Elaborazione del linguaggio naturale

4 luglio 2008

La **linguistica computazionale** è il luogo di incontro di linguistica teorica e applicata con le tecnologie informatiche. Studia i problemi teorici e applicativi relativi al linguaggio e al suo uso nell'informatica. Lo sviluppo è attivo in due direzioni, cioè nella realizzazione di:

- strumenti informatici per lo studio e la ricerca sul linguaggio (e sulla mente)
- applicazioni informatiche di largo uso (correttori ortografici, information retrieval) che sfruttano competenze linguistiche

Per **linguaggio naturale** si intendono tutte le lingue parlate correntemente dagli esseri umani. Il termine *naturale* nasce come contrapposizione a **linguaggio formale**, inteso come linguaggio artificiale completamente formalizzato e, possibilmente, privo di ambiguità, di cui viene fatto largo uso in informatica (ad esempio, tutti i vari linguaggi di programmazione sono linguaggi formali).

1 Cenni storici

La storia dell'elaborazione del linguaggio naturale si suddivide tra diverse discipline, prendendo diversi nomi:

- Linguistica computazionale (linguistica)
- Elaborazione del linguaggio naturale (informatica)
- Riconoscimento del parlato (elettronica)
- Neurolinguistica (psicologia)

1.1 1940-1950 - II guerra mondiale

- Automi a stati finiti; teoria dei linguaggi formali (algebra e teoria degli insiemi per la formalizzazione dei linguaggi). Chomsky, Backus, Naur (grammatiche BNF per la formalizzazione dei linguaggi).
- Algoritmi probabilistici per il riconoscimento del parlato. Sviluppo della teoria dell'informazione (Shannon), che non riguarda la forma o il contenuto, ma si basa sulle procedure di trasmissione e ricevimento: rumorosità del canale, codifica e decodifica, entropia di un linguaggio
- La Machine Translation è una delle prime applicazioni desiderate (soprattutto a scopi militari, durante la Guerra Fredda)

1.2 1957-1970 - Due paradigmi

- **Simbolico:** *studia il linguaggio naturale tramite regole e grammatiche.* Uso *forte* delle regole.

  Teoria dei linguaggi formali (algoritmi di parsing top-down e bottom up). Intelligenza artificiale (teorie logiche; sviluppo di sistemi domanda-risposta tramite pattern matching, ricerca di keyword e semplici euristiche).
- **Stocastico:** parte da documenti reali e li analizza per estrarre le "regole" probabilistiche che li regolano. Uso *debole* delle regole, che possono essere trasgredite.

  Metodo bayesiano, tramite uso di dizionari e corpora. Riconoscimento OCR.

1.3 1970-1983 - Quattro paradigmi

- **Stocastico:** sviluppo algoritmi di riconoscimento (HMM)
- **Logic-based:** unificazione delle strutture in feature (riconosce le interconnessioni tra le parti del discorso, analizzabili in modo più potente che non dalle grammatiche context-free.
- **Natural language understanding:** Winograd (SHRDLU), parsing ben compreso, primi lavori seri su semantica e discorso
- **Discourse modelling:** analisi delle sottostrutture del discorso, risoluzione automatica dei riferimenti

1.4 1983-1993 - Empiricismo e FSModels

In questo periodo ci sono pochi finanziamenti per il campo dell'elaborazione del linguaggio naturale. L'uso della rappresentazione denotativa aveva ostacolato tutta l'intelligenza artificiale e i pochi risultati avevano fatto perdere fiducia ai finanziatori.

La logica è troppo limitata rispetto alla realtà: rende impossibile risolvere alcuni problemi. Sono quindi necessarie altre rappresentazioni.

L'uso delle grammatiche è troppo lento per fornire buoni risultati: scrivere una grammatica completa ed efficace per una lingua richiede anni, ma la lingua si evolve molto più in fretta.

- Ritorno all'utilizzo dei modelli a stati finiti, per la fonologia, la morfologia e la sintassi
- Ritorno all'empiricismo: lavori di IBM per il riconoscimento del parlato basandosi su modelli probabilistici; approcci data-driven (ossia più incentrati su dati preesistenti che non su un modello) per il POS tagging, il parsing e l'annotazione, per la risoluzione delle ambiguità.
- *Natural Language Generation.*

1.5 1994-oggi - I due campi si incontrano

L'approccio simbolico e quello stocastico si stanno riunendo.

Le difficoltà incontrate con il primo trovano nuove vie di soluzione. Si uniscono ad un pesante uso delle metodologie data-driven e dei modelli probabilistici.

Nascono nuovi ambiti applicativi, come il Web, e nuove possibilità dovute all'aumentata capacità di elaborazione dei sistemi.

Si sono raggiunti risultati positivi nel parsing, nei modelli di interazione, nella morfologia e nell'uso di dizionari e corpora.

2 L'uomo, la macchina e il linguaggio

Il linguaggio è la caratteristica e il processo cognitivo che distingue l'uomo dalle altre specie. Taluni ritengono anche che possa essere utilizzato come indicatore del livello mentale di una persona (erroneamente, in quanto non sempre è detto che ci sia connessione diretta tra le due cose)

In **campo informatico** si cerca di raggiungere la capacità di **comprendere**, **tradurre** e **generare** il linguaggio con diverse finalità (traduzione, controllo, ...). Il compito è **difficile**, a causa dell'elevata ambiguità del linguaggio naturale, e ha portato quindi alla nascita di un gran numero di modelli per tentare di affrontare il problema.

Quando si utilizza il linguaggio naturale in ambito informatico, bisogna fare un'ulteriore distinzione tra lingua scritta e parlata, in quanto gli elaboratori, per poter utilizzare il parlato, devono prima digitalizzarlo, introducendo ulteriore ambiguità dovuta all'imperfezione del processo di trasformazione (la stessa osservazione si applica ai testi scritti digitalizzati tramite procedimenti di OCR, Optical Character Recognition).

2.1 Modelli

Tra i vari modelli utilizzati per l'analisi del linguaggio naturale, possiamo ricordare:

- Macchine a stati finiti
  - Deterministiche e non deterministiche
  - FSA (Finite State Automata) e FST (Finite State Translators)
  - Automi pesati

  - Modelli Markoviani e Modelli Markoviani Nascosti (HMM, Hidden Markov Models)

- Sistemi a regole formali
  - Grammatiche per linguaggi regolari, CFG, Feature Augmented Grammars
- Logica
  - Logica del primo ordine (FOL, First Order Logic), calcolo dei predicati
- Teoria della probabilità
  - Macchine a stati finiti, Sistemi a regole e sistemi basati sulla logica possono essere arricchiti con probabilità (PCFG, Probabilistic Context Free Grammars)
- Tecniche di ricerca nello spazio degli stati e programmazione dinamica
  - Applicabili a sistemi a stati finiti e a sistemi a regole

3 Parsing

I primi parser nascono all'inizio degli anni '70.

Si sviluppano da studi di Chomsky sulla struttura delle frasi e dai dibattiti sulla psicolinguistica (quanto siamo predisposti alle lingue? Quanto impariamo?).

Negli anni '80 vengono sviluppate nuove strategie, più efficaci, in grado di trattare anche input mal formati e risoluzione dei riferimenti. Nascono le grammatiche con attributi che permettono di associare parsing sintattico e interpretazione semantica. Il mapping diretto delle frasi in una struttura semantica è considerato uno standard.

4 Morfologia e dizionari

I primi programmi di elaborazione del linguaggio naturale, fino agli anni '80, usavano dizionari di migliaia di parole per effettuare il POS tagging in fase di parsing. I primi prototipi, tuttavia, erano poco potenti per scarsità di vocabolario.

Si studiarono poi i *morphology based look-up systems*, cioè sistemi basati su un insieme di dizionari di segmenti lessicali (basi, prefissi, suffissi, desinenze) e su un insieme di regole per la formazione delle parole.

I primi approcci erano troppo semplicistici perchè sviluppati per l'inglese, che è una lingua poco flessa. L'approccio standard utilizzato attualmente viene dalla Finlandia ed è stato sviluppato da Koskenniemi. Sfrutta gli stessi principi elencati prima, ma include una parte fonemica particolarmente sofisticata e in grado di trattare aspetti morfo-fonemici complessi.

4.1 Definizioni

Morfema è la più piccola unità significativa di un linguaggio. Una parola, o anche una sua parte (radice, suffisso, ecc)

infissi aggiunta posta in mezzo ad una parola per alterarne il significato: **hingi** -> h**um**i**ngi** (esempio dal filippino)

circonfissi una parte precede la parola e una parte segue: **sag**en -> ge**sag**t (esempio dal tedesco)

Inflessioni Modificano parte della parola alterandone alcune caratteristiche. Possono essere **regolari** o **irregolari**.

- Plurale/singolare
- Genitivo sassone
- Terza persona dei verbi inglesi

Declinazioni Sono trasformazioni di una parola sorgente per ottenere una nuova parola appartenenete ad una diversa classe grammaticale, ottenute con l'aggiunta di prefissi, suffissi, ecc...

- nome → verbo
- nome → avverbio
- nome → aggettivo

- aggettivo → nome
- verbo → aggettivo

Il linguaggio è molto flessibile. Avere le regole di flessione e declinazione ci permette di trattare parole nuove (es: fax) e immediatamente derivarne aggettivi, verbi, ecc. Ciò non sarebbe possibile se avessimo solo un lessico più grande comprendente tutte le forme possibili.

4.2 I corpora

Un *corpus* è un **insieme di testi riguardanti un certo argomento**, sufficientemente **rappresentativi** delle varie sottoaree di interesse dell'argomento stesso. La rappresentatività è uno degli indici della bontà di un corpus. La **dimensione** è un altro indice: se contiene fino a 1M parole, è di livello medio-basso. Fino a 100M è di buon livello. Intorno ai 500M di parole è di ottimo livello.

L'uso dei corpora risale alla linguistica computazionale e alla machine translation.

I corpora hanno il loro uso principale negli **approcci statistici**. Hanno portato alla realizzazione di metodi per l'inferenza delle feature linguistiche dai corpora tramite tecniche di apprendimento automatico (HMM, metodi statistici e reti neurali).

All'atto della generazione, è necessario affrontare i **problemi legati ai *raw data***, come ad esempio la presenza della punteggiatura, degli spazi e degli errori: è necessario ripulire i dati tramite pre-processing prima di utilizzarli. Si procede all'eliminazione di aspetti legati alla **formattazione** che non interessano. Si applicano euristiche per studiare la presenza di maiuscole e minuscole (sono titoli? inizi di frasi? sigle? l'identificazione è difficile!), gli apostrofi (elisione o genitivo sassone), i trattini (parole composte? divisione in sillabe? raggruppamento di sottoporzioni delle frasi?).

Si devono considerare anche **omografie** (una parola, più significati), **segmentazione** (più formati per una sola informazione: numeri di telefono, date), ***ditto tag*** (parole/espressioni sempre uguali a se stesse: "in spite of", "in order to", "because of").

I testi all'interno di un corpus **sono annotati** da esperti tramite linguaggi di markup e tagging (esistono vari standard per fare ciò). Si va dalla marcatura delle semplici strutture di base, fino alla marcatura sintattica completa (ad es, UPenn Treebank, banca di alberi sintattici della University of Pennsylvania). Di solito si usa un POS tagging utilizzando un tag set standard (Brown, C5, Upenn, ...), applicato a mano o con tecniche automatiche.

Lo **studio delle collocazioni** (vedere 6) **e della frequenza** (anche solo dei bigrammi più frequenti) porta a raccogliere molte informazioni statistiche utili per la generazione del linguaggio. Ad esempio, seppur grammaticalmente corrette, "strong computer" e "powerful tea" non hanno significato, mentre lo hanno "strong tea" a "powerful computer". Tuttavia **le parole componenti una forma idiomatica possono comparire non adiacenti**. Si analizza quindi il corpus con finestrature di dimensione variabile e si calcolano media e varianza che caratterizzano la distribuzione della distanza tra le parole del corpus. Per evitare di individuare delle relazioni casuali presenti nel corpus, si può quindi utilizzare il test t di Student o il test χ^2 per decidere se è possibile rifiutare l'ipotesi nulla di esistenza di una reale forma idiomatica. Si possono inoltre utilizzare il rapporto di verosimiglianza delle ipotesi di dipendenza e indipendenza oppure i rapporti di frequenze relative di più termini per individuare la probabilità che i termini siano correlati. Un esempio di tali misure, è la **mutua informazione:** $I(x,y) = log_2 \frac{P(x,y)}{P(x)P(y)} = log_2 \frac{P(x|y)}{P(x)}$**,** che è una buona misura dell'indipendenza, ma non della dipendenza, in quanto in caso di dipendenza il punteggio ottenuto dipende dalla frequenza delle parole componenti l'espressione.

Una delle prime applicazioni dell'individuazione di forme idiomatiche, è la possibilità di tradurle in lingue straniere.

4.3 Parsing morfologico

Data una parola, si occupa di risalire alla sua struttura e alle sue porzioni (identificandole tramite tag) e di comprendere il ruolo che queste hanno.

> es: cats → cat, nome, plurale

Per poter costruire un parser morfologico dobbiamo avere:

- lessico
- morfotassi (ordine dei morfemi, modellabile con un FSA)
- regole ortografiche (modalità di combinazione dei morfemi)

Un riconoscitore morfologico molto semplice può essere costruito tramite una FSA che riconosce i termini lettera per lettera, fino a giungere allo stato finale.

4.3.1 Parsing morfologico a due livelli

È il metodo più usato per effettuare il parsing morfologico (inventato da Koskenniemi). Usa due traduttori a stati finiti (che se applicati in serie sono equivalenti ad un unico traduttore). Lavora sul **livello lessicale** (lemma + tag contenenti le informazioni lessicali) e su quello **superficiale** (parola completa, come la leggiamo) del linguaggio, passando attraverso un **livello intermedio** (che comprende la parola già declinata, a meno della correttezza ortografica. Ad es, fox^s invece di foxes, con "^" che rappresenta il **marcatore di fine morfema**.

È in grado di trattare **regole** per i plurali irregolari. Utilizza un marcatore di fine morfema per poter poi applicare altre regole, come ad esempio quelle che gestiscono i plurali inglesi in -es.

Il passaggio tra il livello intermedio e quello lessicale avviene tramite **analisi lessicale** e sfrutta la presenza di un lessico. Il passaggio tra il livello intermedio e quello superficiale sfrutta l'applicazione delle **regole ortografiche** ed è effettuato tramite un insieme di traduttori a stati finiti che applicano le regole.

Questo è l'approccio usato da PC-KIMMO, che usa EngLex come descrizione della morfologia della lingua inglese.

4.3.2 Stemming

È un procedimento di analisi lessicale alternativo al parsing morfologico a due livelli, utilizzabile **solo per trovare la radice** (*stem*) di una parola, eliminando prefissi, suffissi, ecc.

La radice può essere utilizzata come classe di equivalenza per più parole (volo, volare, ecc) ed è quindi utile per l'information retrival.

Gli algoritmi di stemming sono più semplici e non fanno utilizzo di un lessico online.

Il più famoso algoritmo di stemming è l'**algoritmo di Porter**, che usa regole di riscrittura in cascata, implementabili con FST.

Non potendosi appoggiare ad un lessico, è più frequente la presenza di errori: estrazione di una radice errata (organization → organ), oppure incapacità di trovare una radice ("European" dovrebbe restituire "Europe", ma non lo fa).

5 Aree di ricerca e applicazioni

- **Sistemi di dialogo** (prenotazioni automatizzate, ecc)
- **Multilinguismo**

  Soprattutto su Internet. Possibilità di effetture query in linguaggio naturale invece che query di tipo database
- **Classificazione e recupero dei documenti**

  WordNet è molto sfruttato per l'analisi dei testi
- **Acquisizione e apprendimento**

  Sviluppo di tecniche di apprendimento non supervisionato per catturare i cambiamenti della lingua senza doverli codificare a mano all'interno dei tool, a partire da corpora di addestramento

6 Problematiche dell'elaborazione del linguaggio

- Omografie e omofonie
- Polirematiche (concetti unici espressi da più parole. Es: "Giovedì Santo").
- Fonetica

  Le lingue hanno pochi fonemi (suoni che compongono le parole) che formano moltissimi morfemi (parole) che possiamo collegare in modo infinito
- Estensibilià dei significati

  Es: navigare (nel mare) -> (sul web)
- Sinonimi

  Sono usati in modo completamente intercambiabile in generale, ma per alcuni significati specifici possono essere diversi: es: "Il *vocabolario* dell'autore è molto forbito". *Dizionario* e *vocabolario* sono in generale sinonimi, ma non in questa frase.

- Collocazioni (forme idiomatiche)

  Coppie/gruppi di parole usate spesso vicine. ES: "*Obliterare* il *biglietto*", "*Compilare* il *modulo*".

  Corrispondono ad un modo convenzionale di dire qualcosa. Non sempre il significato è completamente deducibile dal significato delle componenti (non è basato sulla pura composizionalità.

  Inizialmente erano state trascurate dalle analisi linguistiche strutturali, ma sono importanti per il parsing e l'analisi, la generazione e la lessicografia.

- Ambiguità sintattica
- Ambiguità su elementi:
  - anaforici: che si riferiscono a elementi già introdotti nel testo. Es: pronomi
  - cataforici: riferiti a elementi che saranno introdotti in seguito nel testo.
- Metalinguisticità

  La lingua può parlare della lingua.

7 La parola

La parola è **l'unità fondamentale del linguaggio** (comprensione e produzione). Non sempre è facile da individuare all'interno del linguaggio parlato (per un PC, o per una persona nel caso di lingue straniere).

È un'**unità di persistenza** di una lingua (è abbastanza stabile nell'evoluzione della lingua) ma anche unità **di mutazione** (può dare origine a nuove parole).

Le parole possono essere di **classe aperta**, se ne possono essere create di nuove facilmente (nomi, aggettivi, verbi) o di **classe chiusa** (pronomi, articoli).

Le parole possono poi essere connesse in strutture sintattiche, anche molto diverse fra loro (es: la sintassi di un libro di filosofia è molto diversa da quella di un manuale tecnico).

7.1 Elaborazione delle parole

Quando si lavora con le parole è possibile **ricercarle** (riconoscerle) e sostituirle, **verificarne la correttezza** (ad esempio, a partire dalle informazioni di tagging), **generarle**, **contarle** e **predirle**.

Per fare ciò sono disponibili diverse risorse: **corpora, lessici, dizionari**.

7.1.1 Espressioni e linguaggi regolari

Permettono di definire delle stringhe di caratteri.

Sono state usate, al fine di eseguire ricerca e sostituzione, per realizzare ELIZA, un bot che simula dialoghi di ipotetiche sedute di psicoterapia Rogeriana.

8 Errori di ortografia e predizione

Possiamo ottenere un testo contenente errori a partire da diverse fonti (OCR, riconoscimento di scrittura a mano, digitazione errata, ecc). Taluni errori, come l'inversione di due caratteri durante la digitazione da tastiera, sono abbastanza frequenti.

Possiamo quindi sfruttare calcoli sulla probabilità di una data sequenza di parole e sulla predizione della prossima parola per sviluppare strumenti di rilevamento e correzione degli errori.

I modelli probabilistici si basano sulla **regola di Bayes** e sul modello di **rumorosità del canale**.

8.1 Modello bayesiano

Il problema della correzione di errori ortografici può essere trattato come il **mapping** di una stringa di simboli (quella potenzialmente errata, uscita da un canale rumoroso) in un altra (quella corretta).

Il modello Bayesiano ci dice che la parola che stimiamo ($\hat{w}$) è quella più probabile data l'osservazione che abbiamo fatto (O) e date le parole w contenute nel vocabolario V a nostra disposizione:

$$\hat{w} = \arg\max_{w \in V} P(w|O)$$

Non siamo in grado di conoscere $P(w|O)$ direttamente (è in pratica la soluzione del nostro problema).
Tuttavia, per la regola di Bayes, sappiamo che

$$P(w|O) = \frac{P(O|w)P(w)}{P(O)}$$

quindi, sostituendo ed eliminando il denominatore (ininfluente nel calcolo del massimo, perchè lo sappiamo essere sempre costante)

$$\hat{w} = \arg\max_{w \in V} \frac{P(O|w)P(w)}{P(O)} = \arg\max_{w \in V} P(O|w)P(w)$$

dove $P(w)$ è la probabilità di trovare la parola nel corpus (vocabolario) e $P(O|w)$ è la probabilità di trovare l'osservazione quando so che la parola è w (ricavata da appositi corpora di errori), entrambe grandezze note.
Nell'applicazione del metodo Bayesiano ai correttori ortografici, O è la parola digitata, e w è la parola corretta presente nel corpus.
In realtà, $P(O|w)$ non è nota, ma è stimabile, secondo due approcci:

- a partire dai dati annotati, si creano delle matrici di confusione, in funzione del numero di cancellazioni (xy è stato scritto come y), inserimenti (x è stato scritto come xy), sostituzioni (x è stato scritto come y) e trasposizioni (xy è stato scritto come yx) di lettere all'interno della parola
- la matrice di confusione viene inizializzata con valori uguali (si considera ogni carattere sostituibile, scambiabile, ecc, con qualunque altro carattere). Poi l'algoritmo di correzione viene applicato su un insieme di errori ortografici. Le matrici di confusione vengono ricalcolate sfruttando l'accoppiamento tra forme battute e forme corrette (tramite l'algoritmo EM, Expectation Maximization).

Tutto ciò è valido nell'**ipotesi semplificativa di un solo errore per parola**.
Rimuovendo tale ipotesi, **consideriamo invece la distanza tra stringhe** come parametro. Definiamo quindi ***Minimum Edit Distance*** tra due stringhe il numero minimo di operazioni di editing necessarie per trasformare l'una nell'altra. Per calcolare tale valore è possibile assegnare pesi uguali alle varie operazioni (Del, Ins, Sub, Trans), oppure pesi diversi. Si può inoltre assegnare una probabilità ad ogni operazione sfruttando le matrici di confusione calcolate in precedenza, passando così da *Minimum Edit Distance* a *Maximum Probability Alignment*, che può essere usato per **stimare la similitudine Bayesiana di una parola** contenente più di un errore **con una possibile candidata alla correzione**.

8.2 N-grammi

Gli n-grammi sono modelli che **usano N-1 parole per predire la successiva**.
La predizione serve per vari scopi, come il riconoscimento del parlato, dei manoscritti, per la correzione degli errori ortografici o per la comunicazione aumentativa (linguaggi di tipo iconico usati per migliorare le capacità comunicative, ad esempio in presenza di disabilità. Si basano su disegni che rappresentano le parole, che vengono affiancati per produrre frasi. Non sono ideogrammi).
Il calcolo della probabilità di una sequenza di parole risulta utile anche per il POS tagging, la Word Sense Disambiguation e il parsing probabilistico.
Per poter attuare la predizione, sfruttano i conteggi eseguiti su un corpus, compresa l'analisi della punteggiatura.
La probabilità dell'n-esima parola di una sequenza $P(w_1, w_2, \dots w_{n-1}, w_n)$, se consideriamo che ogni parola compaia al suo posto come un evento indipendente, è calcolabile come

$$P(w_1^n) = P(w_1)P(w_2|w_1^1)P(w_3|w_1^2)\cdots P(w_n|w_1^{n-1}) = \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

dove ogni $P(w_k|w_1^{k-1})$ è la probabilità che la parola w_k sia preceduta di $k-1$ posizioni da w_1.
Questo porterebbe a dover conoscere le probabilità di precedenza anche a distanze molto elevate. Per ovviare al problema, si considerano solo le probabilità dei bigrammi (ossia la probabilità di avere una parola conoscendo la precedente), tramite l'approssimazione $P(w_k|w_1^{k-1}) \approx P(w_k|w_{k-1})$, ottenendo quindi:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$$

Si riesce ad effettuare il calcolo ricavando dal corpus i valori necessari. Per ottenere una probabilità compresa tra 0 e 1, si contano tutti i bigrammi del tipo che ci interessa, dividendoli per la somma di tutti i bigrammi che condividono la stessa prima parola: $P(w_k|w_{k-1}) = \frac{\#(w_{n-1}w_n)}{\sum_w \#(w_{n-1}w)}$.

Smoothing e discounting Per ovviare al problema delle parole non presenti nel corpus o presenti con frequenze molto basse, si possono attuare diverse strategie.

Add-one smoothing aggiungere 1 ad ogni elemento della matrice dei conteggi dei bigrammi prima di normalizzare. In questo modo nessun elemento sarà mai a zero. Tuttavia il risultato è abbastanza impreciso

Witten-Bell e Good-Turing discounting usiamo i conteggi relativi a ciò che si è riscontrato una sola volta (quindi la probabilità più bassa che abbiamo rilevato) come una stima di tutto ciò che non si è mai incontrato nel corpus.

Backoff Se non abbiamo nel corpus esempi di un particolare trigramma ($w_{n-2}w_{n-1}w_n$) che ci aiutino a calcolare $P(w_n|w_{n-1}w_{n-2})$, usiamo la probabilità del bigramma $P(w_n|w_{n-1})$ o, in sua assenza, quella dell'unigramma $P(w_n)$.

Training set e test set Il corpus deve essere progettato con attenzione. Se contiene troppo poche parole, rischia di rigenerare esattamente i testi che sono stati immessi, invece che nuovi testi con lo stesso stile. Sono necessari milioni di parole per avere un buon corpus.

È necessario avere, oltre al set usato per l'addestramento, un altro set utilizzato per verificare la qualità del corpus.

9 Grammatica e parsing

Il linguaggio naturale può essere in parte formalizzato utilizzando delle *Context-Free Grammars*, ad esempio distinguendo, all'interno delle frasi, una parte nominale (NP), una parte preposizionale (PP, che specifica meglio il soggetto introdotto dalla parte nominale) e una parte verbale (VP). Una frase può essere inoltre composta da più parti coordinate e subordinate.

Le grammatiche di questo tipo possono essere analizzate tramite il **parsing con l'algoritmo di Early**. Tuttavia, è proibitivo costruire una grammatica CFG completa per una lingua. Si usano quindi altri approcci, come le grammatiche CFG probabilistiche (**PCFG**), che tuttavia sono abbastanza inefficienti, e le **Lexicalized PCFG**, che introducono la specializzazione in funzione delle categorie di parole o anche di singole parole specifiche.

Le regole delle grammatiche possono essere utilizzate sia per l'analisi (accettare/rifiutare stringhe, associare alberi che descrivono la composizione di stringhe) che per la generazione di stringhe appartenenti al linguaggio.

Una **derivazione** è una sequenza di regole (che formano l'albero sintattico) applicate ad una stringa, tale che copre tutti e soli gli elementi della stringa.

All'interno di una grammatica, il **lessico** è quella parte che trasforma i simboli nonterminali in terminali (es: Pronoun→me | I | you | it | ...).

La **ricorsione**, cioè la presenza di un nonterminale sia nella parte sinistra sia nella parte destra di una regola, permette la costruzione di frasi interessanti, che vanno a specificare un numero di dettagli sempre maggiore a seconda di quanto richiesto. Chomsky ha provato che un linguaggio Context Free può essere generato da un FSA se e solo se la sua CFG non contiene regole con ricorsione del tipo $A \rightarrow \alpha A \beta$, cioè con un nonterminale all'interno, ma solo con nonterminale sinistro o destro.

9.1 La sintassi

La sintassi, dal greco *syntaxis*, composizione, si riferisce al **modo in cui le parole venogono poste insieme**, e alla **relazione** esistente **tra esse**. Scopo della sintassi è modellare la conoscenza che le persone posseggono in merito alla lingua madre.

La sintassi viene definita dalla grammatica. In particolare parliamo di **grammatica prescrittiva** (che definisce come le persone dovrebbero parlare) e di **grammatica descrittiva** (che descrive come le persone effettivamente parlano.

La sintassi presenta alcune **idee chiave**:

constituency (Costitutività / composizionalità) può essere descritta tramite le **grammatiche CFG**, anche espresse in forma BNF (Backus-Naur Form), che catturano la **composizione** (come le parole vengono raggruppate in unità e come i diversi tipi di unità si comportano) e l'**ordinamento** (quali sono le regole che governano l'ordinamento delle parole e la costituzione di macro unità del linguaggio) delle unità costitutive del linguaggio.
Ad esempio, nella semplice grammatica: S→NP VP, NP e VP sono unità costitutive.
Tutte le parole che compongono, all'interno di una frase, un'unita costitutiva, non possono essere separate senza variare significato. Possono tuttavia essere spostate, in gruppo, da un punto all'altro della frase senza problemi (es: Parto il 17 settembre = Il 17 settembre parto).
Un esempio di unità costitutiva sono le Noun Phrases (NP). Per tutte le unità costitutive di un certo tipo, deve essere possibile fare delle affermazioni generali che riguardano ognuna di esse (es: Le "Noun Phrases" possono comparire prima del verbo).

Per l'inglese, altre unità costituive sono S (sentence), Verb Phrases (VP) e Prepositional Phrases (PP). Ogni unità costitutiva può essere a sua volta suddivisa in categorie:

$NP \rightarrow Pronoun$ you

$NP \rightarrow$ Proper-Noun John

$NP \rightarrow Det\,Noun$ the president

$NP \rightarrow Nominal$

$Nominal \rightarrow Noun\,Noun$ morning flight

relazioni grammaticali Possibili esempi di frasi, corrispondenti a diverse relazioni grammaticali tra unità costitutive:

Dichiarative $S \rightarrow NP\,VP$

Imperative $S \rightarrow VP$

Domande sì/no $S \rightarrow Aux\,NP\,VP$

Domande WH $S \rightarrow WH\,Aux\,NP\,VP$

Alcune grammatiche vengono estese con il concetto di **testa**. La testa è la parte dominante all'interno della sua sezione di frase. Ad esempio, è il nome in una NP.

sottocategorizzazione esprime i vincoli che una unità impone sul numero e sul tipo sintattico degli argomenti che si possono associare. Ad esempio, in una VP, invece di avere solo V, si può specificare IntransV, piuttosto che TransV.

Le moderne grammatiche distinguono fino a 100 sottocategorizzazioni per i verbi.

L'uso delle sottocategorizzazioni introduce un notevole vantaggio con un piccolo overhead. La sottocategorizzazione aiuterà l'analisi semantica, che chiede di determinare: <**chi**> fece <**cosa**> <**a chi**> <**in quale evento**>.

dipendenze lessicali Tra le varie parole di una frase esistono delle dipendenze da rispettare, come ad esempio la **concordanza** di genere e numero. Un possibile modo di gestirle con grammatiche CFG consiste nell'estendere la grammatica con nonterminali appositi, che considerino non solo il ruolo delle unità costituenti all'interno della frase, ma anche il genere e il numero dell'unità. A causa dell'altro numero di combinazioni possibili, tuttavia, ciò porta ad un notevole aumento nel numero delle regole.

movement / long-distance dependency dipendenza tra parole lontane nella frase. Le parti proposizionali $PP \rightarrow Preposition\,NP$ (es: from Boston), spesso generano ambiguità, perchè non è ben chiaro a dove debbano essere attaccate.

9.2 Parsing

Il parsing è il processo che consiste nel considerare una stringa e una grammatica e costruire uno (o più) albero(/i) sintattico(/i) corretto(/i) per quella stringa.

Successivamente bisogna scegliere l'albero giusto tra tutti quelli possibili.

Un **albero corretto** è un albero che copre tutti e soli gli elementi dell'ingresso ed ha S al vertice.

Il Penn Treebank è un corpus comprendente un gran numero di frasi già sottoposte a parsing, assieme ai relativi alberi sintattici.

Il più efficiente algoritmo di parsing è l'**algoritmo di Early**, che è in grado di effettuare il parsing anche risolvendo il problema della ricorsione sinistra senza dover alterare la grammatica o limitare artificialmente la ricerca. L'algoritmo lavora espandendo contemporaneamente tutti le possibili produzioni che sono alla base della frase che si sta generando.

Un'altro algoritmo di parsing interessante per l'elaborazione del linguaggio naturale è il parsing parziale (***shallow parsing***), che permette di identificare gruppi di parole connesse tra loro.

9.2.1 Probabilistic Context Free Grammars (PCFG)

Per **eliminare le ambiguità nel parsing** si ricorre all'uso di **metodi probabilistici**, aumentando le grammatiche con indicazioni relative alla probabilità (ogni produzione contiene la **regola** della grammatica **e** la **probabilità che** la regola **sia applicata**) e modificando i parser in modo che vengano considerate solo le configurazioni più probabili, restituendo infine solo la più probabile in assoluto. Le probabilità si ricavano da un database annotato (treebank).

L'insieme di regole che espandono lo **stesso simbolo non terminale** ha **somma** delle probabilità **uguale a 1**.

La **probabilità di una sequenza di parole** (frase) è la probabilità del suo albero nel caso di assenza di ambiguità, o la somma delle probabilità degli alberi in caso di ambiguità.

Per effettuare un parsing probabilistico sono necessari:

- una grammatica
- un ampio dizionario con POS tagging annotato con le probabilità
- un parser

Esempio di ambiguità risolvibile probabilisticamente: "Can you book TWA flights?". La probabilità permette di stabilire se stiamo intendendo "Puoi prenotare i voli per TWA?" (come se fosse una persona, o una ditta) oppure "Puoi prenotare i voli della TWA?" (come se fosse una compagnia aerea).

La **probabilità di un albero** è il prodotto della probabilità delle regole legate alla derivazione (assunzione dell'indipendenza dell'espansione dei vari nonterminali). In realtà, esistono dipendenze strutturali e lessicali che confutano l'indipendenza, e permetterebbero di capire se la regola applicata è quella a probabilità minore. Ma non vengono considerate dalle PCFG.

Per risolvere tali problemi, sono state introdotte le grammatiche **Probabilistic Lexicalized CFG**, che prendono in considerazione il comportamento particolare di talune classi di parole o anche singole parole. In queste grammatiche, ogni nonterminale dell'albero di parsing viene annotato con la sua **testa lessicale** (tramite arricchimento delle regole). In base a questo attributo, si modifica il comportamento della grammatica ove necessario.

9.2.2 Feature based grammars

Permettono di associare vincoli alle categorie grammaticali (es: se so che un NP è femminile plurale, dovrò associarlo ad una parola che sia femminile plurale).

Forma una gerarchia di feature (connesse tramite sussunzione) con ereditarietà degli attributi dal figlio al padre.

Le strutture di feature sono rappresentabili tramite una matrice o con un grafo con archi orientati. Ogni feature è associata ad un valore. Esempi di feature sono la categoria (NP, VP, ...) il genere, il numero, la persona (1a, 2a, 3a).

Le feature possono essere rappresentati con **attributi annidati**. Ad esempio, la feature HEAD (testa dell'unità) può avere come valore una richiesta di concordanza (AGREEMENT, espressa a sua volta da un'insieme di feature) di numero e persona con un'altra unità.

L'**unificazione** è un modo per **integrare la conoscenza** espressa in diversi vincoli: date due strutture di attributi compatibili essa produce la struttura più generale che contiene tutte le informazioni dell'input. Al contempo, **controlla i vincoli**, infatti fallisce se le due strutture date sono incompatibili.

L'uso dell'unificazione e delle strutture di feature rende possibile una trattazione elegante dei vincoli che sarebbe complessa utilizzando le sole CFG.

Le **regole grammaticali vengono arricchite con strutture di attributi** (*feature*), che possono rappresentare delle caratteristiche della regola o dei vincoli sulla sua applicazione. In tal modo si riescono ad associare strutture complesse sia a elementi del lessico sia a intere categorie grammaticali.

Esiste un campo apposito, *SUBCAT*, per indicare la **sottocategorizzazione** dei verbi (transitivo, intransitivo, ecc). Nel caso di verbi molto particolari, la sottocategorizzazione può corrispondere col lessico, in modo da indicare esattamente il comportamento. L'ideale sarebbe riuscire a sottocategorizzare **quanto basta** per avere nuove informazioni senza appesantire eccessivamente la computazione.

Per risolvere le dipendenze richieste ai fini della comprensione corretta di una sottocategorizzazione si tiene una lista (*gap-list*) di elementi che andranno a risolvere le sottocategorizzazioni di verbi successivi.

Per trattare le feature based grammar si può continuare a usare l'**algoritmo di Early**, debitamente **esteso per supportare gli attributi**.

10 Part-Of-Speech tagging

Il POS tagging è l'unico tipo di analisi utilizzato dalla linguistica computazionale, al contrario di ciò che si fa tradizionalmente (analisi grammaticale, analisi logica, ecc).

Per POS tagging si intende il **processo di assegnazione** di POS tag o marcatori di classi lessicali ad ogni parola del corpus.

Il tagging può essere **basato su regole** oppure **statistico**. Esiste anche il tagging **TBL** che combina le idee degli altri due.

Il POS tagging aiuta a costruire l'albero ottenuto dal parsing della frase, con poco costo. Nell'elaborazione del linguaggio naturale, capita spesso di avere algoritmi molto buoni a basso costo, ma algoritmi molto pesanti per raggiungere l'ottimo. Di solito si usano più approcci, per ottenere il meglio di entrambi.

Serve a **disambiguare parole graficamente uguali ma con significati e pronunce diverse**. Serve anche per l'information retrieval: conoscere la classe lessicale di una parola ci può dare delle informazioni utili sul suo ruolo nella frase.

Le varie Part-Of-Speech sono anche dette categorie lessicali, tag lessicali, classi di parole, classi morfologiche.

Tradizionalmente, sin dai tempi antichi, esistono 8 Part-Of-Speech: **Nome** (N), **Verbo** (V), **Aggettivo** (ADJ), **Preposizione** (P), **Avverbio** (ADV), **Articolo** (DET: da *Determiner*), **Pronome** (PRO), **Congiunzione** (*conjunction*). Oggi in realtà si usano molte più di 8 classi. Esistono infatti diversi **insiemi di tag**. La scelta di un insieme di tag è molto importante ai fini di un buon risultato. Si può prediligere la semplicità (e quindi poche richieste computazionali ma minor precisione) oppure la granularità (più tag e più precisione, ma più peso di elaborazione). Esempi di set di tag sono C5, C7 o il UPenn TreeBank tagset.

Tutti i tagset comprendono le 8 classi base, più altro per rappresentare casi particolari, come le WH-question, oppure la particella "to" (che essendo molto particolare ha una classe dedicata).

Ad ogni parola, spesso, può corrispondere più di un tag POS, a seconda del contesto in cui viene usata. Il problema che il POS tagging deve affrontare è **determinare quale POS tag associare ad una istanza di parola**.

Tagging di parole sconosciute Le nuove parole nascono molto rapidamente: più di 20 al mese (prendendo un giornale come riferimento).

Per trattarle con il POS tagging sì può:

1. assumere che **siano nomi**
2. assumere che abbiano una **distribuzione di probabilità simile a quella delle parole che compaiono una sola volta** nell'insieme di training
3. usare **informazioni morfologiche** (ad es., se finiscono in -ed, saranno verbi).

10.1 Tagging basato su regole - ENGTWOL

Funziona consultando un dizionario. Ad ogni parola della frase che è contenuta nel dizionario, assegna (tramite un Finite State Translator) **tutti i possibili tag**. Sfrutta poi delle **regole scritte a mano**, che consentano di **rimuovere** alcuni **tag in modo selettivo** (applicazione dei vincoli). Alla fine, si rimane con il tagging corretto per ogni parola.

10.2 Tagging statistico

È basato sulla teoria della probabilità.

Sfrutta la probabilità condizionata per avere una stima di quanto spesso ogni parola assume le sue varie forme, contandone le occorrenze nel corpus: $P(V|''race'') = \frac{Count(''race''\ is\ verb)}{total\ Count(''race'')}$.

L'algoritmo dei tag frequenti:

1. Addestramento: per ogni parola:
   (a) Crea un dizionario con tutti i possibili tag per ogni parola
   (b) Prendi un corpus annotato
   (c) Conta il numero di volte che ogni tag è presente per la parola
2. Data una nuova frase:
   (a) Per ogni parola, considera il tag più frequente per quella parola nel corpus

Tuttavia questo approccio non dà buone prestazioni, perchè i corpora sono molto grandi, e il conteggio richiede tempo. Inoltre talune parole potrebbero non essere mai presenti nel corpus, e quindi i loro tag sarebbero sconosciuti. L'accuratezza può arrivare al 90%, ma con altri algoritmi si può fare di meglio.

10.2.1 HMM POS Tagging

Per risolvere i problemi causati dal conteggio, si preferisce usare un tagger basato sugli Hidden Markov Models. Un tagger di questo tipo sceglie una **sequenze di tag per un'intera frase** invece che per una sola parola: $\hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n|w_1^n) = \arg\max_{t_1^n} P(w_1^n|t_1^n)P(t_1^n)$

Per fare ciò dobbiamo usare un Modello di Markov Nascosto perchè ciò che vogliamo fare è predire la sequenza di etichette (che sono "nascoste") conoscendo però le parole. Non stiamo predicendo una parola conoscendo le precedenti, bensì un'etichetta, dovendo però basarci su altre etichette a loro volta predette.

Inoltre, i modelli di **HMM ottimizzano l'intera sequenza**, non la singola scelta. La scelta dell'etichetta è fatta in modo tale da non osservare solo la predizione in corso, ma l'assegnazione della sequenza di etichette che meglio descrive l'intera frase.

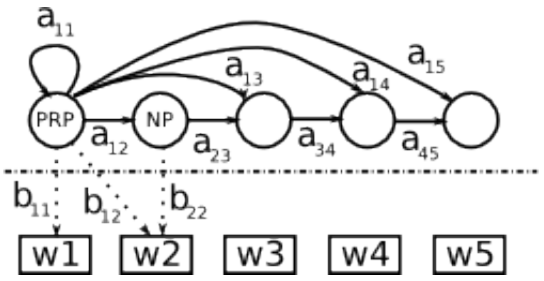
Utilizziamo due **ipotesi semplificative:**

- la **probabilità di una parola dipende solo dal proprio POS tag**, indipendentemente dagli altri POS tag e dalle altre parole che la circondano: $P(w_1^n|t_1^n) \approx \prod_{i=1}^{n} P(w_i|t_i)$
- la probabilità di un tag dipende solo da quella del tag che lo precede (**bigrammi**): $P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1})$

Da questo e dalla formula precedente ricaviamo che:

$\hat{t}_1^n = \arg\max_{t_1^n} \prod P(w_i|t_i)P(t_i|t_{i-1})$

che contiene la probabilità che ha un tag di seguire un'altro tag e la probabilità che una certa parola w_i sia quella associabile al tag t_i.



I dati necessari per svolgere i calcoli sono contenuti in due matrici: una, A, indica le probabilità di transizione tra i tag, l'altra, B, indica la probabilità che ogni tag corrisponda ad una certa parola. Tali matrici sono calcolabili a partire da un corpus annotato, tramite conteggio.

Considerazioni sugli HMM Una catena markoviana è un caso particolare di automa a stati finiti pesato (cioè in cui ad ogni transizione è associata la probabilità che tale transizione si realizzi), in cui solo la sequenza di ingresso determina la sequenza degli stati attrversati. Una catena markoviana è utile per assegnare probabilità a sequenze non ambigue.

Nel caso del POS tagging **non possiamo utilizzare una catena markoviana**, perchè non osserviamo la sequenza di tag. Consideriamo quindi come eventi osservati le parole, come eventi nascosti i POS tag (che pensiamo come fattori probabilistici del nostro modello). Utilizziamo poi l'**algoritmo di Viterbi** (che procede passo per passo, ma ottimizza l'intera sequenza) come algoritmo di decoding per determinare la sequenza di variabili nascoste a fronte di una sequenza di osservazioni in HMM.

Estensioni Per aumentare le potenzialità dell'analisi con gli HMM è possibile aumentare la dimensione della finestra, usando **trigrammi** invece di bigrammi. Ciò potrebbe tuttavia provocare problemi, perchè ampliando la finestra aumenta la possibilità che non esista nel corpus la sequenza di n parole che cerchiamo.

10.3 Tagging basato su trasformazioni (TBL - Brill Tagging)

È una **combinazione delle metodologie di tagging stocastico e** di quello **basato su regole**.

Utilizza regole per specificare i tag in determinati contesti, e utilizza le tecniche stocastiche di machine learning, utilizzando un corpus annotato e un dizionario con i tag più frequenti come input.

L'**idea di fondo** è:

1. Assegna il tag più probabile ad ogni parola come valore iniziale
2. Modifica i tag utilizzando regole definite a mano ed eseguite con un ordine specifico (le regole producono errori, che vengono corretti da quelle eseguite in seguito).

Ogni regola è composta da due parti: un ambiente di triggering e una regola di riscrittura.

Le regole vengono apprese con il seguente algoritmo:

1. Etichetta ogni parola con il tag più probabile (dal dizionario)
2. Considera ogni possibile trasformazione e seleziona quella che migliora il tagging

3. Ri-applica i tag al corpus applicando le regole
4. Ripeti 2-3 fino a quando è soddisfatto un criterio di fermata (es: X% di correttezza rispetto al corpus di training)

Le regole così ottenute sono compatte, e possono essere controllate da esseri umani.

Problemi Le prime 100 regole producono 96.8% di accuratezza. Le prime 200 producono il 97%. È **difficile innalzare il livello di accuratezza**.

Inoltre, il tagger TBL è più **lento** di quello HMM, anche se **funziona meglio con parole sconosciute**.

10.4 Valutazione di un tagger

Per testare il buon funzionamento dell'algoritmo abbiamo bisogno di un secondo corpus etichettato a mano che funga da test-set. La percentuale di correttezza è calcolabile come il numero di parole taggate correttamente nel test set fratto il numero totale di parole che il test set contiene.

Per evitare di dover usare due set diversi, spesso si usa il 90% di un corpus come training set e il restante 10% come test set.

Lo stesso approccio, ma senza il training set (che non è necessario) può essere applicato ad un tagger basato su regole.

11 Semantica

La semantica è lo **studio del significato delle occorrenze linguistiche**. Punta alla rappresentazione formale del significato delle componenti linguistiche e di come esse si combinano nel significato delle frasi.

Nell'ambito dell'ELN, esistono appositi algoritmi per il mapping tra componenti e significati, utili per applicazioni come *query answer* e *information retrieval*.

L'analisi semantica mira alla definizione della relazione tra costrutto linguistico (frase) e mondo esterno a cui si riferisce.

Una rappresentazione semantica ha diverse **proprietà**:

- **Verificabilità** rispetto allo stato del mondo come è rappresentato nella base di conoscenza
- È una rappresentazione non **ambigua**
- Permette **Word Sense Tagging** (Word Sense Disambiguation)
- Permette di fare **inferenza** sulle conoscenze (tramite l'uso di variabili).

La conoscenza semantica può essere espressa tramite **calcolo di predicati del primo ordine** (FOPC), che gode delle proprietà sopra elencate e presenta alcuni concetti rilevanti dal punto di vista linguistico: categorie, eventi, tempo e punto di riferimento, rappresentazione delle credenze (beliefs), operatori e logica modale.

11.1 Analisi semantica guidata dalla sintassi (*syntax-driven*)

Segue un approccio basato sul **principio di composizionalità**: il significato di una frase è ricavabile dall'insieme dei significati delle sottoparti (NB: non solo dal significato delle singole parole, ma anche dal loro ordine, dal raggruppamento e dalle relazioni che intercorrono tra esse). Funziona **arricchendo l'albero sintattico con annotazioni semantiche** (predicati in FOPC che ne rappresentano il significato), che potranno essere utilizzate nei passi successivi anche per disambiguare (assieme alle conoscenze di contesto). Non effettua direttamente la disambiguazione, perchè non è compito dell'analizzatore semantico.

Le annotazioni semantiche **possono essere specializzate a livello lessicale**, per gestire le particolarità di taluni termini, o i loro differenti significati, ove individuabili a partire dalla sintassi.

A volte le annotazioni vengono scritte utilizzando la ***lambda notation***, determinando una maggiore flessibilità (ad esempio, per inserire espressioni provenienti da sottoalberi): la scrittura $\lambda x P(x)(A)$ indica che bisogna sostituire tutte le x che compaiono in $P(x)$ con delle A, ottenendo $P(A)$. Si possono anche avere più lambda-sostituzioni in un'unica espressione, che verranno effettuate da sinistra a destra.

Ad esempio, nel caso $\lambda x \lambda y Near(x, y)$, se si trova scritto $\lambda x \lambda y Near(x, y)(casa)$ si ottiene $\lambda y Near(casa, y)$, e quindi, successivamente, $\lambda y Near(casa, y)(scuola)$ porta a $Near(casa, scuola)$.

Punti da trattare con attenzione sono l'ambito di visibilità (*scope*) dei quantificatori e la presenza di forme idiomatiche, difficili da rilevare.

Il problema dell'approccio syntax-driven e delle *semantic grammar* (grammatiche arricchite con attributi semantici) è che **le categorie della grammatica sono diverse da quelle della semantica**. Gli elementi chiave per la comprensione semantica

spesso risultano sparsi sull'albero sintattico, e gli alberi includono *costituenti* utili per l'analisi sintattica ma inutili per quella semantica, perchè troppo generali (NP, VP, ...).

Combinare la sintassi e la semantica di un dominio in **un'unica rappresentazione non consente il riuso della grammatica**, che risulta specifica per il dominio.

L'**estrazione di informazioni** è più semplice dell'analisi semantica completa, perchè la conoscenza che si desidera estrarre può essere descritta da template relativamente semplici e predefiniti. Tutto il resto può essere ignorato.

12 Semantica lessicale

Il *significato* è stato studiato da tre prospettive:

- il significato delle **singole parole**
- come i significati delle singole parole si combinano, portando al significato delle **frasi**
- come i significati delle frasi si combinano portando al significato di un **testo** o di un **discorso**.

La **semantica lessicale** (*lexical semantics*) **è lo studio del significato delle singole parole**.

Quando si analizza il significato delle parole bisogna considerare **omonimia**, **polisemia**, **sinonimia**, ... **ruoli tematici**.

Le risorse informatiche per la semantica lessicale sono *WordNet* e gli algoritmi di *Word Sense Disambiguation*.

Qualche definizione:

lessema una voce del lessico, cioè una coppia composta da una voce e un solo significato

lessico una collezione di lessemi

omonimia lessemi che condividono la forma *fonologica*, *ortografica* o entrambe, ma hanno significati distinti e non collegati. Possono essere cioè *omografi* (bank (banca) vs. bank (argine)) o *omofoni* (*write* e *right*, oppure *piece* e *peace*). L'omonimia può portare a **problemi** nei campi del **text-to-speech**, in quanto non è facile identificare la corretta pronuncia da adottare, o dell'**information retrieval** nel caso di omografi, nonchè della **traduzione automatica** o del **riconoscimento del parlato**.

polisemia un singolo lessema con più significati *legati* fra loro ("bank" può essere sia l'edificio, sia l'istituzione finanziaria). Molte parole frequenti hanno più di un significato (sorprattutto i verbi): il numero di significati è direttamente correlato alla frequenza.

Per accorgersi se una parola ha più di un significato si può usare lo **zeugma test**, che consiste nel riunire due frasi contenenti tale parola in una sola (connessa dalla parola stessa) e vedere se ciò che si ottiene ha un senso. Se non ce l'ha, la parola ha più significati.

metafora e metonimia tipi speciali di polisemia. Metafora: "Germany will *pull* Slovenia out of its economic slump". Metonimia: "The *White House announced* yesterday that..."

sinonimia parole diverse che hanno lo stesso significato. In realtà la corrispondenza non è mai esatta (perchè dovrebbero esistere?) ma hanno qualche piccola variante in casi specifici (es: big/large: "my big sister" vs "my large sister").

antonimia parole che sono opposte rispetto ad un aspetto del loro significato

iponomia e ipernomia il significato di un lessema è un sottoinsieme del significato di un altro. Cane è un'*iponomia* di canide. Canide è un'*ipernomia* di cane.

ruoli tematici sono generalizzazioni semantiche sui ruoli propri dei verbi. Ad esempio, *-er* indica il ruolo tematico di "agente dell'azione" (AGENT). Altri ruoli tematici sono: experiencer, force, theme, result, content, instrument, beneficiary, source, goal. A partire dai ruoli tematici, possiamo fare delle inferenze (*selectional restrictions)* sulle parole che sono presenti nella frase. Ad esempio, se abbiamo il verbo *eat*, il suo agente deve essere qualcosa in grado di mangiare. Tali inferenze sono troppo costose se basate sul calcolo dei predicati del primo ordine. Usiamo quindi gli iponimi di WordNet per codificare le selectional restrictions.

Particolarmente difficile da comprendere con i ruoli tematici è la **semantica profonda**, cioè il significato delle frasi ricche di figure retoriche e definizioni "poetiche", perchè in tali frasi le Selection Restrictions vengono violate sistematicamente e intenzionalmente.

È utile poter effettuare un calcolo della similitudine tra due parole. Può servire as esempio per la machine translation, oppure nel campo dell'information retrieval per realizzare l'espansione della query (cioè per riformulare la query con parole dal significato più generico o con più parole, al fine di individuare un maggior numero di risultati).

WordNet WordNet è un database lessicale organizzato gerarchicamente. Funge sia da thesaurus online, sia da dizionario (include le definizioni).

Internamente è organizzato tramite i *synset*, insiemi di sinonimi aventi un'unica definizione e più *word form* (forme verbali, quindi parole) associate. La lista di parole che compongono il synset, per WordNet, **È**, di fatto, il significato del synset.

Include gerarchie di relazioni tra synset, siano essi nomi (hyponym, hypernym, has-member, member-of, has-part, part-of, antonym), verbi o aggettivi (hypernim, troponym, entails, antonym).

13 Word Sense Disambiguation e Information Retrieval

La Word Sense Disambiguation si occupa, data una parola in un contesto, di decidere quale senso attribuirle.

Le *Selectional Restriction* semantiche possono essere usate come aiuto alla WSD, per risolvere gli argomenti ambigui con predicati (verbi) non ambigui o i predicati ambigui con argomenti non ambigui. Tuttavia ciò non può essere fatto sempre, perchè, come visto in precedenza, le Selectional Restriction vengono violate sistematicamente. Per usare questo approccio bisogna quindi categorizzare le possibili violazioni delle restrizioni, al fine di sfruttarle al meglio.

Un approccio possibile è quello dell'**apprendimento supervisionato**, usando un corpus di training di parole arricchite con tag relativi al significato della parola nel particolare contesto. Si può così addestrare un classificatore che possa **etichettare le parole in nuovi testi**.

Per sfruttare queste tecniche di machine learning, è necessario ricorrere ad apposite rappresentazioni dei dati di addestramento in ingresso, decidendo se limitarsi agli aspetti (*features*) di superficie facilmente estraibili da un testo, oppure utilizzando features che richiedono un'analisi più complessa (ad esempio alberi di parsing). Più tale analisi è approfondita, minore sarà essere il carico di lavoro del sistema di ML.

Rappresentazioni di superficie Tra le possibili rappresentazioni di superficie, troviamo le informazioni di collocazione e di co-occorrenza:

collocazione features relative alle parole in una specifica posizione vicino alla parola target. Possono essere ottenute osservando le parole vicine per ricavare informazioni. Il significato di una parola è strettamente correlato a quello delle parole che le sono vicine.

co-occorrenza features relative a parole che occorrono in posizione qualunque nella finestra (di solito si limitano al calcolo della frequenza).

13.1 Classificatori

Il problema di WSD può essere considerato come un problema di classificazione, utilizzando le tecniche di Data Mining. La scelta della tecnica dipende, in parte, dall'insieme di feature che è stato usato: alcune tecniche lavorano meglio/peggio con features a valori numerici, altre con feature che hanno un ampio insieme di valori.

- **Naive bayes:** utilizzano la formula di Bayes per esprimere la probabilità di ottenere un significato, dato un insieme di feature, in funzione della probabilità di ogni singola feature, noto il significato, sotto l'ipotesi grossolana (naive) di indipendenza delle feature.
- **Decision lists**
- **Decision trees**
- **Neural nets**
- **Support Vector Machines**

13.2 Information retrieval

Si occupa dell'archiviazione e del recupero di documenti, tramite indicizzazione basata sulle parole.

I metodi a **bag of words** portano all'estremo l'interpretazione del principio di **semantica composizionale**: il significato dei documenti è legato unicamente alle parole che li compongono.

I documenti vengono rappresentati come vettori di feature, che rappresentano i termini che appaiono al loro interno. I termini sono pesati in funzione della loro importanza rispetto alla semantica.

I sistemi di information retrieval vengono valutati in base a questi parametri:

Relevance giudizio umano.

Recall $R = \frac{\#doc\,rilevanti\,restituiti}{\#tot\,doc\,rilevanti}$

Precision $P = \frac{\#doc\,rilevanti\,restituiti}{\#doc\,restituiti}$

F-measure misura complessiva che valuta le prestazioni, in funzione di un parametro β che assegna il peso relativo di P e R. Se $\beta < 1$ P pesa di più, se $\beta > 1$ R pesa di più.

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

Si può migliorare ulteriormente la qualità dei risultati tramite tecniche di **relevance feedback** (si estrae un piccolo insieme di documenti e si fa scegliere all'utente quelli che maggiormente lo interessano. Si arricchisce il vettore che rappresenta l'interrogazione sommando un vettore con info relative ai documenti che interessano e si sottrae un vettore con info relative a quelli che non interessano) o **query expansion** (si aggiungono termini correlati con quelli dell'interrogazione di partenza usando un thesaurus).

Altre analisi effettuabili sui documenti con tecniche di Information Retrieval

Categorizzazione

Clustering scopre un insieme di cluster ragionevoli per un dato insieme di documenti, massimizzando la similarità nello stesso cluster e minimizzando la similarità tra i cluster. È applicabile con tecniche di unsupervised ML.

Segmentazione suddivide il documento in sottoporzioni coerenti dal punto di vista semantico. È utile soprattutto per applicazioni con documenti molto grandi ed eterogenei in argomento e per l'estrazione delle sottoparti interessanti del documento.

Riassunto Utilizza l'analisi semantica e tecniche di generazione, oppure l'assegnazione del livello di importanza a tutte le frasi che compongono il documento, seguita dalla definizione di una soglia di interesse e dall'inserimento nel riassunto di tutte e sole le frasi che superano la soglia.

14 La gestione del dialogo

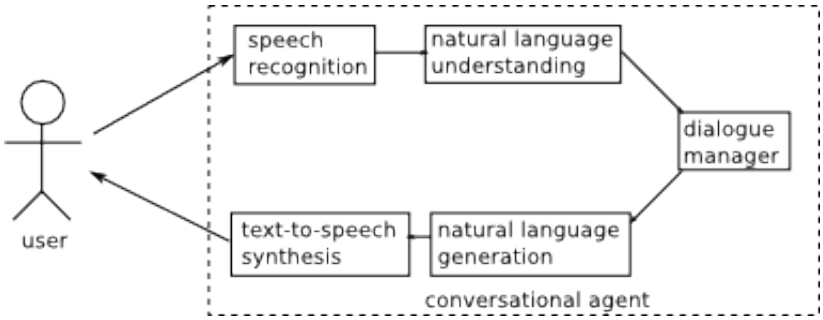
I componenti di un sistema di dialogo devono **riconoscere il parlato** (ASR, *Automatic Speech Recognition*), **comprendere i costrutti** (NLU, *Natural Language Understanding*), **gestire il dialogo**. Necessitano inoltre del supporto alla **pragmatica** (l'insieme di conoscenze che formano il "senso comune") al fine di poter attuare il **grounding** (necessità che ci sia una base comune a tutti i partecipendi al discorso. Si possono porre domande al fine di ricondursi al grounding, nel momento in cui si teme che questo possa essere perso). Si fa uso anche della **confirmation**, cioè la richiesta di conferma sulle informazioni acquisite.

Gli *agenti conversazionali* sono attualmente utilizzati per sistemi di prenotazione di viaggi, smistamento chiamate telefoniche, tutoring (il sistema interroga l'utente su un determinato argomento, commentando la correttezza delle risposte e fornendo ulteriori approfondimenti), comunicazione con robot, applicazioni speciali (sistemi orientati alle abilità della persona, con più o meno capacità di dialogo in funzione dell'abilità d'uso delle interfacce tradizionali da parte dell'utente).

Quando si progetta una agente conversazionale, bisogna prevedere anche la possibilità di **interazioni inattese**, cioè in cui le risposte non sono quelle che si attendono ma sono magari ugualmente importanti e da prendere in considerazione.

La conferma della correttezza della comprensione può essere richiesta esplicitamente, tramite una domanda, oppure semplicemente esprimento i concetti acquisiti in una forma diversa, e ascoltando se ci sono o meno correzioni da parte dell'utente. La gestione del **silenzio** è molto importante: il silenzio contiene informazionie! Nell'ambito della confirmation, può essere spesso interpretato come silenzio-assenso.

14.1 L'architettura dei sistemi di dialogo



14.1.1 Automatic Speech Recognition

Si occupa di riconoscere il parlato e trasformarlo in testo scritto, che verrà poi elaborato dagli algoritmi successivi.

I sistemi ASR lavorano meglio se possono **vincolare le parole** che il parlante probabilmente sta per dire: progettare un ASR per un campo specifico (es: prenotazioni online) invece che uno generale, quindi, porta a risultati migliori, perchè l'area semantica è più ristretta.

I sistemi ASR possono adattarsi al parlante (soprattutto tramite procedure di addestramento), ma non sono privi di errori. **Non si può fare affidamento sul fatto che le parole riconosciute siano corrette**.

14.1.2 Natural Language Understanding

Si occupa dell'analisi semantica del testo riconosciuto. La rappresentazione interna più utilizzata per la semantica nei sistemi di dialogo è quella a **frame e slot**. Un frame rappresenta i concetti in tutte le loro sfaccettature (definisce gl slot, i "campi"), e ha una serie di slot che possono essere man mano riempiti con le conoscenze che si sono acquisite durante il dialogo (o pregresse).

La rappresentazione semantica può essere generata riempiendo gli slot **tramite una grammatica**, che consente di individuare la posizione delle informazioni mancanti. Si utilizza quindi un parser per tener conto delle regole ed etichettare semanticamente la frase. A partire da tale etichettatura, si compilano gli slot mancanti.

In alternativa, si può usare una **cascata di trasduttori a stati finiti**, oppure **regole sintattiche con attachment semantici** (come in VoiceXML, un linguaggio per la progettazione di dialoghi basato su XML, che si combina bene con l'approccio basato su frame a iniziativa combinata).

14.1.3 Gestore del dialogo

È la parte del sistema che si occupa di gestire l'andamento complessivo del dialogo.

A meno di trovarsi in sistemi molto semplici, come il bot ELIZA, il gestore del dialogo ha bisogno di **memorizzare lo stato**, per evitare di ripetere le stesse domande in continuazione.

Per fare ciò esistono tre architetture:

- A stati finiti

  È il metodo più semplice. Il sistema controlla completamente la conversazione con l'utente. Rivolge all'utente una serie di domande e ignora o interpreta erroneamente qualunque cosa che l'utente dica e che non sia una risposta diretta alle domande del sistema.

- Basata su frame

- Planning agents

In tutti i casi bisogna però fare attenzione a mantenere la **computabilità del modello** altrimenti risulta inutile in quanto non è possibile fare inferenze su di esso.

L'iniziativa del dialogo Considerare di chi è l'iniziativa del dialogo significa considerare, in ogni momento della conversazione, chi ha il controllo, cioè chi decide chi deve parlare e quando.

È il tipo di gestore del dialogo a definire il tipo di iniziativa del sistema.

In generale, l'iniziativa può essere:

- **Singola**

- **del sistema**
  Il sistema gestisce tutta la conversazione (come nel caso di architettura FSA)
  Sistemi di questo tipo sono **semplici** da costruire, e l'utente sa sempre cosa stanno per dire. Si trovano a gestire parole note e argomenti conosciuti, quindi ottengono migliori prestazioni per ASR e NLU. Sono tuttavia **troppo limitati**.
- **dell'utente**
  L'utente gestisce tutta la conversazione, ponendo una domanda alla volta, e il sistema risponde. Non si formulano domande concatenate e non si genera un dialogo. Un'impostazione di questo genere può essere utilizzata per effettuare **semplici interrogazioni di DB**. Un esempio di "dialogo" di questo tipo è la ricerca sul Web.

Il problema dei sistemi ad iniziativa singola è che il dialogo coinvolge due interlocutori, quindi l'utente potrebbe voler dire qualcosa che non sia una diretta risposta alla domanda formulata dal sistema, oppure rispondere a più di una domanda in una frase. Il problema può essere parzialmente risolto utilizzando un sistema ad **iniziativa singola con comandi universali**: in tal modo si ha più flessibilità in quanto l'iniziativa rimane al sistema, tuttavia l'utente ha una serie ben definita di comandi che sono sempre a sua disposizione (ad es, tutti gli stati dell'FSA possono essere integrati con i comandi *help* e *correct*). Tuttavia non si realizza una vera iniziativa combinata

- **Combinata**

  Entrambi gli interlocutori possono prendere l'iniziativa in ogni momento, per condurre il dialogo. Un modo per realizzarla è **usare la struttura dei frame** per guidare il dialogo: l'utente può rispondere a più di una domanda per volta (e il sistema riempirà più slot), oppure il sistema può rivolgere domande specifiche per riempire gli slot ancora mancanti.

  Quando il frame è completo, si esegue l'interrogazione corrispondente. Se l'utente formula più domande, si preparano più frame (architettura a **frame multipli**). In tal caso, bisogna poter passare il controllo da un frame all'altro seguendo ciò che l'utente dice e comprendendo in quale casella e in quale frame deve essere inserita l'informazione fornita dall'utente.

  Sistemi di questo tipo si adattano bene all'uso di *VoiceXML*.

  In ogni caso, **si evita di vincolare l'ordine con un'architettura a stati finiti**.

Grounding e confirmation Un dialogo è un'azione collettiva realizzata da parlanti e ascoltatori. Affinchè possa avere luogo, è necessario che ci sia un territorio comune (*common ground*) di credenze e conoscenze possedute da parlanti e ascoltatori. Comprende i precedenti scambi comunicativi. Perchè avvenga uno scambio reciproco di informazioni è necessario **raggiungere il territorio comune** e **mantenerlo**, anche attraverso chiarimenti espliciti: tale operazione viene detta **grounding**.

Inoltre, durante un dialogo, gli agenti che svolgono un'azione (ad esempio comunicare un'informazione) richiedono una prova del raggiungimento del risultato (ad esempio, tra persone, un cenno col capo al termine di una frase), in misura dipendente dal tipo di azione in corso. Infatti, gli utenti di interfacce basate sul parlato restano disorientati se il sistema non dà riscontri.

Esempi di riscontri (chiamati ***confirmation***) sono:

attenzione continuata B presta attenzione ad A con continuità

prossimo contributo significativo B avvia il prossimo contributo rilevante (ad esempio, richiede ulteriori informazioni non ancora fornite)

acknowledgement B annuisce, o dice qualcosa tipo "sì", "certo", "già".

dimostrazione B mostra ad A di aver capito parafrasando, riformulando o ripetendo alla lettera il contributo di A o completando in modo collaborativo la sua frase.

La conferma permette di rilevare e correggere l'eventuale presenza di errori nel riconoscimento del parlato, oppure interpretazioni errate da parte dell'ascoltatore.

La **conferma** può anche essere **ottenuta esplicitamente**, ponendo una domanda diretta per verificare se l'informazione acquisita è giusta (ad esempio, in caso di forte dubbio sulla correttezza da parte del sistema).

La conferma esplicità è più difficile e lunga da realizzare, anche se è più facile per l'utente correggere il sistema.

La conferma implicita è più naturale, più veloce e più semplice (se il sistema si comporta nella maggior parte dei casi in modo corretto).

14.1.4 Natural Language Generation

La generazione del testo in uscita può seguire due approcci principali:

- Semplici template (prescripted sentences)
- Unificazione: usa le regole grammaticali per costruire nuove frasi

14.2 Dialoghi complessi

I sistemi a stati finiti o basati su frame possono gestire solo dialoghi semplici: hanno problemi con le domande inattese da parte dell'utente, perchè non riescono a modellarle.

Analogamente, si possono avere dei problemi quandi vengono fornite informazioni a prima vista non direttamente correlate alla domanda che è stata posta (es: "Quando vuoi fare il viaggio?" "Ho un meeting là il 12 di questo mese": implica che il viaggio deve essere fatto prima). Come può l'agente inferire queste informazioni da ciò che l'utente dice? Sfruttando il principio di ***Conversational Implicature*** di Grice:

> "Per mettere in condizioni l'ascoltatore di inferire correttamente, deve essere rispettato il principio di cooperazione" (Grice 75)

Questo è un implicito accordo di cooperazione alla comunicazione tra il parlante e gli ascoltatori. Si suppone cioè che l'altro stia cooperando.

Il principio di Grice può essere dettagliato con 4 massime:

Rilevanza sii rilevante (incisivo). Se qualcosa viene nominato, allora è importante, e bisogna cercare di capire in che modo lo è, per sfruttare le informazioni che questo comporta.

Quantità fai in modo che il tuo contributo non sia nè troppo informativo, nè troppo poco, rispetto a quanto richiesto.

Qualità fai in modo che il tuo contributo sia vero. Non dire cose false o non sufficientemente provate o evidenti (NB: si può dire il falso se ciò è evidente, ad esempio quando si usa l'ironia, sottolineata da un apposito tono di voce).

Modalità evita l'ambiguità e la poca chiarezza. Sii conciso e ordinato.

14.2.1 Atti linguistici (*speech acts*)

Una produzione linguistica è un tipo di azione (Austin, 1962). Quando si dice qualcosa si provoca un effetto sulla realtà.

Ogni produzione include in sé tre atti:

atto locutorio la produzione con un particolare significato (la frase in sé)

atto illocutorio l'atto di domandare, rispondere, promettere, ... che si sottintende nel produrre la frase

atto perlocutorio gli effetti che la frase mira a produrre sul pensiero, il sentimento o le azioni.

Searle ha fornito una categorizzazione degli atti linguistici in 5 classi:

Assertives il parlante afferma qualcosa come vero

Directives il parlante cerca di far far qualcosa all'ascoltatore

Commissives il parlante si impegna a far qualcosa in futuro

Expressives il parlante esprime il proprio stato psicologico relativo a una certa situazione

Declarations generano un nuovo stato del mondo per mezzo delle parole ("mi licenzio", "sei licenziato").

14.2.2 Atti dialogici

Sono atti con una struttura interna specifica. Incorporano aspetti legati al grounding e alla confirmation, nonchè altre funzioni legate al dialogo e alla conversazione di cui Austin e Searle non tengono conto (ringraziamenti, richieste di informazioni, suggerimenti, domande, risposte, ...).

Esistono diversi insiemi di categorie di atti dialogici, tra cui ricordiamo **Verbmobile Dialogue Acts** e **DAMSL**.

Un dialogo può essere interamente etichettato con gli atti dialogici, per capire il ruolo di ogni frase.

Identificare automaticamente gli atti dialogici è complesso: non possiamo guardare solo la forma della produzione, non possiamo guardare la forma sintattica della superficie (un ordine e una domanda possono corrispondere entrambi ad una richiesta: "Mi dai il tuo panino?" "Dammi il tuo panino"), perchè c'è troppa ambiguità.

Bisogna ricorrere ad una **classificazione statistica**, che prenda in considerazione i molti aspetti delle frasi che ci possono indicare di che tipo di *Dialogic Act* si tratta, come ad esempio la posizione delle parole, la prosodia (modulazione della pronuncia), la struttura della conversazione. Si possono anche utilizzare una serie di regole scritte a mano, che possano aiutare ad individuare un particolare tipo di DA.

14.2.3 Planning-based conversational agents

Quando si progetta un sistema di dialogo, soprattutto se complesso, è necessario porre da subito **attenzione all'utente** e ai suoi scopi.

I sistemi di dialogo più raffinati riconoscono il grounding e si impegnano attivamente per realizzarlo, riconoscono le intenzioni dell'utente ed i suoi atti dialogici, nonchè le inferenze indicate da Grice. Rispettano quindi gli obblighi legati al dialogo (rispondono alle domande, realizzano comandi.

Per riuscire a fare ciò si usano le tecniche di ragionamento dei sistemi di **planning**, che rappresentano, oltre ai **fatti**, anche i **desideri**, le **credenze** e le **intenzioni**, riescono ad agire in modo tale da **raggiungere un obiettivo**, tramite apposite **azioni**, identificate da:

precondizioni condizioni che devono essere vere affinche sia possibile compiere l'azione

corpo insieme di scopi parzialmente ordinati che devono esser raggiunti nello svolgimento dell'azione, ossia ciò che è necessario fare per rendere veri gli effetti dell'azione.

effetti condizioni che diventano vere come risultato dello svolgimento con successo dell'azione.

Il funzionamento dei sistemi planning-based

1. **While** la conversazione non è finita
   (a) **if** l'utente ha completato un turno
      i. interpreta ciò che l'utente ha pronunciato
   (b) **if** il sistema ha degli obblighi in attesa
      i. assolvi gli obblighi
   (c) **else if** il sistema detiene il turno
      i. **if** il sistema ha compreso i conversation acts
         A. chiama il generatore per produrre la risposta vocale
      ii. **else if** manca il grounding per qualcosa
         A. cerca di creare il grounding
      iii. **else if** gli obbiettivi di alto livello non sono soddisfatti
         A. prova a soddisfare gli obiettivi
      iv. **else** rilascia il turno o prova a terminare la conversazione
   (d) **else if** nessuno detiene il turno, oppure c'è stata una lunga pausa
      i. prendi il turno

14.2.4 Valutazione dei sistemi di dialogo

I sistemi di dialogo possono essere valutati in funzione di alcuni parametri:

- valutazione del successo rispetto al compito
  - percentuale di sottoscopi realizzati in modo completo
  - correttezza di ogni domanda/risposta/indicazione
  - correttezza complessiva delle soluzioni adottate
- valutazione dei costi
  - tempo di completamento in turni/secondi
  - numero di interrogazioni
  - $\frac{numero\,di\,turni\,necessari\,per\,la\,sola\,correzione\,di\,errori}{numero\,totale\,dei\,turni}$
- inappropriatezza (verbosità, ambiguità)

  - delle domande, delle risposte, della messaggistica di errore del sistema

- soddisfazione dell'utente
  - le risposte sono state prodotte rapidamente?
  - il sistema capisce le domande al primo tentativo?
  - il sistema è facilmente utilizzabile da persone non in confidenza con strumenti informatici? è accessibile (anche nei confronti di persone con disabilità)

14.3 Risoluzione dei riferimenti

All'interno di un dialogo o di un discorso esistono diversi riferimenti (ad es, **pronomi**) che devono essere risolti al fine di comprendere correttamente il significato del testo.

Nella frase "John went to Bill's car dealership to check out an Acura Integra. He looked at it for half an hour", possiamo identificare diversi elementi:

referring expression John, he. Sono le espressioni che si riferiscono al referente.

referent l'entità reale (nel nostro caso, John)

coreferenziazione diciamo che "John" e "he" si **coreferenziano**, perché si riferiscono alla stessa entità

antecendente "John". Termine a cui ci si riferisce e che compare prima del termine che si riferisce ad esso.

anafora "he". Termine che fa riferimento a qualcosa che è già comparso in precedenza.

catafora (non presente in questa frase). Termine che si riferisce a qualcosa che verrà citato in seguito: "Before he bought *it*, John checked over the Integra very carefully".

Termini come *that* possono dar vita a vari tipi di riferimenti. Possono infatti riferisi ad un atto linguistico, ad una proposizione, ad una modalità descrittiva, ad un evento, ad una combinazione di diversi eventi.

Il problema dei riferimenti riguarda anche gli **articoli** che introducono le noun phrase, che si possono riferire a qualcosa di ancora ignoto (*indefinite noun phrases*, introdotte dall'articolo **indeterminativo**), oppure a qualcosa di identificabile dall'ascoltatore perchè già menzionato in precedenza, oppure identificabile in base alla conoscienza comune o intrinsecamente unico (*definite noun phrases*, introdotte dall'articolo **determinativo**).

L'**utilizzo di inferenze** da vita a riferimenti potenzialmente complessi da trovare: "I bought an Acura Integra today, but *the engine* seemed noisy". Per comprendere la frase, bisogna sapere che ogni auto ha un motore. Altrimenti "the" non ha senso.

Anche la risoluzione dei **pronomi** non è semplice, in quanto ogni pronome può riferirsi, potenzialmente, a molti termini. Si affronta questo problema con i seguenti approcci:

- Hard constraints on coreference
- Soft constraints on coreference

La risoluzione si basa sulla **concordanza** del **numero**, della **persona**, del **genere** e del **caso**. Sfrutta vincoli sintattici. Usa inoltre una regole: **selectional restriction** (ossia si sceglie il riferimento in base al verbo che accompagna il pronome. Se il verbo è "drive", il riferimento sarà "car", e non "garage"), **occorrenza recente** (a parità di concordanza, si sceglie il riferimento che è comparso più di recente nella frase), **ruolo grammaticale** (il *soggetto* è un riferimento preferito rispetto ai complementi), **menzione ripetuta** (se ci sono tanti riferimenti ad un termine - ad esempio a "John" - è probabile che anche i prossimi riferimenti si collegheranno a tale termine), **preferenza di parallelismo** (se due frasi vicine hanno la stessa struttura, e in una delle due un termine è sostituito da un pronome, è probabile che tale pronome si riferisca al termine che sostituisce), **preferenza semantica del verbo** (*verb semantic preference*: ogni verbo ha una causa implicita. Per taluni l'oggetto, per altri il soggetto. I pronomi fanno di solito riferimento a tale causa).

Pronoun resolution algorithm Un algoritmo di risoluzione dei pronomi è stato sviluppato da Lappin e Leass. Considera solo i riferimenti recenti e le concordanze sintattiche.

Funziona in due passi:

- **Aggiorna il modello** del discorso: quando incontra una noun phrase aggiunge un tag al modello del discorso e le assegna un **valore di importanza**, pesandolo in base ad alcuni fattori, come la recentezza o le preferenze rispetto al ruolo grammaticale. I valori vengono poi **dimezzati dopo che ogni frase viene processata**. La relativa noun phrase diventa quindi sempre meno preferita.
- **Risolve i pronomi**: sceglie l'antecedente più importante.

L'algoritmo colleziona i potenziali riferimenti **fino a 4 frasi prima**. Inoltre cancella i riferimenti che non soddisfano eventuali vincoli di numero e genere con il pronome e quelli che non soddisfano i vincoli di coreferenza sintattica.

Coerenza Un testo può essere definito coerente se:

- fa un uso appropriato delle relazioni di coerenza tra le sottoparti del discorso (*rethorical structure*)
- usa una sequenzializzione appropriata delle sottoparti del discorso (*discourse/topic structure*)
- fa un uso appropriato delle espressioni che includono riferimenti (*referring expressions*).

Le **Hobbs Coherence Relations** sono una serie di etichette da applicare alle parti del discorso e che indicano le relazioni di coerenza che sono presenti tra esse:

Result lo stato o l'evento asserito da S0 causa o potrebbe causare l'evento asserito da S1

Explanation lo stato o l'evento asserito da S1 causa o potrebbe causare lo stato o l'evento asserito da S0

Parallel dall'asserzione di S0 è possibile inferire $p(a_1, a_2, \ldots)$ e dall'asserzione di S1 $p(b_1, b_2, \ldots)$, dove a_i e b_i sono simili per ogni i. Es: "John bought an Acura. Bill leased a BMW"

Elaboration dalle asserzioni di S0 e S1 si può inferire la stessa proposizione P, anche se in forma più o meno elaborata. "John bought an Acura this weekend. He puchased a beatiful new Integra for 20 000$ on Saturday afternoon".