Lavorare con i Corpora

Marco Baroni

20 febbraio 2005

1 Usare i corpora nel lavoro linguistico

- Procedura:
  1. Raccogli testi in lingua/su argomento/in stile di interesse.
  2. (Opzionale) Annotali, e.g., con informazioni su parte del discorso e meta-dati (autore, data, etc.)
  3. (Opzionale per corpora di piccole dimensioni) Indicizza dati in modo da renderne possibile la consultazione.
  4. Esplora questioni linguistiche attraverso concordanze, liste di frequenza.
  5. (Opzionale) Estrai automaticamente informazioni da corpus con filtri statistici/linguistici.
- Una versione sofisticata di quello che quasi tutti fanno con Google.
- Alcuni corpora più o meno famosi: Brown, LOB, BNC, EuroParl, CORIS/CODIS, *La Repubblica*...
- Alcune questioni: dimensioni, livello di annotazione, parallelismo, paragonabilità, bilanciamento, "disposable corpora".

2 Dati testuali e testo

- Nei computer, i dati testuali spesso sono conservati in formati differenti dal formato testo.
- Per trasformare un insieme di documenti in un corpus, è prima di tutto necessario convertirli in formato testo.
- Questa operazione è più semplice, ovviamente, quando i documenti sono già in formato testo.
- Fortunatamente, l'HTML, il linguaggio in cui sono scritte gran parte delle pagine web, viene conservato in documenti in formato testo puro, che ne rende facile l'utilizzo come materiali per un corpus.

2.1 Che cos'è il formato testo

2.1.1 Bits, bytes, caratteri e codici

- Per un sistema operativo, un *file* è una sequenza di *bits* (numeri binari: 0/1).
- Una sequenza di 8 bits costituisce un *byte*.
- Un file di testo è semplicemente un file in cui ciascun byte viene usato per rappresentare lettere dell'alfabeto, numeri, simboli, caratteri di controllo (a capo e simili), seguendo un *codice* (tabella di corrispondenza tra bytes e caratteri) ben definito.
- Un *programma* in grado di manipolare files di testo legge la sequenza di bytes contenuti in un file e tratta ciascuna sequenza di 8 bits (ciascun byte) come un carattere.
- Vari codici standardizzati specificano corrispondenza tra bytes e caratteri.
- Uno dei primi e il più famoso è il codice ASCII.
- Il codice più diffuso per le lingue dell'Europa occidentale (incluse lingue "minori" quali basco, faroese e finlandese) è un'estensione dell'ASCII nota come latin-1 (o ISO-8859-1).
- Esempi:

```
01000001 -> A
01100001 -> a
00110010 -> 2
11101000 -> è
00001010 -> \n (new line)
```

- Per noi umani (programmatori a parte), i numeri binari non sono il massimo, e dunque le tabelle ASCII/latin-1 tipicamente riportano i numeri corrispondenti in base 10 (decimale), 8 (ottale) o 16 (esadecimale)[1]

carattere	binario	ottale	esadecimale	decimale
A	01000001	101	41	65
a	01100001	141	61	97
2	00110010	62	32	50
è	11101000	350	E8	232
\n	00001010	12	A	10

Tabella 1: Alcuni caratteri e bytes corrispondenti in latin-1

[1] A dire il vero, gran parte degli umani non programmatori si trovano bene soltanto con i decimali.

- Prima di abbandonare l'affascinante mondo dei bits, si noti che con 8 bits (cioè un byte) possiamo rappresentare 256 oggetti distinti (2^8: da 00000000 a 11111111).
- Dunque, un codice basato su bytes singoli non può rappresentare più di 256 caratteri.
- Conseguenze:
  - Proliferare di standard diversi per lingue diverse (in parte, sotto l'egida dell'ISO).
  - Sistemi a più bytes per lingue con alfabeti molto estesi (cinese, giapponese...)
- Unicode: la soluzione?
  - Codice "standard" che si propone di rappresentare *tutti i caratteri conosciuti*.
  - Definito in astratto, implementazioni differiscono (le più comuni: utf-8, utf-16), molta confusione tra i non esperti (e anche tra gli esperti).
  - Unicode-support non ancora molto diffuso (lo sarà mai?)
  - The good news: latin-1 è un sotto-insieme di unicode/utf-8.

2.1.2 Andare a capo con dos, mac e unix

- Caratteri di controllo in ASCII/latin-1 ci riportano al tempo in cui i computer mandavano fisicamente l'output in stampa ad una telescrivente (una specie di macchina da scrivere controllata dal computer).
- Al tempo, quando voleva andare a capo, il computer doveva dare due istruzioni alla telescrivente:
  - Riporta il carrello a sinistra.
  - Scorri in giù di una riga.
- Dunque, servivano due caratteri di controllo: il *carriage return* (CR, \r, decimale 13) e la newline o *line feed* (LF, \n, decimale 10).
- Con l'avvento dei monitor, non c'è più bisogno di dare istruzioni per lo spostamento meccanico di una testina, dunque non è necessario rappresentare l'*a capo* come una sequenza di due caratteri.
- Con grande senso d'armonia, le tre famiglie di sistemi operativi più comuni hanno scelto tre convenzioni diverse:
  - Dos/Windows: CR LF
  - Mac: CR
  - Unix: LF
- I problemi che si incontrano trasferendo un file di testo da un sistema all'altro dipendono quasi sempre dal modo diverso di indicare gli *a capo*.

2.2 Testo puro vs. altri formati per i dati testuali

- Il testo puro è il formato più adatto (anzi, l'unico formato proponibile) per la gestione di corpora e altri database linguistici (lessici, liste di frequenza, sistemi di relazioni concettuali...)[2]
- Formati quali *pdf* e *doc* (il formato dei files di MS Word) possono essere più adeguati per altri scopi (per esempio, possono essere più piacevoli all'occhio umano), ma non sono utilizzabili per l'analisi linguistica.
- Il vantaggio del formato testo sta nella sua semplicità: un file di testo è costituito da una sequenza di bytes, dove ciascun byte rappresenta un carattere secondo un codice pubblicamente disponibile.[3]
- La semplicità del formato testo fa sì che praticamente tutti i tools (Unix e non), gli editors e in generale tutte le applicazioni per la manipolazione di dati testuali, lo supportino.
- Inoltre, è estremamente facile sviluppare nuove applicazioni che gestiscano il formato testo.
- Ovviamente, anche files in altri formati, quali *pdf* e *doc*, dal punto di vista del computer, sono sequenze di bits.
- Tuttavia, solo certi programmi (MS Word, Acrobat Reader) sono in grado di interpretare correttamente tali sequenze di bits.
- Nel caso di *doc* e altri formati della Microsoft, la situazione è aggravata (dal nostro punto di vista) dal fatto che si tratta di formati *proprietari*: ossia, in parole povere, la maniera in cui i bits vanno organizzati per costruire un file *doc* (o *xls*, o *ppt*...) di senso compiuto è un segreto industriale della Microsoft.
- Di conseguenza, dati in questi formati possono venire utilizzati solo con programmi Microsoft, e sono inutili ai nostri fini.
- Ovviamente, un file in formato *pdf* o *doc* può venir letto anche da un programma che si aspetta testo puro.
- Tuttavia, tale programma tratterà ogni sequenza di 8 bits nei files in questione come un carattere ASCII/latin-1, e il risultato non avrà molto senso.

[2] Alcune applicazioni linguistiche, come WordSmith Tools e l'IMS Corpus Workbench, mantengono i dati in un loro formato interno. Tuttavia, anche queste applicazioni richiedono testo puro in input, e possono produrre testo puro come output.

[3] Se il file si basa su codici per lingue asiatiche o certe implementazioni di unicode, ciascun carattere è rappresentato da una sequenza di bytes, ma l'idea di base è la stessa.

- Se proprio bisogna lavorare con formati di questo tipo, esistono command line tools (quali `antiword` e `pdftotext`) che convertono da questi formati a testo, con risultati alterni.[4]

2.3 L'HTML

- L'HTML viene scritto in files di testo puro.
- La formattazione e altri aspetti non strettamente testuali (in particolare: collegamenti intra- ed inter-testuali) non sono espressi al livello dei bits, ma esplicitamente come istruzioni specificate all'interno del testo.
- Per esempio, una parola in grassetto in HTML si rappresenta così:

```
<b>hi</b>
```

- Una sequenza di questo genere viene trattata come normalissimo testo da un programma che legge testo, mentre programmi specializzati (*in primis*, web browsers quali Mozilla/Firefox e Internet Explorer) interpretano le istruzioni di formattazione in HTML, visualizzando la pagina come specificato (in questo caso, nascondendo `<b>` e `</b>` e facendo vedere la stringa **hi** in grassetto).
- Dal nostro punto di vista, l'HTML è un'ottima cosa: significa che gran parte delle pagine web sono già in formato testo – pronte per essere usate come corpora!
- Ci sono programmi sofisticati per ripulire l'HTML, ma l'idea di base è semplicissima: basta rimuovere le *tags* (`<b>` et sim.) dall'HTML e si ottiene testo relativamente pulito.

2.3.1 XML: un altro markup language

- Ci riferiamo a linguaggi quali HTML e XML, dove informazioni metatestuali di vario genere vengono espresse in testo puro usando una sintassi particolare, con il termine di *markup languages*.
- XML è un "meta-linguaggio" adatto a rappresentare dati strutturati, concentrandosi sui contenuti e la struttura (a differenza di HTML, che è pensato soprattutto per la visualizzazione).
- XML, tra l'altro, è un ottimo linguaggio per codificare lessici, database terminologici e simili:

[4] Tra l'altro, i files *doc* salvati come testo seguono un codice che è simile al latin-1, ma non è proprio latin-1, creando una serie di piccoli ma fastidiosi problemi in fase di tokenizzazione.

```
<lexicon lang="it">
<form id="1" eng_corresp_id="33">
<lemma>cane</lemma>
<pos>N</pos>
<meaning>essere peloso che abbaia</meaning>
</form>
<form id="2" eng_corresp_id="12">
<lemma>gatto</lemma>
<pos>N</pos>
<meaning>essere peloso che miagola</meaning>
</form>
</lexicon>
```

3 La rete come corpus

- Internet è una gigantesca rete di computer connessi che mettono a disposizione, attraverso vari protocolli, database di documenti.
- Questi documenti sono in larga parte serviti al pubblico in HTML (ossia, come abbiamo visto, in formato testo).
- Quante parole ci sono in rete? Secondo le stime obsolete e conservative di Kilgarriff e Grefenstette (Introduction to the Special Issue on the Web as Corpus, *Computational Linguistics* 29, 2003):

  inglese 76,598,718,000

  tedesco 7,035,850,000

  francese 3,836,874,000

  spagnolo 2,658,631,000

  italiano 1,845,026,000

  finlandese 326,379,000

  esperanto 57,154,000

  latino 55,943,000

  basco 55,340,000

  albanese 10,332,000

- Cfr.: Brown: 1M, BNC: 100M, *La Repubblica*: 380M.

3.1 Automatizzare la navigazione

- Browsers quali Mozilla/Firefox o Internet Explorer espletano principalmente due funzioni:
  - Connettersi a un server e scaricarne il documento richiesto.

  - Visualizzare tale documento secondo le istruzioni specificate dall'HTML, mostrando la formattazione, i links ipertestuali, le immagini, ecc.

- Se vogliamo estrarre un corpus dalla rete, la funzione di visualizzazione non ci interessa, e scaricare a mano attraverso il browser tutti i documenti potenzialmente interessanti salvandoli come testo è un processo lento e noioso.

- Per fortuna, esistono tools che ci permettono di ottenere materiale dalla rete in modalità non interattiva: per es., scaricando tutte le pagine di un sito in un sol colpo, seguendo links automaticamente, eseguendo ricerche automatiche su Google, ecc.

3.1.1 I BootCaT Tools

- `http://sslmit.unibo.it/~baroni/bootcat.html`

- M. Baroni e S. Bernardini. BootCaT: Bootstrapping Corpora and Terms from the Web. *LREC 2004*.

- L'idea di base:

  1. Comincia con un piccolo insieme di "seeds" (parole che sembrano tipiche del dominio d'interesse).
  2. Combina i seeds a caso in una serie di tuple (triplette, coppie...), in modo da trovare pagine che si spera siano caratteristiche del dominio d'interesse.
  3. Usa tali combinazioni come queries in una serie di ricerche su Google, e scarica le prime N pagine trovate per ciascuna query.

- Nuovi seeds possono venire estratti dal corpus, e la procedura ripetuta.

- Interfaccia a Google tramite Google APIs.[5]

4 Distribuzioni di frequenza

- Frequenze giocano ruolo fondamentale in lavoro su corpora.

- E' dunque importante avere idee chiare su distribuzione tipica di frequenze in corpora.

[5] `http://www.google.com/apis/`

4.1 Terminologia e concetti di base

- *Tokens*: parole in un corpus (due istanze della stessa parola sono due tokens separati).
- *Types*: parole *distinte* in corpus.
- Problemi di *tokenizzazione* (e.g., *New York* è un token o due?) e attribuzione di tokens a types (e.g., *Il* e *il* vanno assegnati a types distinti?)
- Lista di frequenza (numeri di tokens per type):

type	f	type	f
again	2	he	1
and	3	her	1
another	1	that	2
bark	1	this	1
barks	6	will	1
dog	3	with	1
friends	1		

- Due modi di organizzare dati di frequenza per analizzarli:
  - *Spettri di frequenze*: per ciascun *livello* di frequenza, quanti tipi hanno quella frequenza:

f	V(f)
1	8
2	2
3	2
6	1

  - *Profili rango/frequenza*: Assegna ranghi a tipi sulla base di loro frequenza (parola più frequente ha rango 1, seconda parola più frequente ha rango 2, ecc., con assegnazione arbitraria per parole con la stessa frequenza):

r	f	r	f
1	6	8	1
2	3	9	1
3	3	10	1
4	2	11	1
5	2	12	1
6	1	13	1
7	1		

top frequencies			*bottom frequencies*		
rank	fq	word	rank range	fq	randomly selected examples
1	62642	the	7967-8522	10	recordings undergone privileges
2	35971	of	8523-9236	9	Leonard indulge creativity
3	27831	and	9237-10042	8	unnatural Lolotte authenticity
4	25608	to	10043-11185	7	diffraction Augusta postpone
5	21883	a	11186-12510	6	uniformly throttle agglutinin
6	19474	in	12511-14369	5	Bud Councilman immoral
7	10292	that	14370-16938	4	verification gleamed groin
8	10026	is	16939-21076	3	Princes nonspecifically Arger
9	9887	was	21077-28701	2	blitz pertinence arson
10	8811	for	28702-53076	1	Salaries Evensen parentheses

Tabella 2: Frequenze alte e basse nel Brown

4.2 Patterns tipici di frequenza

- Parole più e meno frequenti in Brown corpus riportate in tavola 2, profilo rango/frequenza e spettro di frequenze in figura 1.
- Tipico andamento "pochi giganti, un esercito di nani":
  - Le 10 parole (tipi) più frequenti corrispondono al 23% dei tokens nel Brown.
  - Le parole con frequenza 3 o inferiore corrispondono al 70% dei tipi, e a solo 5% dei tokens nel Brown.
  - Quasi metà delle parole capitano una sola volta (*hapax legomena*).
- Effetto di distribuzione fortemente asimmetrica su misure di tendenza centrale:
  - frequenza media: 19;
  - frequenza mediana: 2;
  - moda delle frequenze: 1.
- Pattern non limitato a Brown, ma riscontrato in praticamente tutti i corpora conosciuti, indipendentemente da lingua, dimensioni, tipo, come mostrato nelle figure 2 (profili rango/frequenza) e 3 (spettri di frequenze).
- Pattern fortemente asimmetrico di frequenze noto come distribuzione zipfiana, da George Kingsley Zipf, che nella prima metà del secolo propose la seguente legge che mette in relazione rango e frequenza:

$$f = \frac{C}{r}$$

- Si noti che legge di Zipf predice calo molto rapido di frequenza tra i primi ranghi, e lunghissima coda di parole a bassa frequenza.

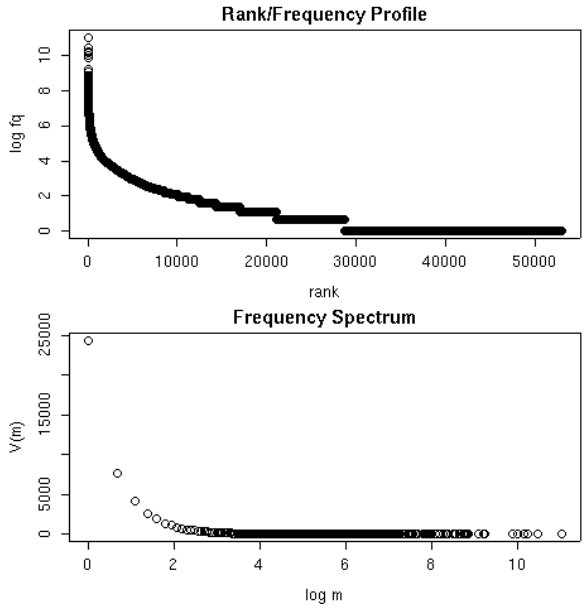

Figura 1: Profilo rango/frequenza e spettro di frequenze nel Brown.

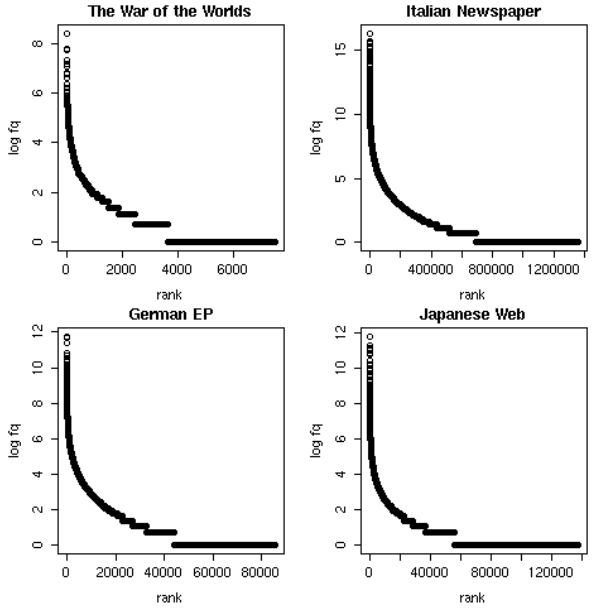

Figura 2: Profili rango/frequenza della novella *The War of the Worlds* (in alto a sinistra), del corpus *la Repubblica* (in alto a destra), dell'annata 2002 dell'*EuroParl* tedesco (in basso a sinistra) e di un corpus di pagine web giapponesi (in basso a destra).

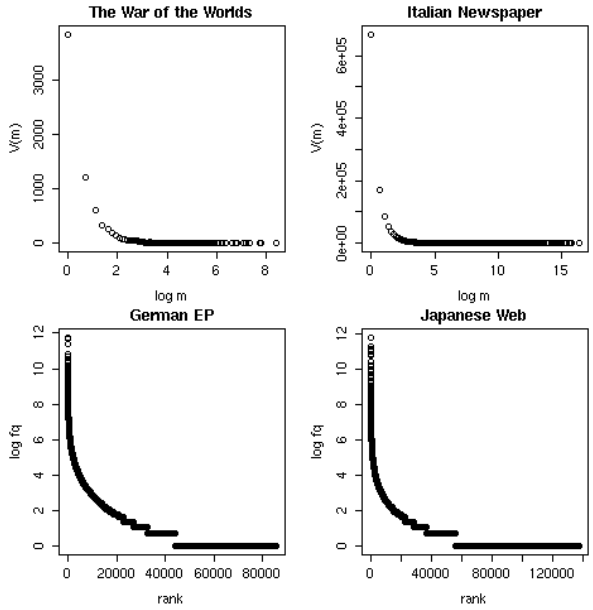

Figura 3: Spettri di frequenze della novella *The War of the Worlds* (in alto a sinistra), del corpus *la Repubblica* (in alto a destra), dell'annata 2002 dell'*EuroParl* tedesco (in basso a sinistra) e di un corpus di pagine web giapponesi (in basso a destra).

4.3 Conseguenze della distribuzione zipfiana

- *Data sparseness*: per quanto grande sia un corpus, la maggioranza delle parole che capitano in esso hanno frequenza bassissima e molte parole della lingua in analisi non capitano affatto (si pensi a conseguenze per lessicografi/terminografi).
- Molte misure/tecniche statistiche standard non si applicano alle distribuzioni di frequenza. Per es.:
  - Come abbiamo visto, la media non è in alcun modo una misura significativa essendo "gonfiata" da frequenza di piccolo set di parole frequentissime; la mediana e la moda sono stabili ma poco interessanti (la mediana è sempre 1 o 2, la moda sempre 1).
  - Non possiamo paragonare numero di tipi in corpora diversi (per misurare "ricchezza lessicale" di corpora), visto che numero di parole cresce con dimensioni di corpora, come indicato dalla continua presenza di parole con frequenza 1 (vedi figura 4).[6]
  - Tecniche statistiche che assumono normalità dei dati (andamento a "campana") non sono adeguate per studio di distribuzione di frequenze.

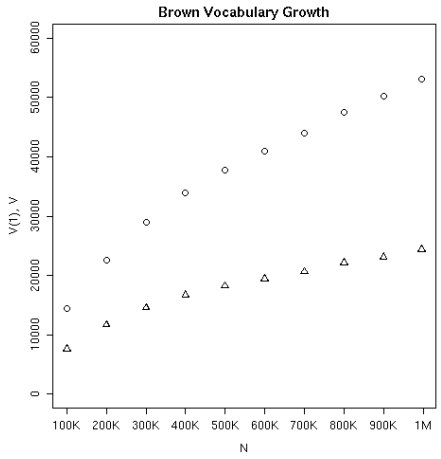

Figura 4: Numero di tipi (cerchi) e numero di tipi con frequenza 1 in funzione del numero di tokens nel Brown.

[6] La *type/token ratio*, una misura che talvolta si incontra nella letteratura, è priva di significato, perché diminuisce sistematicamente man mano che le dimensioni del corpus aumentano.