Grammar Induction

Apprendimento di strutture sintattiche tramite esempi: applicazione al linguaggio naturale

Elaborazione del Linguaggio Naturale
A.A. 2003/2004

Daniela Lepri

Sommario

- Introduzione
- Grammar Induction: Teorie e Metodi
- Tecniche di apprendimento di Linguaggi Formali
- Tecniche di apprendimento del Linguaggio Naturale
- Risultati e Conclusioni
- Riferimenti

Sommario

- **Introduzione**
  - Cos'è la Grammar Induction
  - Base induttiva: Stringhe e Sequenze
  - Paradigmi dei Linguaggi Formali
  - Paradigmi del Linguaggio Naturale
- Grammar Induction: Teorie e Metodi
- Tecniche di apprendimento di Linguaggi Formali
- Tecniche di apprendimento del Linguaggio Naturale
- Risultati e Conclusioni
- Riferimenti

Cos'è la Grammar Induction

Una definizione informale:

Processo di apprendimento di grammatiche da un insieme di informazioni sul linguaggio.

- Origini:
  Disciplina sviluppatasi intorno agli anni '60, a seguito degli studi di Gold [Gold, 1967] "Language Identification in the Limit".
- Alias:
  - "Grammatical Inference"
  - "Automatic Language Acquisition"

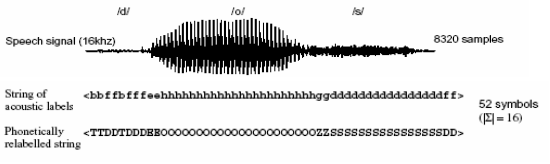
Cos'è la Grammar Induction (2)

- Oggetto di studio nei seguenti ambiti:
  - Machine Learning
  - Acquisizione del Linguaggio
  - Linguistica Computazionale
  - Pattern Recognition
  - Neural Networks
  - Linguaggi Formali
  - Teoria dell'Informazione
  - Teoria degli Automi
- Campi di applicazione:
  - Syntactic pattern recognition
  - Adaptive intelligent agents
  - Bioinformatica
  - Acquisizione del Linguaggio Naturale
  - Data Mining

Base induttiva: Stringhe e Sequenze

Esempi di sequenze (stringhe) di base, su cui inferire strutture sintattiche:

- Sequenze acustico-fonetiche



- Sequenza DNA

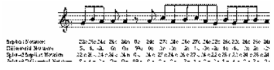
∑={A,C,G,T}
|∑|=4

....ATCGCGATGATGCGATTTGATCATGAAAAATAGATAGATAACCCATTAGGA....

Base induttiva: Stringhe e Sequenze (2)

Esempi di sequenze (stringhe) di base, su cui inferire strutture sintattiche:

- Brani musicali



∑={A, B,C ,D, E, F, G, A#, ... Ab, … Ab2, …}

- Mosse al gioco degli scacchi
- Frasi del Linguaggio Naturale

  The dog eats
- Codice sorgente di un programma

Sommario

Definizione Formale di Grammatica

$G = (N, \Sigma, R, S)$

N : Insieme finito dei simboli Non-Terminali
Σ : Insieme finito dei simboli Terminali
$S \in N$: Simbolo Non-Terminale iniziale
$R \subset (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$: Insieme di Regole

Notazione per le Regole di Produzione:

$\alpha \rightarrow \beta, \quad \alpha \in (N \cup \Sigma)^* N (N \cup \Sigma)^*, \quad \beta \in (N \cup \Sigma)^*$
$\alpha \rightarrow \beta_1 \mid \beta_2 \mid \cdots$

Grammatiche e Linguaggi

Derivazione Elementare:
$\mu\,\alpha\,\delta \underset{G}{\Longrightarrow} \mu\,\beta\,\delta$ iff $\exists(\alpha \rightarrow \beta) \in R, \quad \mu, \delta \in (N \cup \Sigma)^*$

Derivazione $\overset{*}{\underset{G}{\Longrightarrow}}$: sequenza finita di derivazioni elementari

$D_G(x)$: insieme delle *Derivazioni* di una stringa $x \in \Sigma^*$ tale che $S \overset{*}{\underset{G}{\Longrightarrow}} x$

Una G è *ambigua* se $\exists x \in \Sigma^*$ tale che $|D_G(x)| > 1$

Linguaggio generato dalla Grammatica:
$\mathcal{L}(G) = \{ x \in \Sigma^* \mid S \overset{*}{\underset{G}{\Longrightarrow}} x \}$

Gerarchia di Chomsky

0: Unrestricted

1: Context Sensitive

$\alpha \rightarrow \beta\,, \qquad |\alpha| \leq |\beta|$

➔ 2: Context Free (**CFG**)

$B \rightarrow \beta\,, \qquad B \in N$

➔ 3: Regular, Finite-State (**FS Automata**) or Right Linear

$A \rightarrow aB$ or $A \rightarrow a\,, \quad A, B \in N, \ a \in \Sigma \cup \{\lambda\}$

Regular Grammars and Finite-State Automata

- Regular Grammars (**RG**):
  $G = (N, \Sigma, R, S)$,
  $(A \rightarrow aB) \in R \ \vee \ (A \rightarrow a) \in R, \quad A, B \in N, \quad a \in \Sigma$
- Finite-State Automata (**FSA**):
  $\mathcal{A} = (Q, \Sigma, \delta, q_0, F), \quad q_0 \in Q, \quad F \subseteq Q, \quad \delta : Q \times \Sigma \rightarrow 2^Q$

Risultato di equivalenza:
Per ogni RG esiste un FSA che riconosce lo stesso linguaggio

Stochastic Grammars e Linguaggi

- Una *Stochastic Grammar G'* è la grammatica *G*, alla quale sono state associate probabilità ad ogni produzione

$$G' = (G, p), \quad G = (N, \Sigma, R, S), \quad p : R \to [0, 1]$$

- G' è *propria* se $\forall A \in N \quad \sum_{A \to \beta \in R} p(A \to \beta) = 1$
- Probabilità di una stringa *x* generata da *G'*

$$\forall x \in \Sigma^* \quad p(x|G') = \sum_{d \in D_G(x)} p(d), \qquad p(d) = \prod_{(A \to \beta) \in d} p(A \to \beta)$$

Sommario

Grammatiche che generano il Linguaggio Naturale

- Secondo il cosiddetto approccio "generativista", il linguaggio naturale può essere generato da una Grammatica Formale.
- Approccio fondato da Noam Chomsky
- Motivato dal fatto che le persone sono in grado di generare frasi che sono sintatticamente corrette e totalmente nuove (cioè che non hanno mai sentito prima)
- La Grammatica Generativa racchiude in sé le regole sintattiche del linguaggio naturale.

Grammatiche che generano il Linguaggio Naturale(2)

- *Simboli Terminali*: Categorie Lessicali o part-of-speech (Verb, Noun, Adjective, Adverb, Preposition, Determiner)
  - Per comodità, i rispettivi lessemi, sono a volte inclusi nella grammatica come simboli terminali.
- *Simboli Non-Terminali*: Categorie Sintattiche (Sentence, Noun Phrase, Verb Phrase, Prepositional Phrase, Relative clause)
- Solitamente si utilizzano Grammatiche Context Free

Perché Context Free

- E' provato che gli Automi a Stati Finiti non sono abbastanza potenti per modellare tutte le frasi di un Linguaggio Naturale

  Ad Esempio:
  - Centre-embedded sentences (Frasi "annidate")
    - Esempio semplice:
      (The book (the professor wrote) is good)
    - Esempio più complicato:
      (The book (the professor (the students (who are doing well) like) wrote) is good)
  - E' simile al linguaggio non regolare: $a^n bc^n \mid n > 0$
- Le CGF riescono a modellizzare queste strutture ricorsive.
- Una CFG per $a^n bc^n \mid n > 0$
  <S>::= a<S>c | b

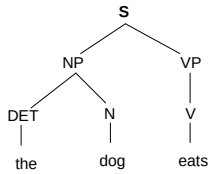
Esempio di CFG per il Linguaggio Naturale

Esempio di CFG elementare per l'Inglese:

1. S -> NP VP	Sentence
2. NP -> DET N | N	Noun Phrase
3. VP -> V | V NP	Verb Phrase
4. DET -> the | this | that | my | many | which…	Determiners
5. N -> dog | girl | park…	Nouns
6. V -> eats | eat | look…	Verbs

Esempio di generazione e Parse Tree: "the dog eats"

- Derivazione:
  S → NP VP → DET N VP → DET N V →*the dog eats*
- Tagging: [N, dog] [V, eats] [DET, the]
- Bracketing: [S [NP [DET,the] [N,dog]] [VP [V,eats]]]
- Parse Tree:



Sommario

- Introduzione
- **Grammar Induction: Teorie e Metodi**
  - Definizione Formale di GI
  - "Language identification in the limit" e Computabilità
  - Classificazione dei metodi
- Tecniche di apprendimento di Linguaggi Formali
- Tecniche di apprendimento del Linguaggio Naturale
- Risultati e Conclusioni
- Riferimenti

Definizione Formale di GI

- Teorie e metodi per apprendere induttivamente strutture sintattiche comuni ad un insieme di esempi e informazioni (*Training Data*), che si presume siano generati dallo stesso processo.

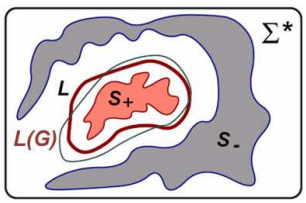
Definizione Formale di GI (2)

- Le informazioni *(Training Data)* a disposizione sono solitamente campioni di stringhe definite sullo stesso alfabeto.
  - *Positive Set :* $S_+ \subset L$
  - *Negative Set :* $S_- \subset \Sigma^* - L$
- Problema: trovare *G* tale che:
  - $S_+ \subset \mathcal{L}(G)$
  - $S_- \cap \mathcal{L}(G) = \emptyset$

Teorema di Gold: "Language Identification in the limit"

Condizione sufficiente per equivalenza tra il linguaggio generato dalla grammatica inferita L(G) *e il linguaggio target* L. *[Gold, 1967]*

- Se il Training Set $S_+ \cup S_-$ è sufficientemente ampio e rappresentativo, allora si ha: $\mathcal{L}(G) = L$



"L(G) → L per **|S+US-|→∞"**

Computabilità del Teorema di Gold

- Ogni classe di linguaggi Recursively Enumerable (cioè, per i quali si può costruire una Turing Machine) può essere identificata al limite.
- Non sempre si conosce S-
- Senza S- non si identificano al limite nemmeno i linguaggi Regolari

Classificazione dei Metodi

- Utilizzando solo S+
  - Problema: *sovra-generalizzazione*
    - Induzione di una grammatica il cui linguaggio è una generalizzazione (include strettamente) del linguaggio target.
    - Tale processo è irreversibile senza S-
- Utilizzando sia S+ che S-
  - Non si ha sovra-generalizzazione
  - I linguaggi Regolari sono Identificabili al Limite
  - I linguaggi Context-Free non lo sono
- Inferenza di Stochastic Grammars
  - Possono essere appresi efficacemente utilizzando unicamente S+
  - Controllano sovra-generalizzazione con informazioni statistiche dedotte dal Positive Set
  - Più usati in pratica

Sommario

- Introduzione
- Grammar Induction: Teorie e Metodi
- **Tecniche di apprendimento di Linguaggi Formali**
  - Apprendimento di Finite-State Automata
    - Un Algoritmo Generico con S+ e S- e RPNI
    - Un Algoritmo Generico per indurre Stochastic Grammars
    - Applicazioni e Risultati
- Tecniche di apprendimento del Linguaggio Naturale
- Risultati e Conclusioni
- Riferimenti

Apprendimento di Finite-State Automata

- Un primo contatto con gli approcci della GI
- Riconoscono Linguaggi Regolari
- Le proprietà dei Linguaggi Regolari sono ben note
- I Metodi di Inferenza sono più semplici da sviluppare e comprendere
- Ogni linguaggio può essere approssimato con precisione arbitraria con un Linguaggio Regolare (in senso stocastico)
- Molti problemi pratici sono modellati efficacemente da FSA

Il Prefix Tree Acceptor

- Insieme dei prefissi di un linguaggio

$$L \subseteq \Sigma^* : \quad Pr(L) = \{u \in \Sigma^* | uv \in L,\ v \in \Sigma^*\}$$

- Prefix Tree Acceptor sul Positive Set

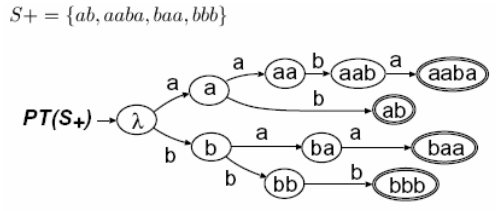
$$PT(S_+) = (Q, S, \delta, q_0, F)$$

$$Q = Pr(S_+);\ \ q_0 = \lambda;\ \ F = S_+;\ \ \delta(u, a) = ua \ \text{ iff } \ u, ua \in Pr(S_+)$$

Automa che riconosce tutte e solo le stringhe contenute in S+

PT(S+): Esempio

$S+ = \{ab, aaba, baa, bbb\}$



Un Algoritmo Generico con S+ e S-

- 1. Costruisci il Prefix Tree Acceptor PT(S+), sul Positive Set.
- 2. Scegli due stati sui quali tentare una *merge* (fusione)
- 3. Effettua la merge su tali stati
- 4. Se esiste una stringa nel Negative Set, accettata dal nuovo automa, annulla la merge e ripristina l'automa precedente
- 5. Ripeti 2-3-4 fino a quando viene raggiunto il criterio di Stop (precisione desiderata)

Regular Positive-Negative Inference (RPNI)

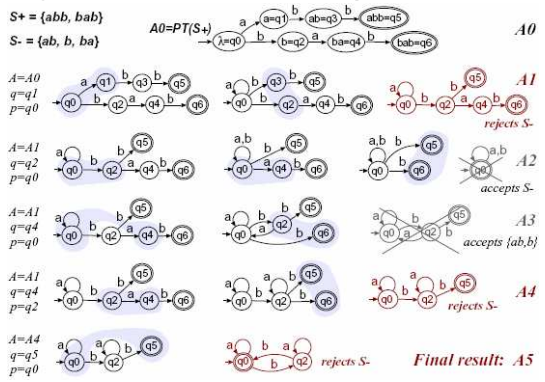
- Sviluppado da Oncina e Garcia [Oncina, 92]
- Utilizza sia S+ che S-
- Costruisce un Automa a Stati Finiti Deterministico, compatibile con S+ e S-, senza cercare di minimizzare l'automa stesso
- Riesce ad identificare efficientemente al limite, ogni linguaggio regolare.

Algoritmo RPNI

```
Algorithm  RPNI  (Regular  Positive & Negative  Inference)
Input:     S+ S-
Output:    A: DFA which accepts S+ and do not accept R-

Method: A=PT(S+); (let Q(A) denote the set of states of A)
        forall q in Q(A) in lexicographic order do
          forall p < q in lexicographic order do
              A'=merge(A,p,q)
              while A' is not deterministic do
                    select q', q'' which violate determinism
                    A'=merge(A',q',q'')
              endwhile
              if no string from S- is accepted by A'
              then A=A'
              endif
          end forall p
        end forall q
end RPNI
```

Algoritmo RPNI: Esempio

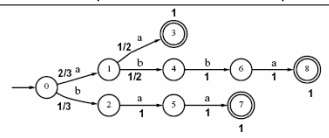


Un Algoritmo Generico per indurre Stochastic Grammars

Probabilistic Prefix Tree Acceptor (**PPTA**):

E' un PTA alle transizioni del quale sono associate delle probabilità

PPTA è *proprio*



- 1. Costruisci PPTA
- 2. **Seleziona** una coppia di stati (i, j)
- 3. Se (i, j) sono **Compatibili**, effettua la merge di i e j
- 4. Ripeti 2,3 fino a che l'Automa raggiunge il livello di precisione prestabilito

Un Algoritmo Generico per Stochastic Grammars (2)

- Vari criteri di Selezione e Compatibilità possibili
  - Selezione della coppia di stati:
    - Ordine Breadth-first nel PPTA
    - Ordine Lessico-grafico degli stati
  - Compatibilità della coppia di stati:
    - Privilegiati stati con transizioni uscenti che hanno probabilità simili (simile quantità di stringhe riconosciute passando per quella transizione) (ALERGIA)
    - Privilegiate coppie che minimizzano il trade-off tra la perdita di precisione (probabilità di riconoscere il linguaggio target) e la diminuzione della dimensione dell'Automa (MFI)

Applicazioni e Risultati

- Esempi di campi di applicazione con successo:
  - Speech Recognition
    - Riconoscimento della lingua parlata
    - Modellazione del Linguaggio
  - Elaborazione Digitale di Musica
    - Apprendimento e riconoscimento automatico di stili musicali
- Funzionano solo con Linguaggi Regolari
- Non applicabili per indurre Grammatiche del Linguaggio Naturale

Sommario

- Introduzione
- Grammar Induction: Teorie e Metodi
- Tecniche di apprendimento di Linguaggi Formali
- **Tecniche di apprendimento del Linguaggio Naturale**
  - Classificazione dei metodi
  - Vari Approcci
- Risultati e Conclusioni
- Riferimenti

Tecniche di Apprendimento del Linguaggio Naturale

- Inducono Grammatiche Context Free
- E' impossibile indurre CFG da un testo
  - conseguenza del risultato di Gold sul concetto di "Language identification in the limit"
- Alternativa: PAC Learning – Probably Approximately Correct Learning
  - Ci si accontenta di un'approssimazione che rispetti il criterio di precisione desiderato

Tecniche di Apprendimento del Linguaggio Naturale (2)

- Vari approcci utilizzati:
  - Algoritmi genetici [Lankhorst, 1994]
  - Reti Neurali [Honkela, 1995]
  - Minimum Descriptioni Length Principle (MDL): minimizzano la Grammatica [Wolff, 1988]
  - Metodi Euristici basati sul merging di categorie e valutazione della probabilità di successo della Grammatica indotta: Algoritmo inside-outside [Baker, 1979], [Lari & Young, 1992]

Classificazione dei metodi

- (Supervised) vs (Unsupervised)
  - I primi prevedono un intervento umano durante il processo di inferenza
    - - Generalmente operano su testi pre-tagged: richiedono lavoro aggiuntivo per annotare il testo
    - + Risultati migliori: più precisi grazie alle info aggiuntive
  - I secondi sono metodi totalmente automatizzati
    - + Generalmente operano senza informazioni aggiuntive sulla struttura del testo: meno costosi.
    - - Meno precisi: non hanno idea del linguaggio target
    - + Funzionano anche quando è impossibile annotare il testo (Es. dati non linguistici, DNA, codici cifrati)
- (Solo Positive Set) vs (Positive e Negative Set)
  - Negative Set solitamente costituito da frasi casuali

Sommario

- Introduzione
- Grammar Induction: Teorie e Metodi
- Tecniche di apprendimento di Linguaggi Formali
- Tecniche di apprendimento del Linguaggio Naturale
  - Classificazione dei metodi
  - **Vari Approcci**
    - EMILE 4.1
    - An Evolutionary Approach (Algoritmo Genetico)
- Risultati e Conclusioni
- Riferimenti

EMILE 4.1 [Adriaans, 2000]

- Basato sull'algoritmo di EMILE 3.0 sviluppato da Adriaans [Adriaans, 1992, 1999]
- Rientra nei paradigmi *PAC* (Probably Approximately Correct Learning)
- *Unsupervised*: uno dei primi algoritmi efficienti operanti su testo non annotato.
- *S+:* utilizza solo esempi positivi
- Costruisce una ***Categorial Grammar***
- Assume che il Linguaggio Naturale sia ***shallow*** ("semplice")
  - Ogni costrutto sintattico ha una frase che lo esemplifica, e la quale lunghezza è di ordine logaritmico rispetto alla complessità della Grammatica inferita.

Categorial Grammar

- Classe di grammatiche che generano tutti i linguaggi Context-Free
- Basate sull'assegnamento di *Categorie* sintattiche alle parole del lessico
  - Esempio: [Cat, NOUN]
- Una *Categoria* grammaticale è caratterizzata da:
  - *Espressioni* che appartengono alla categoria
  - *Contesti,* nei quali compaiono le suddette espressioni
    - Un contesto che appare in più Categorie, viene chiamato *Ambiguo*
- Esempio di coppia *contesto/espressione*:
  - "John (makes) tea "
  - Informalmente: L'*espressione* "makes" appare nel *contesto* "John (.) tea"
  - Regola di derivazionie formale: makes →John \ σ / tea
    - "A\σ" = Espressione σ può essere preceduta da A
    - "σ/B" = Espressione σ può essere seguita da B
- Proprietà: *Principio di Sostituzione*
  - Espressioni della stessa categoria possono essere sostituite tra di loro in tutti i contesti appartenenti alla categoria stessa.
    - Esempio: "John (likes) tea"
- Utilizzate per comodità nell'ambito della Morfologia e della Linguistica Computazionale

EMILE 4.1: Principi di funzionamento

L'induzione di EMILE si basa su due fatti:

- ***Principio di sostituzione*** delle Categorial Grammars
- In un testo sufficientemente vasto, ricorrono con frequenza le stesse combinazioni contesto/espressione (***cluster***)

EMILE 4.1: Algoritmo

- 1. Estrae tutti i possibili c*lusters* contesto/espressione dal Set di frasi
- 2. Raggruppa espressioni che appaiono nello stesso contesto
- 3. Raggruppa i contesti nei quali appaiono le stesse espressioni
- 4. Sceglie le categorie grammaticali in base al metodo *2-dimensional clustering*
- 5. Trasforma le categorie grammaticali così trovate in regole di derivazione

EMILE: Esempio *1D-clustering*

- Frasi utilizzate
  - "John makes tea", "John likes tea"
- 1. Estrae tutte le possibili combinazioni di coppie "context/expression"
- 2. Raggruppa espressioni che compaiono nello stesso contesto

	(.) makes tea	John (.) tea	John makes (.)	(.) tea	John (.)	(.)	(.) likes tea	John likes (.)
John	x						x	
makes		x						
tea			x					x
John makes				x				
makes tea					x			
John makes tea							x	
likes		x						
John likes				x				
likes tea					x			
John likes tea							x	

- Tabella context/expression per 1D-clustering

EMILE: Esempio *1D-clustering (2)*

- Clusters contesto/espressione indotti:

[{'makes', 'likes'}, 'John (.) tea']
[{'John makes', 'John likes'}, '(.) tea']
[{'makes tea', 'likes tea'}, 'John (.)']
[{'John makes tea', 'John likes tea'}, '(.)']

EMILE: Esempio *1D-clustering (3)*

- 3. Raggruppa contesti, nei quali compaiono le stesse espressioni
- Per esemplificare il passo 3, consideriamo le frasi:
  - "John makes tea"
  - "John likes tea"
  - "John makes coffee"
  - "John likes coffee"

	John (.) tea	John (.) coffee	John makes (.)	John likes (.)
makes	x	x		
likes	x	x		
tea			x	x
coffee			x	x

- Parte rilevante della Tabella context/expression per 1D-*clustering*

- Clusters contesto/espressione indotti:

[{'makes', 'likes'}, {'John (.) tea', 'John (.) coffee'}] ← Verb
[{'tea', 'coffee'}, {'John makes (.)', 'John likes (.)'}] ← Noun

- Categorie Grammaticali (Verb, Noun) indotte direttamente dai Clusters

EMILE: Problema *1D-Clustering*

- Problema: 1D-Clustering non sa gestire contesti *ambigui*
- Frasi utilizzate
  - "John makes tea", "John likes tea"
  - "John makes coffee", "John likes coffee"
  - "John is eating", "John likes eating"

	John (.) tea	John (.) coffee	John (.) eating	John makes (.)	John likes (.)	John is (.)
makes	x	x				
likes	x	x	x			
is			x			
tea				x	x	
coffee				x	x	
eating					x	x

- Tabella context/expression per *2D-clustering*

EMILE: Problema *1D-Clustering* (2)

- Intuitivamente, si possono dedurre le seguenti Categorie Grammaticali:
  - Noun-Phrases NP ('tea', 'coffe')
  - Verb-Phrases VP ('makes', 'likes')
  - 'Ing'-Phrases IP ('eating')
  - Auxiliary Verbs AV ('is')
- Clusters contesto/espressione indotti con 1D-clustering:

[{'John (.) tea', 'John (.) coffee'}, {'makes', 'likes'}]
[{'John (.) eating'}, {'likes', 'is'}]
A → [{'John makes (.)'}, {'tea', 'coffee'}]
B → [{'John likes (.)'}, {'tea', 'coffee', 'eating'}]
[{'John is (.)'}, {'eating'}]

- NB: il contesto "John likes (.)" è ambiguo!
  - Compare sia nella categoria Noun-Phrase che in quella Ing-Phrase
- I clusters "A" e "B" non possono essere raggruppati con 1D Clustering (poco naturale)

EMILE: Esempio *2D-Clustering*

- Matrice con blocchi di dimensione massima in evidenza

	John (.) tea	John (.) coffee	John (.) eating	John makes (.)	John likes (.)	John is (.)
makes	x	x				
likes	x	x	x			
is			x			
eating					x	x
tea				x	x	
coffee				x	x	

- Per l'induzione dei clusters context/expression:
  - Non è richiesto che il blocco sia totalmente contenuto nella matrice
  - E' sufficiente che una certa percentuale di esso lo sia
    - E' improbabile che S+ contenga tutte le frasi necessarie a "riempire" un blocco

EMILE: Esempio 2D-Clustering(2)

- Clusters context/expression indotti:

[{'John (.) tea', 'John (.) coffee'}, {'makes', 'likes'}]
[{'John (.) eating'}, {'likes', 'is'}]
→ [{'John makes (.)', 'John likes (.)'}, {'tea', 'coffee'}]
[{'John is (.)', 'John likes (.)'}, {'eating'}]
[{'John (.) tea', 'John (.) coffee', 'John (.) eating'}, {'likes'}]
[{'John likes (.)'}, {'eating', 'tea', 'coffee'}]

- Induzione più vicina all'intuizione umana
- Gli ultimi 2 clusters sono ridondanti:
  - Coppie context/espression già presenti nei cluster precedenti
- Alla fine del ciclo di esecuzione, EMILE rimuove questi clusters superflui.

EMILE: Determinazione delle regole di derivazione

- Per ogni espressione *e* appartenente alla categoria *T* si ha*: [T]* → *e*
- Sottoespressioni di un tipo che sono anche espressioni di un altro tipo, vengono sostituite con quest'ultimo, esempio:
  - EMILE ha già identificato la categoria
    - [T1] → dog | cat | parrot
  - EMILE identifica le seguenti espressioni come appartenenti alla categoria T2
    - {I feed my dog, I feed my cat, I feed my parrot}
  - Regola inferita
    - [T2] → I feed my [T1]

EMILE: Esempio (5)

- Determinazione di regole di derivazione ricorsive
  - Le seguenti espressioni appartengono alla categoria [S]
    - "Mary drinks tea"
    - "John observes that Mary drinks tea"
  - Regola inferita
    - [S] → John observes that [S]
  - Frase indotta
    - "John observes that John observes that Mary drinks tea"

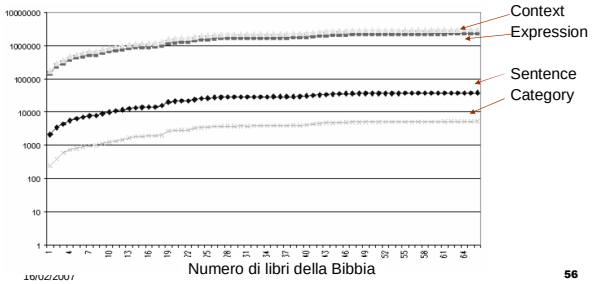
EMILE: The Medline experiments

- Esecuzione di EMILE sul dominio bio-medico, su un corpora di medie dimensioni (3000 linee di testo libero)
- Dominio troppo piccolo per convergenza
- Alcuni risultati comunque interessanti: categorie lessicali

[16] ⇒ School of Medicine, University of Washington, Seattle 98195, USA [16] ⇒ University of Kitasato Hospital, Sagamihara, Kanagawa,Japan [16] ⇒ Heinrich-Heine-University, Dusseldorf, Germany [16] ⇒ School of Medicine, Chi a University	Istituti Accademici
[94] ⇒ Chinese [94] ⇒ Japanese [94] ⇒ Polish	Lingue
[101] ⇒ 32 : Cancer Res 1996 Oct [101] ⇒ 35 : Genomics 1996 Aug [101] ⇒ 44 : Cancer Res 1995 Dec [101] ⇒ 50 : Cancer Res 1995 Fe [101] ⇒ 54 : Eur J Biochem 1994 Sep [101] ⇒ 58 : Cancer Res 1994 Mar	Riferimenti a Riviste
[105] ⇒ identified in 13 cases (72 [105] ⇒ detected in 9 of 87 informative cases (10 [105] ⇒ o served in 5 (55	Osservazioni-verbi

EMILE: The Bible experiment

- Testo vasto ed omogeneo (6MB)
- Il numero di espressioni e contesti incontrati, ad un certo punto, inizia a convergere



EMILE: conclusioni

- + L'esperimento sulla Bibbia "incrina" la congettura che apprendere il linguaggio naturale usando unicamente S+ è infattibile.
- + Teoricamente, al limite, apprende la corretta struttura grammaticale del linguaggio a partire da frasi del linguaggio stesso
- + Efficiente e scalabile
  - Valido soprattutto su Large Text Corpora
- - Vengono create più categorie grammaticali del necessario
- - Non identifica i costrutti più semplici: NP VP; solo delle variazioni di essi.
- - Se parametri sui cluster ammissibili (%di completezza del blocco, nella matrice context/expressions, richiesta) sono poco ristretti, c'è rischio di sovra-generalizzazione
- Possibili applicazioni:
  - Apprendimento di informazioni semantiche da testo
    - Dominio del Wolrd Wide Web
  - Apprendimento delle strutture morfologiche delle parole

An Evolutionary Approach (Algorimto Genetico)

- Proposto da M. Aycinena et altri. [AYCINENA]
- Evolve una grammatica per il linguaggio naturale utilizzando un algoritmo genetico
- Supervised: il testo utilizzato è annotato con part-of-speech tags (Inglese)
- Utilizza sia S+ che S-
- Genera CFG: rappresentate come stringhe.
- Valutazione dello stato di "forma" di una Grammatica con una fitness evaluation function

Evolutionary: Esempio di individuo

- Le Grammatiche sono Stringhe di non-terminali e pre-terminali (part of speech tags)
- Ogni 3 caratteri una produzione

  SABABCBCDCAE

  S→AB

  A→BC

  B→CD

  C→AE

Evolutionary: Algoritmo

- 0. La popolazione iniziale (insieme delle possibili grammatiche) è generata casualmente da una distribuzione uniforme di cromosomi di una certa lunghezza
- 1. Esegui ciclo scegli-alleva-rimpiazza
  - A. Scegli un individuo (grammatica) a caso
  - B. **Accoppialo** con il suo compagno ideale (in base alla fitness function) in modo da generare due figli
  - Rimpiazza il genitore più debole con il figlio più in forma

Evolutionary: Incrocio e Mutazione

- La procreazione dei due nuovi figli avviene nel modo seguente
  - Incrocio:
    - seleziona una produzione a caso in entrambi i genitori
    - Scegli un punto di partenza nelle produzioni (1° o 2 °)
    - Scambia le produzioni da quel punto in poi
  - Mutazione:
    - La mutazione consiste nello scambiare un simbolo non-terminale o pre-terminale con un alltro simbolo non-terminale o pre-terminale
    - Per ogni simbolo nella stringa di G, decidiamo se effettuare la mutazione

Evolutionary: Fitness Evaluation Function

- Lo stato di forma della Grammatica è
  - in relazione diretta al Numero di frasi del Positive Set correttamente riconosciute
  - in relazione inversa al Numero di frasi generate casualmente (Negative Set) riconosciute erroneamente dalla grammatica
  - Scontato esponenzialmente, se di lunghezza inferiore

Evolutionary: Experiments

- L'algoritmo è stato eseguito su testi letterari in lingua inglese
  - The Wizard of Oz, Alice in Wonderland, Tom Sawyer, Brown linguistic corpora
- Ogni testo è stato annotato con i seguenti sette pre-terminali, una frase per riga
  - Nouns, Verbs ,adJective, adveRbs, Prepositions, deTerminers, Other

Evolutionary: Positive and Negative Set

- Il Solo Positive Set porta ad una sovra-generalizzazione della Grammatica
  - S→T N | T J | N V | N J | P N | P V | P J | R J | V J | R S | N S | P S | T S | J S | V S | **S V** | **S J** | **S N**
  - Contiene regole per appendere e pre-pendere troppi simboli. Non utilizzate nel linguaggio naturale
- Il Negative Set è dato da frasi generate casualmente: alta probabilità che non appartengano al linguaggio naturale
  - S→ P J | P N | T N | T J | V J | J J | R J | N J | P S | V S | N S | J S | R S | T S | **S N**
  - Consente di appendere un unico simbolo: Noun
  - In inglese, nella maggior parte dei casi, si può appendere un nome ed avere una frase sintatticamente corretta
    - "Es: "The cat" "The black cat" "The black pussy cat"

Evolutionary: Risultati esperimenti

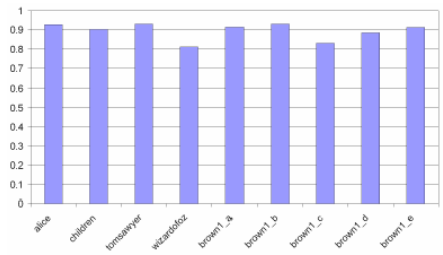
Corpus	Number of generations completed	grammar	% positive examples parsed	% negative examples parsed	fitness
aliceinwonderland	280000	0V0 0R0 0J 0V N0R J 0P N 0P J 0 0 N 0 0 V 0 V J 0P 0 0 N N 0N 0 0 T N 0 J 0 0 T J	92.50%	8.40%	657.364
brown1_a	48500	0N442N0TN0J42 0V 0P 0 3 RN 0 N 0 40N 0N 0V 0R 2 TT 0 0T 40 0T 2V 0 0 0P 40 V 0 0V 0 0R 0 0 J 0V 40 0N 0 J N	94.10%	6.10%	1667.85
brown1_b	280000	0 J J 0N J 0R 0 0N 0 0T 0 0 0N 0 J 0 0 V 0 0 V N 0P 0 0T N 0T J 0P N 0R J 0P J 0 V J	94.70%	6.70%	1227.17
brown1_c	15500	0 4 4 7 R 4 7 T T 1 V J 2 3 P 4 0 J 5 P T 2 J N 4 0 T 7 1 T 4 V 4 2 0 1 4 0 V 1 0 4 4 P T 0 3 P 7 0 8 0 0 N 0 R 7 7 J N 8 J T 4 P 1 1 T 6 5 7 5 6 N 2 4 J P 7 0 N 0 7 J 4 P 5 0 0 R 4 N N 2 V V 5 2 T 4 P N 3 4 N 6 1 5 9 4 J 0 N V 7 7 J 0 N 4 4 0 5 6 N 2 4 J N 0 R 5 2 2 N 5 5 7 4 R P 0 N T 4 T 0 0 4 P 0 N P 3 V 0 1 1 5 4 R O	80.50%	4.70%	382.998
brown1_d	45000	3 V 0 0 0 J 5 T V 0 0 V 0 R 2 3 0 N 0 N V 0 T N 1 T 3 3 N N 0 0 T 3 T V 0 0 N 0 T 3 3 N 9 0 J 3 0 0 P 0 3 0 2 0 R J 0 P 0 0 R 3 0 T 0 P 3 8 N P	86.20%	5.60%	583.586
brown1_e	122000	0 P J 0 P 0 0 V 0 0 N 0 8 T N 0 T 0 0 T J 0 V J 0 J J 0 R J 0 J 0 0 R 0 0 P N 0 0 N 0 N J	93.90%	5.90%	1782.67
children	280000	0 P J 0 V 0 0 V N 0 J J 0 N J 0 0 N 0 N 0 0 T J 0 V J 0 J 0 0 R 0 0 N J 0 P 0 0 T N 0 T 0	91.80%	5.70%	677.211
tomsawyer	280000	0 P J 0 V N 0 N N 0 0 V 0 T 0 0 T N 0 T J 0 0 N 0 0 J 0 N J 0 P 0 0 P N 0 V J 0 V 0 0 R 0 0 J 0 0 N 0 0 R J	92.70%	8.60%	2292.89
wizardofoz	280000	0 0 0 0 P J 0 V 0 0 0 R 0 0 P 0 0 R J 0 V N 0 T N 0 0 0 0 J 0 J 0 T 0 0 P N 0 N V 0 V J 0 T J 0 N 0 0 N N 0 J N	89.50%	9.20%	920.852

0→V0
0→R0
0→0J
0→VN
…

Evolutionary: risultati esperimenti

- Risultati accettabili:
  - G riconosce la maggior parte di S+
  - e una piccola parte di S-
- Col tempo, tende a ridurre la lunghezza della Grammatica
- Per valutare meglio i risultati è stata eseguita una cross-validation
  - Comportamento delle Grammatiche su testi diversi da quelli usati per la generazione.

Evolutionary: cross-validation



Il Metodo si è dimostrato efficace

Evolutionary: Conclusioni

- + Produce Grammatiche con alta precisione
- - La Grammatica indotta è molto diversa da quella costruita intuitivamente dall'uomo
- In Particolare, La G per l'inglese, accetta ongi stringa che termina con uno dei bigrammi previsti nelle produzioni, seguito da zero o più nomi
  - + Riconosce abilmente frasi valide
  - + Rifiuta la maggior parte delle frasi che non sono inglesi
  - - Non ci dice niente sulla struttura delle frasi
  - - Comunque, accetta troppe frasi che non sono valide

Sommario

- Introduzione
- Grammar Induction: Teorie e Metodi
- Tecniche di apprendimento su Linguaggi Formali
- Tecniche di apprendimento sul Linguaggio Naturale
- **Risultati e Conclusioni**
- Riferimenti

Risultati e Conclusioni

- Costruire grammatiche manualmente è un lavoro estremamente lungo ed inoltre non da garanzie di precisione sui risultati
- La Grammar Induction può essere di grande aiuto in questo processo
  - Con metodi completamente automatici (Unsupervised)
    - EMILE
    - Applicazioni a testi con impossibilità di annotazione
  - Con metodi Semi-Automatici [Chung]
    - Possibilità di dare in pasto all'algoritmo conoscenze aggiuntive
    - Possibilità di post-processare la Grammatica indotta manualmente
    - Risultati confortanti
  - Con metodi che prevedono una precedente annotazione del testo
    - Evolutionary approach
    - Risultati precisi

Risultati e Conclusioni (2)

- Applicazione utile nei processi di annotazione del testo automatizzati
  - Metodi Supervised utilizzabili in pratica
- Alcune applicazioni utili, soprattutto per i linguisti
  - Ricerca di informazioni sulla struttura sintattica delle frasi
  - Riconoscimento di categorie semantiche
- **Nessuno pretende di modellare l'acquisizione del linguaggio umano con l'induzione di CFG**
- Grammatiche Context-Sensitive per risultati più vicini al linguaggio umano
  - ADIOS model (Automatic Distillation of structure) [Solan] – metodo unsupervised sul ampi testi non-annotati

References

- [Gold, 1967] Gold, E. M. (1967) Language identification in the limit. *Information and Control*, 10(5):447--474.
- [Oncina, 1992] J. Oncina, P. García Identifying regular languages in polynomial time. *Advances in Structural and Syntactic Pattern Recognition World Scientific*, 1992 99-108
- [Lankhorst, 1994] Grammatical inference with a genetic algorithm. In *Proceedings of the 1994 EUROSIM Conference on Massively Parallel Applications and Development, pages 423-430*
- [Honkela, 1995] T. Honkela, V. Pulkki, T. Kohonen. Contextual relations of words in grimm tales, analyzed by self-organizing map. In *Proceedings of the International Conference on Artifiical Neural Networks.*

References (2)

- [Baker, 1979] J. K. Baker Trainable grammars for speech recognition. In *Speech Communication Papers for the Ninety-seventh Meeting of the Acoustical Society of America,* pages 547-550
- [Lari & Young, 1990] K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language,* 4:35-56
- [Wolff, 1988] J. G. Wolff. 1988. Learning syntax and meanings through optimization and distributional analysis. In *Categories and Processes in Language Acquisition,* pages 85-98
- [Adriaans, 2000] P. Adriaans, M. Trautwein, M. Vervoort.Towards High Speed Grammar Induction on Large Text Corpora

References (3)

- [Adriaans, 1992] P. Adriaans. Language learning from a categorial perspective. Ph.D. thesis.
- [Adriaans, 1999] P. Adriaans. Learning shallow context-free languages under simple distributions.
- [Chung] Semi-Automatic acquisition of domain-specific semantic structures
- [AYCINENA] An Evolutionary approach to natural language grammar induction
- [SOLAN] Unsupervised context-sensitive language acquisition from a large corpus