Introduzione ai metodi di apprendimento automatico in NLP/LC

Che cos'è la linguistica computazionale ?

La linguistica computazionale è un'area di ricerca interdisciplinare a cavallo dell'informatica e della linguistica (e talvolta della psicologia e dell'ingegneria) che si occupa

•sia di mettere in grado i calcolatori di « comprendere » il linguaggio umano (NLP),

•sia di studiare le proprietà formali e matematiche del linguaggio, interpretato come sistema di stringhe, quindi come un sistema matematico.

Applicazioni della Linguistica Computazionale

• L'inglese o l'italiano come lingua di interazione col calcolatore, come per la consultazione di basi dati.

• La traduzione automatica di documenti scientifici e tecnici.

• La generazione automatica di basi dati a partire da documenti tecnici o di rapporti medici a partire da dati visuali (radiografie, per esempio)

• Ricerca automatica di documenti, in basi di testi o sul web

• Filtri, per spam o messaggi elettronici (UNIGE rifiuta 80% dei messaggi ricevuti).

• Riassunto o abstracting automatico, di uno o più documenti.

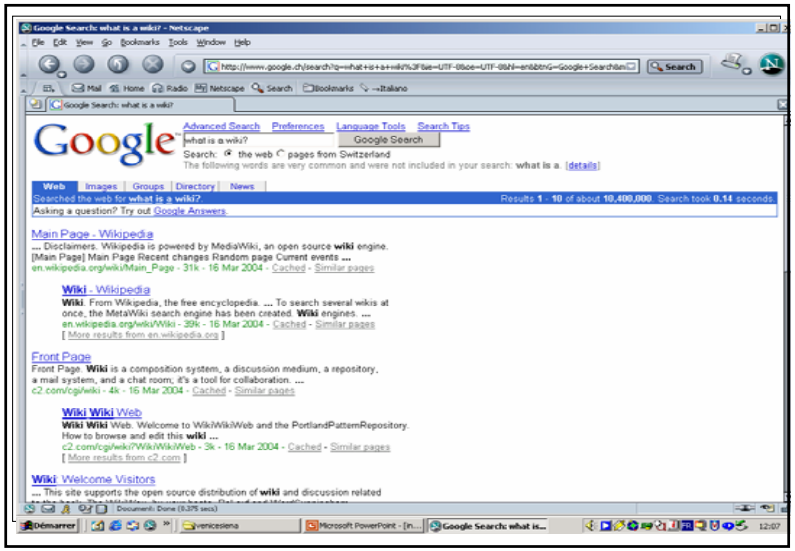
Applicazioni LC: Microsoft Longhorn

Question answering: da dichiarazioni rilasciate a MIT TR.

I motori di ricerca attuali rispondono a eventuali domande con una enumerazione esplicita dei documenti in cui è discusso il tema della domanda.

Ma a volte questo non basta.

Se volessi sapere che cos'é un wiki, non lo capirei da una domanda in Google.



Applicazioni LC: Microsoft Longhorn

La nuova version di Windows Longhorn in via di sviluppo conterrà una forma di apprendimento automatico che si chiama « transformation-based learning » (Brill) che è in grado di ricostruire forme analoghe alla domanda e a cercarle nel web.

Da *what is a wiki* si construisce

A wiki is

_ *is a wiki* etc

Che renderanno più probabile il trovare la risposta, si spera

Applicazioni NLP: Clustering for text mining

•Text Mining is the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources.

•One of the most active application area for text mining is in the biosciences: one of the big current questions in genomics is which proteins interact with which other proteins.

•Text is represented as a « bag of words ». More sophisticated NLP could help.

Applicazioni NLP: Clustering for text mining

To get farther though we need more sophisticated language analysis. A number of us are working on statistical techniques that try to assign semantics, or meaning, to parts of the text. We break off pieces of the problem of analysis, targetted towards particular applications, rather than trying to "read" the articles as a whole. This goal is especially promising in the biosciences due to the nature of the text itself. In some ways it is easier to process automatically than ordinary text. It is less ambiguous and the processes it describes are somewhat mechanical, and so representable in a computer.

The fundamental limitations of text mining are first, that we will not be able to write programs that fully interpret text for a very long time, and second, that the information one needs is often not recorded in textual form.

Applicazioni NLP: Clustering for text mining

I tre problemi principali in questo tipo di applicazioni sono

-La rappresentazione adeguata dei documenti

-La gestione della alta dimensionalità (migliaia di dimensioni)

-**Il raggruppamento dei documenti simili e la loro visualizzazione**

-Per esempio

-Raggruppamento di 13mila documenti della Reuters in gruppi di contenuto per una più facile classificazione o verifica di una classificazione esistente

Applicazioni NLP: Clustering for text mining

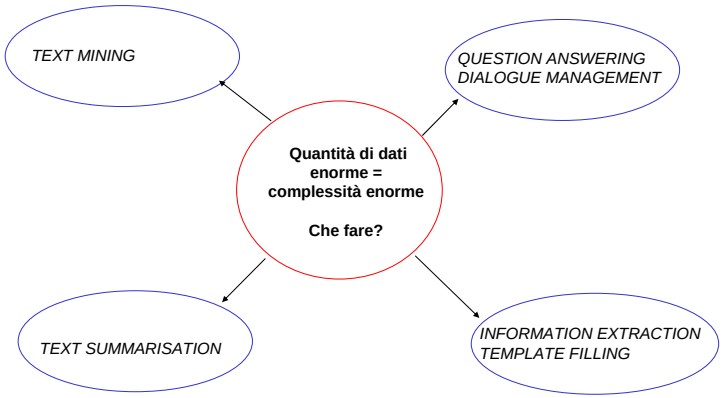
L'algoritmo per il raggruppamento: le carte auto-organizzatrici

L'algoritmo per il raggruppamento: le carte auto-organizzatrici

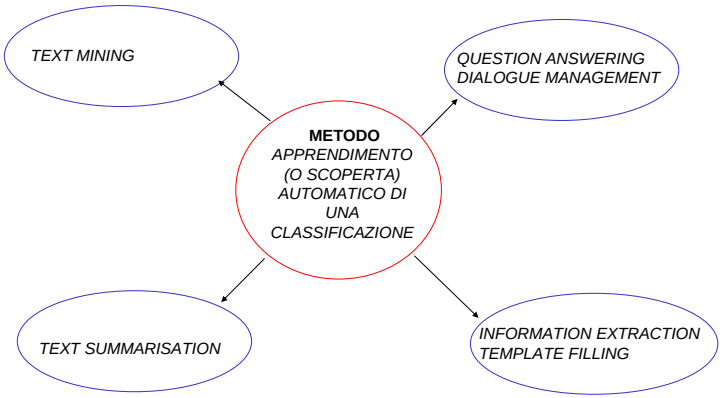
I documenti Reuters rappresentati come carte di Kohonen

I documenti Reuters rappresentati come carte di Kohonen

Che metodo per gestire terabytes di informazione?



L'apprendimento automatico



I principali approcci in linguistica computazionale

Approccio a base di regole

La rappresentazione della conoscenza del dominio è esplicita,
espressa sotto forma di regole. Essa corrisponde alle conoscenze di un esperto della materia.

Ho cose più importanti di cui occuparmi

F → IBAR COMPC
COMPC → SN
SN → N SA F2
…

[F [IBAR Ho IBAR] [COMPC [SN cose [SA piu importanti SA] [F2 [SPD di cui [SV2 occuparmi SV2] SPD] F2] SN] COMPC] F]

Problema approccio basato su regole

• E' molto difficile concepire tutto il sistema di regole necessario a fornire a un calcolatore le conoscenze linguistiche per l'elaborazione del linguaggio.

• E' anche molto difficile gestire la complessità e le interazioni del sistema di regole.

• Soluzione

Invece di un esperto che fornisce al calcolatore le informazioni linguistiche sotto forma di regole, l'esperto annota un testo con informazione linguistica e il programma impara da solo le regole e il loro uso.

I principali approcci in linguistica computazionale (2)

Approccio a base di corpora

La rappresentazione della conoscenza del dominio è implicita,
espressa sotto forma di annotazione di un testo o di un corpus..

[F [IBAR Ho IBAR] [COMPC [SN cose [SA piu importanti SA] [F2 [SPD di cui [SV2 occuparmi SV2] SPD] F2] SN] COMPC] F]

Programma impara automaticamente le regole e la loro frequenza d'uso nel testo

L'approccio a base di corpora

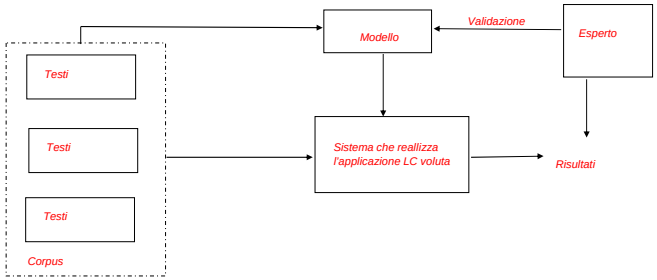
E' molto difficile e anche costoso costruire risorse linguistiche rappresentative in quantità sufficiente.

Non si cerca più di riprodurre la competenza linguistica con modelli che formalizzano le nostre facoltà di comprensione linguistica, ma si cerca di riprodurre, per una classe di applicazioni data,

la performance linguistica associata

Questo, lo si fa con **modelli automaticamente estratti** dai dati, che devono essere in grandi quantità e caratteristici della applicazione voluta.

L'approccio a base di corpora



La validazione dei modelli così ottenuti non è basata sulla loro capacità esplicativa del funzionamento del linguaggio, ma riposa sulla valutazione del **miglioramento della performance** per l'applicazione data.

Previous Criticism

Against corpora -- data available through corpora do not constitute the explicandum of linguistic theory because

• they are performance data, they do not distinguish grammatical from ungrammatical;

• they are skewed (i.e. they do not constitute a good sample)

Against statistics

•grammaticality cannot be defined in terms of probability

•probabilistic models are too complex, hence unlearnable

Previous Criticism: reply

•a grammar (algebraically defined) is needed also for corpus-based approaches

•corpora are mostly annotated

•grammaticality can be defined in terms of probability (in the limit)

Vantaggi

•Acquisizione: identificazione e codifica automatica delle conoscenze necessarie

•Copertura: si coprono automaticamente tutti i fenomeni linguistici nel dominio di applicazione data.

•Robustezza: adattamento più facile al rumore e ai dati imprevisti

•Portabilità: facilità a sviluppare una nuova lingua.

•Valutazione: si arriva a una valutazione sperimentale dei sistemi pratici e delle ipotesi scientifiche

Diversi metodi empirici

Rappresentazioni: probabiliste, simboliche, reti neurali

Addestramento: con supervisione/esempi

o senza supervisione/esempi

Compiti: Riconoscimento della voce;

analisi sintattica (parsing)/disambiguazione;

acquisizione lessicale: sotto-categorizzazione, struttura argomentale,

Disambiguazione del senso delle parole;

traduzione automatica…

Apprendimento: definizione

Definizione

Un programma *apprende* a partire da un'esperimento di addestramento A per eseguire il compito C valutato da una misura di performance P, se la performance P al compito C migliora in seguito all'esposizione ad A.

Esempio

Compito C: classificare i verbi in classi predefinite

Esperimento di addestramento A: base dati di coppie di verbi con i loro attributi e le risposte corrette

Misura di performance P : % di nuovi verbi classificati correttamente (rispetto a una classificazione stabilita da un esperto)

Apprendimento per classificazione

Il compito più studiato in apprendimento automatico (machine learning) consiste nell' inferire una funzione che assegna gli esempi rappresentati come vettori di tratti distintivi ad una classe fra un insieme finito di classi date.

Classificazione: esempio

Sia un insieme di verbi.

Compito: classificazione binaria: verbi di movimento (correre, saltare, passeggiare) e verbi di cambiamento di stato (fondere, cuocere).

Proprietà: per ogni volta che troviamo il verbo nel corpus

è transitivo?

è passivo?

Il suo soggetto è animato?

Apprendimento per classificazione: esempio

Esempio	Trans?	Pass?	Anim?	Classe
correre	5%	3%	90%	MoM
saltare	55%	5%	77%	MoM
fondere	10%	9%	20%	CoS
cuocere	80%	69%	88%	CoS

Se Pass? < 9% e Anim? >20% allora MoM altrimenti CoS

Come classifichiamo il nuovo esempio?
passeggiare T=2% P=1% A=98%

Alberi di decisione

Gli alberi di decisione sono dei classificatori che operano su degli esempi rappresentati da vettori di tratti.

I nodi testano i tratti.

Si forma un arco per ogni tratto.

Le foglie specificano le categorie.

Si possono anche scrivere sotto forma di regole simboliche.

Alberi di decisione

Idea di base: determinare l'attributo che classifica i dati nel modo migliore, fra tutti gli attributi. Si usa questo attributo come radice dell'albero. Si ripete (ricorsivamente) il processo ad ogni arco.

Enunciato informale dell'algoritmo

Si determini l'attributo che divide l'insieme di esempi nel modo migliore (da determinare).

Si usi questo attributo come radice e si creino tanti rami discendenti quanti ci sono valori per l'attributo radice.

Per ognuno dei rami, si ripeta il processo con il sotto-insieme degli esempi classificati da questo ramo

Algoritmo di base dell'induzione di alberi di decisione

AlberoD(*esempi,attributi*)

Se tutti gli esempi appartengono ad una categoria allora

risultato: una foglia con questa categoria come etichetta

Altrimenti se *attributi* è vuoto allora

risultato: una foglia con l'etichetta data dalla categoria maggioritaria in *esempi*

Altrimenti scegliere un attributo A per la radice:

per tutti i valori possibili *vi* di A

sia *esempi-i* il sotto-insieme di cui il valore di A è *vi*

aggiungere un arco alla radice per il test *A = vi*

se *esempi-i* è vuoto allora

creare una foglia di cui l'etichetta è la categoria maggioritaria in *esempi*

altrimenti chiamare ricorsivamente AlberoD(*esempi-i,attributi – {A}*)

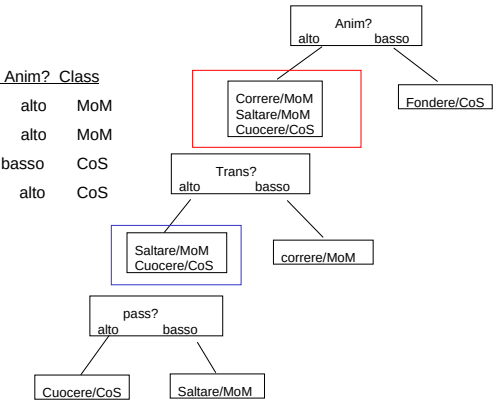
Esempio

Costruire l'albero di decisione a partire dai dati seguenti.

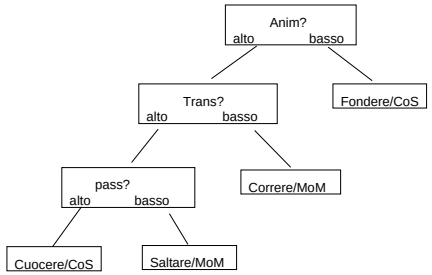
Esempio	Trans?	Pass?	Anim?	Class
correre	basso	basso	alto	MoM
saltare	alto	basso	alto	MoM
fondere	basso	basso	basso	CoS
cuocere	alto	alto	alto	CoS

Esempio

Esempio	Trans?	Pass?	Anim?	Class
correre	basso	basso	alto	MoM
saltare	alto	basso	alto	MoM
fondere	basso	basso	basso	CoS
cuocere	alto	alto	alto	CoS



Esempio



Come scegliere la radice

In generale, ci si pone come obbiettivo di costruire l'albero con il minor numero di nodi (rasoio di Occam)

Abbiamo quindi bisogno di un test che separi i sotto-insiemi puri rispetto a una classe data, poichè in questo modo sono più vicini ad essere delle foglie

L'**entropia** è la misura che indica l'impurità di un insieme di esempi rispetto a una classificazione, quindi le misure utilizzate per scegliere l'attributo radice sono basate sull'entropia

Entropia

L'entropia è la misura che indica il disordine di un insieme di esempi rispetto a una classificazione

L'entropia di un insieme di esempi S rispetto a una classificazione C

$$Entropia\ (S)=\sum_{i=1}^{c}-p_i\log_2 p_i$$

Dove *pi* è la proporzione d'esempi di categoria *i* in S

Esempio

Data la formula per l'entropia calcolare l'entropia dei dati qui sotto

$$Entropia\ (S)=\sum_{i=1}^{c}-p_i\log_2 p_i$$

Esempio	Trans?	Pass?	Anim?	Class
correre	basso	basso	alto	MoM
saltare	alto	basso	alto	MoM
fondere	basso	basso	basso	CoS
cuocere	alto	alto	alto	CoS

Esempio

$$Entropia\ (S)=\sum_{i=1}^{c}-p_i\log_2 p_i$$

Esempio	Trans?	Pass?	Anim?	Class
correre	basso	basso	alto	MoM
saltare	alto	basso	alto	MoM
fondere	basso	basso	basso	CoS
cuocere	alto	alto	alto	CoS

Entropia(Trans?) = -(.5 log2 .5) -(.5 log2 .5) = 1

Entropia(Pass?) = -(.25 log2 .25) -(.75 log2 .75) = 0.5 +0.31 = 0.81

Entropia(Anim?) = -(.25 log2 .25) –(.75 log2 .75) = 0.5 +0.31 = 0.81

Per una classificazione data, la distribuzione uniforme ha entropia massima

Guadagno in informazione (information gain)

Il *guadagno in informazione* di un'attributo è la riduzione attesa dell'entropia se si facesse la partizione in base a questo attributo.

Esempio	Classe
correre	MoM
saltare	MoM
fondere	CoS
cuocere	CoS

Exemple	Trans?
correre	basso
saltare	alto
fondere	basso
cuocere	alto

Exemple	Anim?
correre	alto
saltare	alto
fondere	basso
cuocere	alto

Entropia(Classe) = -(.5 log2 .5) -(.5 log2 .5) = 1

Entropia(Trans?) = -(.5 log2 .5) -(.5 log2 .5) = 1 GI= E(Classe)-E(Trans?)=0

Entropia(Anim?) = -(.25 log2 .25) –(.75 log2 .75) = 0.81 GI= E(Class)-E(Anim?)= 0.19

Misure di performance

Supponiamo di avere 20 verbi, di cui 10 verbi appartengono alla classe E e 10 alla classe O.

Il nostro algoritmo di apprendimento automatico ci dice invece che 9 verbi appartengono alla classe E e 11 alla classe O.

Quali sono gli errori possibili?

Verbi che in realtà sono E ma che sono stati classificati come O.

Verbi che in realtà sono O ma che sono stati classificati come E.

Misure di performance

Verbo	Verità	Algoritmo
Floated	E	O
Hiked	E	E
Hurried	E	O
Jumped	E	E
Leaped	E	E
Marched	E	E
Paraded	E	E
Raced	E	E
Rushed	E	O
Skipped	E	E

Verbo	Verità	Algoritmo
Borrowed	O	O
Carved	O	O
Inherited	O	O
Kicked	O	E
Knitted	O	O
Organised	O	O
Painted	O	O
Played	O	O
Typed	O	E
Washed	O	O
Yelled	O	O

		Algoritmo		
		E	**O**	**Totale**
Verità	**E**	7	3	10
	O	2	8	10
	Totale	9	11	20

Misure di performance

		Algoritmo		
		E	**O**	**Totale**
Verità	**E**	7	3	10
	O	2	8	10
	Totale	9	11	20

Precisione E = 7 / 9 O = 8/11

Copertura E = 7/10 O = 8/10

Esattezza E+O = 7+8/20

Il bias induttivo (inductive bias)

Qualsiasi metodo utilizzato da un sistema di classificazione per scegliere tra due funzioni entrambe compatibili con i dati di apprendimento si chiama *bias induttivo*.

Il bias induttivo è di due tipi

- il *bias di linguaggio*– il linguaggio che rappresenta le definizioni delle ipotesi di apprendimento definisce uno spazio di ipotesi limitato
- il *bias di ricerca* – l'espressività del linguaggio è sufficiente a esprimere tutte le ipotesi possibili, ma l'algoritmo di ricerca preferisce certe ipotesi ad altre

Il bias degli alberi di decisione è di preferire gli alberi più piccoli a quelli più grandi (bias di ricerca)

La futilità dell'apprendimento in assenza di bias

Un apprendista che non utilizza alcuna ipotesi a priori sull'identità del concetto obiettivo non possiede alcuna base razionale per classificare nuovi esempi.

L'apprendimento senza bias è impossibile.

Il biais induttivo descrive la logica con cui l'apprendista generalizza al di là dei dati di apprendimento.

Il rasoio di Occam

Pluralitas non est ponenda sine necessitate

Perchè dare la preferenza alle ipotesi più corte? Perchè ce ne sono di meno.

Ma allora perchè non preferire le ipotesi più specifiche che sono anche poco numerose?

Riassunto della lezione

- Il trattamento del linguaggio cerca di far comprendere il linguaggio ai calcolatori, le sue applicazioni sono le interfacce, la traduzione automatica, la ricerca di documenti,…
- Approccio a base di corpora e statistica
- Risorse linguistiche in grande quantità
- Acquisizione automatica di conoscenze linguistiche
- Importanza della performance e della valutazione sistematica

- Metodo di apprendimento simbolico: alberi di decisione

  Robusti, adatti a molti problemi di classificazione linguistica