sum_{i \le k \le j} \phi(A \to BC) \cdot \alpha_{ik}(B) \cdot \alpha_{k+1,j}(C)$

  per $i < j$. Se $i = j$ invece

  $\alpha_{ii}(A) = \phi(A \to w_{ii})$

- $\beta_{i,j}(A) = \sum_{B,C} \sum_{i \le k < j} \phi(B \to CA)\, \alpha_{k,i-1}(C)\, \beta_{kj}(B) +$

  $+ \sum_{B,C} \sum_{n \ge k > j} \phi(B \to AC) \cdot \alpha_{j+1,k}(C) \cdot \beta_{i,k}(B)$

Utilizzando queste formule si è in grado di paragonare ϕ e ϕ'. L'algoritmo GARANTISCE che la LOG-LIKELIHOOD non decresce, ovvero che

$$LL(\phi) := \sum_{i=1}^{N} \log P_{\phi}(w_i) \le LL(\phi')$$

L'idea è quindi quella di SMETTERE quando la differenza $LL(\phi') - LL(\phi) > 0$ risulta essere ABBASTANZA PICCOLA.

L'implementazione di questi calcoli viene effettuata utilizzando l'algoritmo CYK: durante la fase di PARSING e di costruzione della CHART vengono calcolate le INSIDE PROBABILITIES con un approccio BOTTOM-UP; Successivamente vengono calcolate le OUTSIDE PROBABILITIES con un approccio TOP-DOWN.

In generale però l'algoritmo INSIDE-OUTSIDE si basa sul calcolo del NUMERO ATTESO di volte in cui applicare una data REGOLA al fine di generare una FRASE in modo da poter AGGIORNARE le stime dei parametri.