Word Prediction

seminario per il corso di Elaborazione del Linguaggio Naturale

Francesco Varrato

Lunedi 11 Luglio 2005

Presentazione del seminario

Questo lavoro è rivolto alla esplicitazione delle modalità con cui si può effettuare la *word prediction*, ovvero la predizione di parole. Allo scopo verranno esposti nell'ordine:

- utilità della WP
- linguistica computazionale e statistica
- funzionamento della WP
- possibili sviluppi

Presentazione del seminario

Questo lavoro è rivolto alla esplicitazione delle modalità con cui si può effettuare la *word prediction*, ovvero la predizione di parole. Allo scopo verranno esposti nell'ordine:

- utilità della WP
- linguistica computazionale e statistica
- funzionamento della WP
- possibili sviluppi

Presentazione del seminario

Questo lavoro è rivolto alla esplicitazione delle modalità con cui si può effettuare la *word prediction*, ovvero la predizione di parole. Allo scopo verranno esposti nell'ordine:

- utilità della WP
- linguistica computazionale e statistica
- funzionamento della WP
- possibili sviluppi

Presentazione del seminario

Questo lavoro è rivolto alla esplicitazione delle modalità con cui si può effettuare la *word prediction*, ovvero la predizione di parole. Allo scopo verranno esposti nell'ordine:

- utilità della WP
- linguistica computazionale e statistica
- funzionamento della WP
- possibili sviluppi

Presentazione del seminario

Questo lavoro è rivolto alla esplicitazione delle modalità con cui si può effettuare la *word prediction*, ovvero la predizione di parole. Allo scopo verranno esposti nell'ordine:

- utilità della WP
- linguistica computazionale e statistica
- funzionamento della WP
- possibili sviluppi

A chi e a cosa serve

Si vuole un sistema che sia particolarmente indicato nelle situazioni di:

Problemi

- *persona disabile* che presenti solo difficoltà di tipo linguistico o motorio (ad es. per chi è affetto da dislessia o per chi affetto da malattie neuromuscolari degenerative, o per traumi riportati in seguito ad incidenti);
- necessità di velocizzare la scrittura di *documenti per uffici* di enti pubblici e privati;
- scrittura su *dispositivi mobili* (quali ad es. telefoni cellulari e palmari).

A chi e a cosa serve

Si vuole un sistema che sia particolarmente indicato nelle situazioni di:

Problemi

- *persona disabile* che presenti solo difficoltà di tipo linguistico o motorio (ad es. per chi è affetto da dislessia o per chi è affetto da malattie neuromuscolari degenerative, o per traumi riportati in seguito ad incidenti);
- necessità di velocizzare la scrittura di *documenti per uffici* di enti pubblici e privati;
- scrittura su *dispositivi mobili* (quali ad es. telefoni cellulari e palmari).

A chi e a cosa serve

Si vuole un sistema che sia particolarmente indicato nelle situazioni di:

Problemi

- *persona disabile* che presenti solo difficoltà di tipo linguistico o motorio (ad es. per chi è affetto da dislessia o per chi affetto da malattie neuromuscolari degenerative, o per traumi riportati in seguito ad incidenti);
- necessità di velocizzare la scrittura di *documenti per uffici* di enti pubblici e privati;
- scrittura su *dispositivi mobili* (quali ad es. telefoni cellulari e palmari).

A chi e a cosa serve

Si vuole un sistema che sia particolarmente indicato nelle situazioni di:

Problemi

- *persona disabile* che presenti solo difficoltà di tipo linguistico o motorio (ad es. per chi è affetto da dislessia o per chi affetto da malattie neuromuscolari degenerative, o per traumi riportati in seguito ad incidenti);
- necessità di velocizzare la scrittura di *documenti per uffici* di enti pubblici e privati;
- scrittura su *dispositivi mobili* (quali ad es. telefoni cellulari e palmari).

Chi sono e cosa fanno

Molti prodotti realizzano la predizione solo a livello lessicale (cioé sulla base di un dizionario di frequenza d'uso personalizzato)

Prodotti

- *SofType*: aggiorna le frequenze ma non c'è controllo sintattico;
- *C.A.R.L.O.*: non c'è controllo sintattico;
- *Dedalus*: effettua il controllo sintattico ma non considera le freq. ed inoltre il programma non apprende.

Chi sono e cosa fanno

Molti prodotti realizzano la predizione solo a livello lessicale (cioé sulla base di un dizionario di frequenza d'uso personalizzato)

Prodotti

- *SofType*: aggiorna le frequenze ma non c'è controllo sintattico;
- *C.A.R.L.O.*: non c'è controllo sintattico;
- *Dedalus*: effettua il controllo sintattico ma non considera le freq. ed inoltre il programma non apprende.

Chi sono e cosa fanno

Molti prodotti realizzano la predizione solo a livello lessicale (cioé sulla base di un dizionario di frequenza d'uso personalizzato)

Prodotti

- *SofType*: aggiorna le frequenze ma non c'è controllo sintattico;
- *C.A.R.L.O.*: non c'è controllo sintattico;
- *Dedalus*: effettua il controllo sintattico ma non considera le freq. ed inoltre il programma non apprende.

Chi sono e cosa fanno

Molti prodotti realizzano la predizione solo a livello lessicale (cioé sulla base di un dizionario di frequenza d'uso personalizzato)

Prodotti

- *SofType*: aggiorna le frequenze ma non c'è controllo sintattico;
- *C.A.R.L.O.*: non c'è controllo sintattico;
- *Dedalus*: effettua il controllo sintattico ma non considera le freq. ed inoltre il programma non apprende.

Cosa si richiede che faccia

Si vuole allora creare un algoritmo di efficace di scrittura veloce che:

Soluzioni

- effettua la *predizione delle parole* interrogando un ampio dizionario generale e controllando, tramite una grammatica, le concordanze sintattiche;
- permette la *visualizzazione* delle parole candidate e la rapida scelta tra esse di quella desiderata;
- sia realizzato con un approccio basato sull'*autoapprendimento* al fine di adeguare l'applicazione sia al lessico che al personale stile di composizione dell'utente.

Cosa si richiede che faccia

Si vuole allora creare un algoritmo di efficace di scrittura veloce che:

Soluzioni

- effettua la *predizione delle parole* interrogando un ampio dizionario generale e controllando, tramite una grammatica, le concordanze sintattiche;
- permette la *visualizzazione* delle parole candidate e la rapida scelta tra esse di quella desiderata;
- sia realizzato con un approccio basato sull'*autoapprendimento* al fine di adeguare l'applicazione sia al lessico che al personale stile di composizione dell'utente.

Cosa si richiede che faccia

Si vuole allora creare un algoritmo di efficace di scrittura veloce che:

Soluzioni

- effettua la *predizione delle parole* interrogando un ampio dizionario generale e controllando, tramite una grammatica, le concordanze sintattiche;
- permette la *visualizzazione* delle parole candidate e la rapida scelta tra esse di quella desiderata;
- sia realizzato con un approccio basato sull'*autoapprendimento* al fine di adeguare l'applicazione sia al lessico che al personale stile di composizione dell'utente.

Cosa si richiede che faccia

Si vuole allora creare un algoritmo di efficace di scrittura veloce che:

Soluzioni

- effettua la *predizione delle parole* interrogando un ampio dizionario generale e controllando, tramite una grammatica, le concordanze sintattiche;
- permette la *visualizzazione* delle parole candidate e la rapida scelta tra esse di quella desiderata;
- sia realizzato con un approccio basato sull'*autoapprendimento* al fine di adeguare l'applicazione sia al lessico che al personale stile di composizione dell'utente.

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

- il gestore dei dizionari;
- il parser della frase basato sui file di linguaggio.

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

- il gestore dei dizionari;
- il parser della frase basato sui file di linguaggio.

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

- il gestore dei dizionari;
- il parser della frase basato sui file di linguaggio.

Di cosa necessita

Per il trattamento del linguaggio naturale c'è bisogno di:

- una *base di dati lessicali*;
- un *parser*;
- un *lemmatizzatore*.
- (un editor per le *grammatiche* e per i *dizionari*).

Il software per la gestione delle risorse linguistiche può essere visto come composto da due moduli principali:

Moduli

- il gestore dei dizionari;
- il parser della frase basato sui file di linguaggio.

Tipologie

I dizionari possono essere di tipo:

- *generativo*: memorizzano solo alcune delle forme ammesse per le parole del lessico (in genere solo i lemmi) ⇒ nella ricerca, a partire dalla forma di una parola, si tenta di risalire al lemma tramite le regole di flessione ed alterazione;
- *non-generativo*: sono caratterizzati da una fase di generazione fatta una tantum che produce tutte e sole le forme ammesse in un lessico ⇒ nella ricerca non devono attraversare una fase di elaborazione (il dizionario è completo e occupa più spazio).

Si adotta qui il modello **non-generativo**.

Tipologie

I dizionari possono essere di tipo:

- *generativo*: memorizzano solo alcune delle forme ammesse per le parole del lessico (in genere solo i lemmi) ⇒ nella ricerca, a partire dalla forma di una parola, si tenta di risalire al lemma tramite le regole di flessione ed alterazione;
- *non-generativo*: sono caratterizzati da una fase di generazione fatta una tantum che produce tutte e sole le forme ammesse in un lessico ⇒ nella ricerca non devono attraversare una fase di elaborazione (il dizionario è completo e occupa più spazio).

Si adotta qui il modello **non-generativo**.

Tipologie

I dizionari possono essere di tipo:

- *generativo*: memorizzano solo alcune delle forme ammesse per le parole del lessico (in genere solo i lemmi) ⇒ nella ricerca, a partire dalla forma di una parola, si tenta di risalire al lemma tramite le regole di flessione ed alterazione;
- *non-generativo*: sono caratterizzati da una fase di generazione fatta una tantum che produce tutte e sole le forme ammesse in un lessico ⇒ nella ricerca non devono attraversare una fase di elaborazione (il dizionario è completo e occupa più spazio).

Si adotta qui il modello **non-generativo**.

Radici e suffissi

Le forme di un lemma tendono a variare solo[1]nella parte terminale ⇒ si considerano le parole come formate di due parti:

- *radice*:
- *suffisso*:

---

[1] Questo vale almeno per le lingue indoeuropee.

Radici e suffissi

Le forme di un lemma tendono a variare solo[1]nella parte terminale ⇒ si considerano le parole come formate di due parti:

- *radice*:
    - parte iniziale (sinistra) della parola;
    - di lunghezza variabile;
    - collegata al significato semantico;
- *suffisso*:
    - parte finale (destra) della parola;
    - fissate a 5 caratteri la lunghezza massima e a 3 quella media;
    - identifica gli attributi morfologici e grammaticali.

---

[1] Questo vale almeno per le lingue indoeuropee.

Radici e suffissi

Le forme di un lemma tendono a variare solo[1]nella parte terminale ⇒ si considerano le parole come formate di due parti:

- *radice*:
    - parte iniziale (sinistra) della parola;
    - di lunghezza variabile;
    - collegata al significato semantico;
- *suffisso*:
    - parte finale (destra) della parola;
    - fissate a 5 caratteri la lunghezza massima e a 3 quella media;
    - identifica gli attributi morfologici e grammaticali.

---

[1] Questo vale almeno per le lingue indoeuropee.

Radici e suffissi

Le forme di un lemma tendono a variare solo[1] nella parte terminale ⇒ si considerano le parole come formate di due parti:

- *radice*:
  - parte iniziale (sinistra) della parola;
  - di lunghezza variabile;
  - collegata al significato semantico;
- *suffisso*:
  - parte finale (destra) della parola;
  - fissate a 5 caratteri la lunghezza massima e a 3 quella media;
  - identifica gli attributi morfologici e grammaticali.

---

[1] Questo vale almeno per le lingue indoeuropee.

Radici e suffissi

Le forme di un lemma tendono a variare solo[1] nella parte terminale ⇒ si considerano le parole come formate di due parti:

- *radice*:
  - parte iniziale (sinistra) della parola;
  - di lunghezza variabile;
  - collegata al significato semantico;
- *suffisso*:
  - parte finale (destra) della parola;
  - fissate a 5 caratteri la lunghezza massima e a 3 quella media;
  - identifica gli attributi morfologici e grammaticali.

---

[1] Questo vale almeno per le lingue indoeuropee.

Struttura logica TWL

Nella rappresentazione in memoria le parole possono condividere la sottostringa corrispondente alla *radice*, oppure possono condividere quella corrispondente al *suffisso* ⇒ si adotta una *doppia struttura ad albero* denominata TWL (Tree Word List) per cui:

- ogni parola è *univocamente definita* dal legame tra una radice e un suffisso;
- l'*albero delle radici* (Lexicon TWL) e l'*albero dei suffissi* (Language TWL) sono separatamente memorizzati (rispettivamente negli spazi del dizionario e del linguaggio);
- il TWL di tutte le forme risulta *estremamente compatto*: il dizionario italiano completo (circa 43.000 lemmi e 800.000 forme) occupa meno di 14 Mb.

Struttura logica TWL

Nella rappresentazione in memoria le parole possono condividere la sottostringa corrispondente alla *radice*, oppure possono condividere quella corrispondente al *suffisso* ⇒ si adotta una *doppia struttura ad albero* denominata TWL (Tree Word List) per cui:

- ogni parola è *univocamente definita* dal legame tra una radice e un suffisso;
- l'*albero delle radici* (Lexicon TWL) e l'*albero dei suffissi* (Language TWL) sono separatamente memorizzati (rispettivamente negli spazi del dizionario e del linguaggio);
- il TWL di tutte le forme risulta *estremamente compatto*: il dizionario italiano completo (circa 43.000 lemmi e 800.000 forme) occupa meno di 14 Mb.

Struttura logica TWL

Nella rappresentazione in memoria le parole possono condividere la sottostringa corrispondente alla *radice*, oppure possono condividere quella corrispondente al *suffisso* ⇒ si adotta una *doppia struttura ad albero* denominata TWL (Tree Word List) per cui:

- ogni parola è *univocamente definita* dal legame tra una radice e un suffisso;
- l'*albero delle radici* (Lexicon TWL) e l'*albero dei suffissi* (Language TWL) sono separatamente memorizzati (rispettivamente negli spazi del dizionario e del linguaggio);
- il TWL di tutte le forme risulta *estremamente compatto*: il dizionario italiano completo (circa 43.000 lemmi e 800.000 forme) occupa meno di 14 Mb.

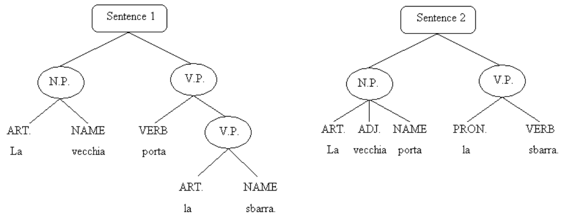
Struttura logica TWL

Nella rappresentazione in memoria le parole possono condividere la sottostringa corrispondente alla *radice*, oppure possono condividere quella corrispondente al *suffisso* ⇒ si adotta una *doppia struttura ad albero* denominata TWL (Tree Word List) per cui:

- ogni parola è *univocamente definita* dal legame tra una radice e un suffisso;
- l'*albero delle radici* (Lexicon TWL) e l'*albero dei suffissi* (Language TWL) sono separatamente memorizzati (rispettivamente negli spazi del dizionario e del linguaggio);
- il TWL di tutte le forme risulta *estremamente compatto*: il dizionario italiano completo (circa 43.000 lemmi e 800.000 forme) occupa meno di 14 Mb.

Ambiguità

Gli *alberi di derivazione sintattica* sono formati da: nodi interni, ovvero i *sintagmi*, archi, ovvero l'applicazione di *regole*, foglie, ovvero le *parole*.

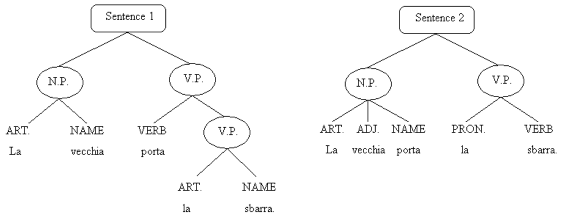


Il parser compie essenzialmente due operazioni:

- *riconoscimento* di una stringa come una *sentence* del linguaggio
- *generazione dell'albero sintattico* assegnata alla *sentence*

Ambiguità

Gli *alberi di derivazione sintattica* sono formati da: nodi interni, ovvero i *sintagmi*, archi, ovvero l'applicazione di *regole*, foglie, ovvero le *parole*.

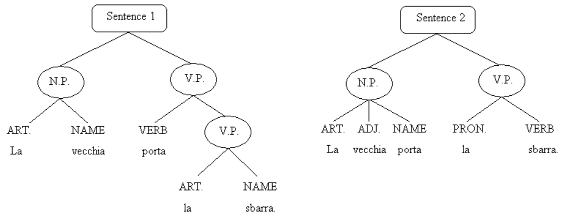


Il parser compie essenzialmente due operazioni:

- *riconoscimento* di una stringa come una *sentence* del linguaggio
- *generazione dell'albero sintattico* assegnata alla *sentence*

Ambiguità

Gli *alberi di derivazione sintattica* sono formati da: nodi interni, ovvero i *sintagmi*, archi, ovvero l'applicazione di *regole*, foglie, ovvero le *parole*.



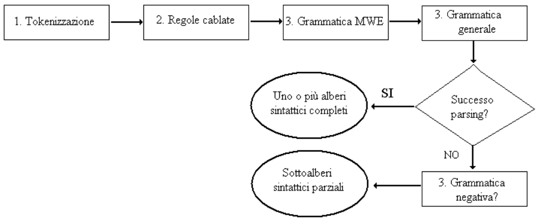
Il parser compie essenzialmente due operazioni:

- *riconoscimento* di una stringa come una *sentence* del linguaggio
- *generazione dell'albero sintattico* assegnata alla *sentence*

Passi principali

Fondamentalmente nella WP il parsing "filtra" i possibili suggerimenti riconoscendo la **POS** (**P**art **O**f **S**peech) delle parole

(così da collocarle correttamente all'interno delle frasi, riducendo le ambiguità).



1. Tokenizzazione

2. Regole cablate

3. Grammatica MWE

3. Grammatica generale

Successo parsing?

SI

Uno o più alberi sintattici completi

NO

3. Grammatica negativa?

Sottoalberi sintattici parziali

Passi principali

- Tokenizzazione separazione e classificazione di ogni parola ⇒ catena di token formata da tutte le clasificazioni (anche multiple);
- Regole cablate (scritte nel linguaggio di programmazione)si opera un primo raffinamento della catena di token;
- Grammatica MWE tramite la grammatica del dizionario si individuano le espressioni polilessicali;
- Grammatica generale (del linguaggio) eliminazione delle classificazioni multiple cercando di riportare tutta le foglie ad un'unica radice;
- Grammatica negativa? produzione di nuovi token relativi agli elementi ritenuti discordanti.

Passi principali

- Tokenizzazione separazione e classificazione di ogni parola ⇒ catena di token formata da tutte le clasificazioni (anche multiple);
- Regole cablate (scritte nel linguaggio di programmazione)si opera un primo raffinamento della catena di token;
- Grammatica MWE tramite la grammatica del dizionario si individuano le espressioni polilessicali;
- Grammatica generale (del linguaggio) eliminazione delle classificazioni multiple cercando di riportare tutta le foglie ad un'unica radice;
- Grammatica negativa? produzione di nuovi token relativi agli elementi ritenuti discordanti.

Passi principali

- Tokenizzazione separazione e classificazione di ogni parola ⇒ catena di token formata da tutte le clasificazioni (anche multiple);
- Regole cablate (scritte nel linguaggio di programmazione)si opera un primo raffinamento della catena di token;
- Grammatica MWE tramite la grammatica del dizionario si individuano le espressioni polilessicali;
- Grammatica generale (del linguaggio) eliminazione delle classificazioni multiple cercando di riportare tutta le foglie ad un'unica radice;
- Grammatica negativa? produzione di nuovi token relativi agli elementi ritenuti discordanti.

Passi principali

- Tokenizzazione separazione e classificazione di ogni parola ⇒ catena di token formata da tutte le clasificazioni (anche multiple);
- Regole cablate (scritte nel linguaggio di programmazione)si opera un primo raffinamento della catena di token;
- Grammatica MWE tramite la grammatica del dizionario si individuano le espressioni polilessicali;
- Grammatica generale (del linguaggio) eliminazione delle classificazioni multiple cercando di riportare tutta le foglie ad un'unica radice;
- Grammatica negativa? produzione di nuovi token relativi agli elementi ritenuti discordanti.

Passi principali

- Tokenizzazione separazione e classificazione di ogni parola ⇒ catena di token formata da tutte le clasificazioni (anche multiple);
- Regole cablate (scritte nel linguaggio di programmazione)si opera un primo raffinamento della catena di token;
- Grammatica MWE tramite la grammatica del dizionario si individuano le espressioni polilessicali;
- Grammatica generale (del linguaggio) eliminazione delle classificazioni multiple cercando di riportare tutta le foglie ad un'unica radice;
- Grammatica negativa? produzione di nuovi token relativi agli elementi ritenuti discordanti.

Passi principali

- Tokenizzazione separazione e classificazione di ogni parola ⇒ catena di token formata da tutte le clasificazioni (anche multiple);
- Regole cablate (scritte nel linguaggio di programmazione)si opera un primo raffinamento della catena di token;
- Grammatica MWE tramite la grammatica del dizionario si individuano le espressioni polilessicali;
- Grammatica generale (del linguaggio) eliminazione delle classificazioni multiple cercando di riportare tutta le foglie ad un'unica radice;
- Grammatica negativa? produzione di nuovi token relativi agli elementi ritenuti discordanti.

Lemmatizzazione

Nei testi (in NL) si trovano le *forme superficiali* ⇒ si vogliono riportare le diverse forme sotto un unico lemma. Per lemmatizzare partendo dalle foglie dell'albero sintattico[2], ad ogni forma flessa si associa la coppia

[(sestupla),(lemma di origine)]

Nella sestupla, dal primo elemento (POS) dipendono gli ultimi *due elementi*, i quali:

- per **sostantivi**, **agg. possessivi** e **pron. possessivi** esprimono info. relative al lemma di origine;
- per i **pron. personali** il 5° elemento indica la funzione sintattica, mentre il 6° è nullo;
- per i **verbi** il 5° e il 6° elemento indicano modo e tempo della forma verbale;
- per le **altre POS** essi sono nulli.

---

[2] Che è il risultato del parser.

Lemmatizzazione

Nei testi (in NL) si trovano le *forme superficiali* ⇒ si vogliono riportare le diverse forme sotto un unico lemma. Per lemmatizzare partendo dalle foglie dell'albero sintattico[2], ad ogni forma flessa si associa la coppia

[(sestupla),(lemma di origine)]

Nella sestupla, dal primo elemento (POS) dipendono gli ultimi *due elementi*, i quali:

- per **sostantivi**, **agg. possessivi** e **pron. possessivi** esprimono info. relative al lemma di origine;
- per i **pron. personali** il 5° elemento indica la funzione sintattica, mentre il 6° è nullo;
- per i **verbi** il 5° e il 6° elemento indicano modo e tempo della forma verbale;
- per le **altre POS** essi sono nulli.

---

[2] Che è il risultato del parser.

Lemmatizzazione

Nei testi (in NL) si trovano le *forme superficiali* ⇒ si vogliono riportare le diverse forme sotto un unico lemma. Per lemmatizzare partendo dalle foglie dell'albero sintattico[2], ad ogni forma flessa si associa la coppia

[(sestupla),(lemma di origine)]

Nella sestupla, dal primo elemento (POS) dipendono gli ultimi *due elementi*, i quali:

- per **sostantivi**, **agg. possessivi** e **pron. possessivi** esprimono info. relative al lemma di origine;
- per i **pron. personali** il 5° elemento indica la funzione sintattica, mentre il 6° è nullo;
- per i **verbi** il 5° e il 6° elemento indicano modo e tempo della forma verbale;
- per le **altre POS** essi sono nulli.

---

[2] Che è il risultato del parser.

Lemmatizzazione

Nei testi (in NL) si trovano le *forme superficiali* ⇒ si vogliono riportare le diverse forme sotto un unico lemma. Per lemmatizzare partendo dalle foglie dell'albero sintattico[2], ad ogni forma flessa si associa la coppia

[(sestupla),(lemma di origine)]

Nella sestupla, dal primo elemento (POS) dipendono gli ultimi *due elementi*, i quali:

- per **sostantivi**, **agg. possessivi** e **pron. possessivi** esprimono info. relative al lemma di origine;
- per i **pron. personali** il 5° elemento indica la funzione sintattica, mentre il 6° è nullo;
- per i **verbi** il 5° e il 6° elemento indicano modo e tempo della forma verbale;
- per le **altre POS** essi sono nulli.

---

[2] Che è il risultato del parser.

Lemmatizzazione

Nei testi (in NL) si trovano le *forme superficiali* ⇒ si vogliono riportare le diverse forme sotto un unico lemma. Per lemmatizzare partendo dalle foglie dell'albero sintattico[2], ad ogni forma flessa si associa la coppia

[(sestupla),(lemma di origine)]

Nella sestupla, dal primo elemento (POS) dipendono gli ultimi *due elementi*, i quali:

- per **sostantivi**, **agg. possessivi** e **pron. possessivi** esprimono info. relative al lemma di origine;
- per i **pron. personali** il 5° elemento indica la funzione sintattica, mentre il 6° è nullo;
- per i **verbi** il 5° e il 6° elemento indicano modo e tempo della forma verbale;
- per le **altre POS** essi sono nulli.

---

[2] Che è il risultato del parser.

Esempi

- *ragazza*: [(S C F S M S),(ragazzo)]

sostantivo (S), comune (C), femminile (F), singolare (S); lemma maschile (M), singolare (S)

- *è* : [(V E N 3 I N),(essere)]

verbo (V), ausiliare essere (E), neutro (N), terza pers. singolare (3), indicativo (I), presente (N)

verbo (V), intrans. con ausiliare essere (F), maschile (M), singolare (S), participio (P), passato (P)

Esempi

- *ragazza*: [(S C F S M S),(ragazzo)]

sostantivo (S), comune (C), femminile (F), singolare (S); lemma maschile (M), singolare (S)

- *è* : [(V E N 3 I N),(essere)]

verbo (V), ausiliare essere (E), neutro (N), terza pers. singolare (3), indicativo (I), presente (N)

- *arrivato*: [(V F M S P P),(arrivare)]

verbo (V), intrans. con ausiliare essere (F), maschile (M), singolare (S), participio (P), passato (P)

Esempi

- *ragazza*: [(S C F S M S),(ragazzo)]

sostantivo (S), comune (C), femminile (F), singolare (S); lemma maschile (M), singolare (S)

- *è* : [(V E N 3 I N),(essere)]

verbo (V), ausiliare essere (E), neutro (N), terza pers. singolare (3), indicativo (I), presente (N)

- *arrivato*: [(V F M S P P),(arrivare)]

verbo (V), intrans. con ausiliare essere (F), maschile (M), singolare (S), participio (P), passato (P)

Esempi

- *ragazza*: [(S C F S M S),(ragazzo)]

sostantivo (S), comune (C), femminile (F), singolare (S); lemma maschile (M), singolare (S)

- *è* : [(V E N 3 I N),(essere)]

verbo (V), ausiliare essere (E), neutro (N), terza pers. singolare (3), indicativo (I), presente (N)

- *arrivato*: [(V F M S P P),(arrivare)]

verbo (V), intrans. con ausiliare essere (F), maschile (M), singolare (S), participio (P), passato (P)

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

- [(io),(mangio),(la)]. . . mela

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

- [(io),(mangio),(la)]. . . mela
- [(noi),(soffochiamo),(gli)]. . . istinti

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

- [(io),(mangio),(la)]. . . mela
- [(noi),(soffochiamo),(gli)]. . . istinti

Come e perché le sestuple

Lemmatizzazione ⇒ ricerca dei costrutti più frequeni considerando la vicinanza degli *oggetti lessicali* tramite le *triple di sestuple*

[(sestupla 1),(sestupla 2),(sestupla 3)]

e attribuendo un valore statistico in base alla *frequenza* della tripletta.

Esempi di costruzioni molto probabili

- [(verbo intransitivo),(preposizione),(nome)]

tipicamente seguito da un complemento indir. introdotto da prep., come:

- [(dormire),(sul),(letto)]. . . di legno
- [(mangiare),(dai),(parenti)]. . . a Pisa

- [(soggetto),(verbo transitivo),(articolo)]

tipicamente seguito da un complemento oggetto o dir., come:

- [(io),(mangio),(la)]. . . mela
- [(noi),(soffochiamo),(gli)]. . . istinti

Gli n-gram

Per predizione la parola successiva ω_n si può stimare la probabilità che essa segua alle $n-1$ parole precedenti $\omega_1, \omega_2, \dots, \omega_{n-1}$, cioé

$$\mathbb{P}(\omega_n|\omega_1\omega_2\dots\omega_{n-1})$$

Bisogna raggruppare le diverse classificazioni delle parole precedenti (*history*) ⇒ **assunzione di Markov**: sulla predizione influiscono solo le ultime parole. Definendo

successione di *variabili aleatorie* $X = (X_1, \dots, X_T)$

i cui valori appartengono a

insieme finito $S = \{s_1, \dots, s_N\}$

le proprietà dei *modelli di Markov* sono

- **orizzonte limitato**: $\mathbb{P}(X_t+1 = s_k|X_1, \dots, X_t) = \mathbb{P}(X_t+1 = s_k|X_t)$
- **stazionarietà**: $\mathbb{P}(X_t+1 = s_k|X_t) = \mathbb{P}(X_2 = s_k|X_1)$

Gli n-gram

Per predizione la parola successiva ω_n si può stimare la probabilità che essa segua alle $n-1$ parole precedenti $\omega_1, \omega_2, \dots, \omega_{n-1}$, cioé

$$\mathbb{P}(\omega_n|\omega_1\omega_2\dots\omega_{n-1})$$

Bisogna raggruppare le diverse classificazioni delle parole precedenti (*history*) ⇒ **assunzione di Markov**: sulla predizione influiscono solo le ultime parole. Definendo

successione di *variabili aleatorie* $X = (X_1, \dots, X_T)$

i cui valori appartengono a

insieme finito $S = \{s_1, \dots, s_N\}$

le proprietà dei *modelli di Markov* sono

- **orizzonte limitato**: $\mathbb{P}(X_t+1 = s_k|X_1, \dots, X_t) = \mathbb{P}(X_t+1 = s_k|X_t)$
- **stazionarietà**: $\mathbb{P}(X_t+1 = s_k|X_t) = \mathbb{P}(X_2 = s_k|X_1)$

Gli n-gram

Per predizione la parola successiva ω_n si può stimare la probabilità che essa segua alle $n-1$ parole precedenti $\omega_1, \omega_2, \dots, \omega_{n-1}$, cioé

$$\mathbb{P}(\omega_n|\omega_1\omega_2\dots\omega_{n-1})$$

Bisogna raggruppare le diverse classificazioni delle parole precedenti (*history*) ⇒ **assunzione di Markov**: sulla predizione influiscono solo le ultime parole. Definendo

successione di *variabili aleatorie* $X = (X_1, \dots, X_T)$

i cui valori appartengono a

insieme finito $S = \{s_1, \dots, s_N\}$

le proprietà dei *modelli di Markov* sono

- **orizzonte limitato**: $\mathbb{P}(X_t+1 = s_k|X_1, \dots, X_t) = \mathbb{P}(X_t+1 = s_k|X_t)$
- **stazionarietà**: $\mathbb{P}(X_t+1 = s_k|X_t) = \mathbb{P}(X_2 = s_k|X_1)$

Le catene di Markov

Se la successione X soddisfa tali proprietà si chiama *catena di Markov*, e la probabilità per essa diviene

$$\mathbb{P}(X_1, \dots, X_T) = \mathbb{P}(X_1)\mathbb{P}(X_2|X_1)\mathbb{P}(X_3|X_2)\dots\mathbb{P}(X_T|X_{T-1})$$

Il WP usa *n-gram* di **tag** (sestuple) al posto di catene di parole (COL PROBLEMA DI PERDITA DI SEMANTICA), per cui la ricerca si sposta su

- la determinazione del tag che ha più probabilità di seguire gli altri,
- la probabilità che una parola venga emessa con un particolare tag.

Le catene di Markov

Se la successione X soddisfa tali proprietà si chiama *catena di Markov*, e la probabilità per essa diviene

$$\mathbb{P}(X_1, \dots, X_T) = \mathbb{P}(X_1)\mathbb{P}(X_2|X_1)\mathbb{P}(X_3|X_2)\dots\mathbb{P}(X_T|X_{T-1})$$

Il WP usa *n-gram* di **tag** (sestuple) al posto di catene di parole (COL PROBLEMA DI PERDITA DI SEMANTICA), per cui la ricerca si sposta su

- la determinazione del tag che ha più probabilità di seguire gli altri,
- la probabilità che una parola venga emessa con un particolare tag.

Le catene di Markov

Se la successione X soddisfa tali proprietà si chiama *catena di Markov*, e la probabilità per essa diviene

$$\mathbb{P}(X_1, \dots, X_T) = \mathbb{P}(X_1)\mathbb{P}(X_2|X_1)\mathbb{P}(X_3|X_2)\dots\mathbb{P}(X_T|X_{T-1})$$

Il WP usa *n-gram* di **tag** (sestuple) al posto di catene di parole (COL PROBLEMA DI PERDITA DI SEMANTICA), per cui la ricerca si sposta su

- la determinazione del tag che ha più probabilità di seguire gli altri,
- la probabilità che una parola venga emessa con un particolare tag.

Le catene di tag

Notazione

ω_i	la parola alla posizione i nel testo (da cui $\omega_{i,k} = \omega_i, \dots, \omega_k$)
t_i	il tag relativo a ω_i (da cui $t_{i,k} = t_i, \dots, t_k$)
$C(t_j)$	il numero di occorrenze di t_j nel *training set*
$C(t_j, t_k)$	il numero di occorrenze di t_j seguito da t_k nel *training set*
$C(\omega_l, t_j)$	il numero di occorrenze di ω_l classificata come t_j nel *training set*

- orizzonte limitato: $\mathbb{P}(t_{i+1}|t_{1,i}) = \mathbb{P}(t_{i+1}|t_i)$
- probabilità *contestuale*: $\mathbb{P}(t_k|t_j) = \frac{C(t_j,t_k)}{C(t_j)}$
- probabilità *locale*: $\mathbb{P}(\omega_l|t_j) = \frac{C(\omega_l,t_k)}{C(t_j)}$
- data la proprietà O e dato l'evento x, la probabilità che x sia descritto da O è $\mathbb{P}(x|O) \Rightarrow$ *regola di Bayes*: $\mathbb{P}(x|O) = \frac{\mathbb{P}(O|x)\mathbb{P}(x)}{\mathbb{P}(O)}$
- $arg\ max_{\{x\}}\ [\mathbb{P}(x|O)] = arg\ max_{\{x\}}\ [\mathbb{P}(O|x)\mathbb{P}(x)]$

Le catene di tag

Notazione

ω_i	la parola alla posizione i nel testo (da cui $\omega_{i,k} = \omega_i, \dots, \omega_k$)
t_i	il tag relativo a ω_i (da cui $t_{i,k} = t_i, \dots, t_k$)
$C(t_j)$	il numero di occorrenze di t_j nel *training set*
$C(t_j, t_k)$	il numero di occorrenze di t_j seguito da t_k nel *training set*
$C(\omega_l, t_j)$	il numero di occorrenze di ω_l classificata come t_j nel *training set*

- orizzonte limitato: $\mathbb{P}(t_{i+1}|t_{1,i}) = \mathbb{P}(t_{i+1}|t_i)$
- probabilità *contestuale*: $\mathbb{P}(t_k|t_j) = \frac{C(t_j,t_k)}{C(t_j)}$
- probabilità *locale*: $\mathbb{P}(\omega_l|t_j) = \frac{C(\omega_l,t_k)}{C(t_j)}$
- data la proprietà O e dato l'evento x, la probabilità che x sia descritto da O è $\mathbb{P}(x|O) \Rightarrow$ *regola di Bayes*: $\mathbb{P}(x|O) = \frac{\mathbb{P}(O|x)\mathbb{P}(x)}{\mathbb{P}(O)}$
- $\mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(x|O)] = \mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(O|x)\mathbb{P}(x)]$

Le catene di tag

Notazione

ω_i	la parola alla posizione i nel testo (da cui $\omega_{i,k} = \omega_i, \dots, \omega_k$)
t_i	il tag relativo a ω_i (da cui $t_{i,k} = t_i, \dots, t_k$)
$C(t_j)$	il numero di occorrenze di t_j nel *training set*
$C(t_j, t_k)$	il numero di occorrenze di t_j seguito da t_k nel *training set*
$C(\omega_l, t_j)$	il numero di occorrenze di ω_l classificata come t_j nel *training set*

- orizzonte limitato: $\mathbb{P}(t_{i+1}|t_{1,i}) = \mathbb{P}(t_{i+1}|t_i)$
- probabilità *contestuale*: $\mathbb{P}(t_k|t_j) = \frac{C(t_j,t_k)}{C(t_j)}$
- probabilità *locale*: $\mathbb{P}(\omega_l|t_j) = \frac{C(\omega_l,t_k)}{C(t_j)}$
- data la proprietà O e dato l'evento x, la probabilità che x sia descritto da O è $\mathbb{P}(x|O) \Rightarrow$ *regola di Bayes*: $\mathbb{P}(x|O) = \frac{\mathbb{P}(O|x)\mathbb{P}(x)}{\mathbb{P}(O)}$
- $\mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(x|O)] = \mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(O|x)\mathbb{P}(x)]$

Le catene di tag

Notazione

ω_i	la parola alla posizione i nel testo (da cui $\omega_{i,k} = \omega_i, \dots, \omega_k$)
t_i	il tag relativo a ω_i (da cui $t_{i,k} = t_i, \dots, t_k$)
$C(t_j)$	il numero di occorrenze di t_j nel *training set*
$C(t_j, t_k)$	il numero di occorrenze di t_j seguito da t_k nel *training set*
$C(\omega_l, t_j)$	il numero di occorrenze di ω_l classificata come t_j nel *training set*

- orizzonte limitato: $\mathbb{P}(t_{i+1}|t_{1,i}) = \mathbb{P}(t_{i+1}|t_i)$
- probabilità *contestuale*: $\mathbb{P}(t_k|t_j) = \frac{C(t_j,t_k)}{C(t_j)}$
- probabilità *locale*: $\mathbb{P}(\omega_l|t_j) = \frac{C(\omega_l,t_k)}{C(t_j)}$
- data la proprietà O e dato l'evento x, la probabilità che x sia descritto da O è $\mathbb{P}(x|O) \Rightarrow$ *regola di Bayes*: $\mathbb{P}(x|O) = \frac{\mathbb{P}(O|x)\mathbb{P}(x)}{\mathbb{P}(O)}$
- $\mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(x|O)] = \mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(O|x)\mathbb{P}(x)]$

Le catene di tag

Notazione

ω_i	la parola alla posizione i nel testo (da cui $\omega_{i,k} = \omega_i, \dots, \omega_k$)
t_i	il tag relativo a ω_i (da cui $t_{i,k} = t_i, \dots, t_k$)
$C(t_j)$	il numero di occorrenze di t_j nel *training set*
$C(t_j, t_k)$	il numero di occorrenze di t_j seguito da t_k nel *training set*
$C(\omega_l, t_j)$	il numero di occorrenze di ω_l classificata come t_j nel *training set*

- orizzonte limitato: $\mathbb{P}(t_{i+1}|t_{1,i}) = \mathbb{P}(t_{i+1}|t_i)$
- probabilità *contestuale*: $\mathbb{P}(t_k|t_j) = \frac{C(t_j,t_k)}{C(t_j)}$
- probabilità *locale*: $\mathbb{P}(\omega_l|t_j) = \frac{C(\omega_l,t_k)}{C(t_j)}$
- data la proprietà O e dato l'evento x, la probabilità che x sia descritto da O è $\mathbb{P}(x|O) \Rightarrow$ *regola di Bayes*: $\mathbb{P}(x|O) = \frac{\mathbb{P}(O|x)\mathbb{P}(x)}{\mathbb{P}(O)}$
- $\mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(x|O)] = \mathit{arg\ max}_{\{x\}}\ [\mathbb{P}(O|x)\mathbb{P}(x)]$

Le catene di tag

Notazione

ω_i	la parola alla posizione *i* nel testo (da cui $\omega_{i,k} = \omega_i, \dots, \omega_k$)
t_i	il tag relativo a ω_i (da cui $t_{i,k} = t_i, \dots, t_k$)
$C(t_j)$	il numero di occorrenze di t_j nel *training set*
$C(t_j, t_k)$	il numero di occorrenze di t_j seguito da t_k nel *training set*
$C(\omega_l, t_j)$	il numero di occorrenze di ω_l classificata come t_j nel *training set*

- orizzonte limitato: $\mathbb{P}(t_{i+1}|t_{1,i}) = \mathbb{P}(t_{i+1}|t_i)$
- probabilità *contestuale*: $\mathbb{P}(t_k|t_j) = \frac{C(t_j,t_k)}{C(t_j)}$
- probabilità *locale*: $\mathbb{P}(\omega_l|t_j) = \frac{C(\omega_l,t_k)}{C(t_j)}$
- data la proprietà O e dato l'evento x, la probabilità che x sia descritto da O è $\mathbb{P}(x|O)$ ⇒ *regola di Bayes*: $\mathbb{P}(x|O) = \frac{\mathbb{P}(O|x)\mathbb{P}(x)}{\mathbb{P}(O)}$
- $arg\ max_{\{x\}}\left[\mathbb{P}(x|O)\right] = arg\ max_{\{x\}}\left[\mathbb{P}(O|x)\mathbb{P}(x)\right]$

Le catene di tag

⇒ la miglior sequenza di tag $\widehat{t}_{1,n}$ che approssimi la frase $\omega_1, \dots \omega_n$ è:

$$\widehat{t}_{1,n} = arg\ max_{\{t_{1,n}\}}\left[\mathbb{P}(t_{1,n}|\omega_{1,n})\right] = arg\ max_{\{t_{1,n}\}}\left[\prod_{i=1}^{n}\mathbb{P}(\omega_i|t_i)\mathbb{P}(t_i|t_{i-1})\right]$$

Dunque nella word prediction

- si possono "`filtrare`" tutte le parole suggerite e mostrare solo le più probabili secondo la grammatica statistica dell'*n*-gram[a]
- il sistema può "`apprendere`" aggiornando le occorrenze usate nel calcolo delle probabilità.
- c'è perdita di informazione "`semantica`", dovuta al fatto che si considerano le POS anziché le parole distinte.

---

[a] Tipicamente non si supera il valore n=4, altrimenti l'algoritmo richiederebbe un tempo eccessivo (nelle tesi è n=3).

Le catene di tag

⇒ la miglior sequenza di tag $\widehat{t}_{1,n}$ che approssimi la frase $\omega_1, \dots \omega_n$ è:

$$\widehat{t}_{1,n} = arg\ max_{\{t_{1,n}\}}\left[\mathbb{P}(t_{1,n}|\omega_{1,n})\right] = arg\ max_{\{t_{1,n}\}}\left[\prod_{i=1}^{n}\mathbb{P}(\omega_i|t_i)\mathbb{P}(t_i|t_{i-1})\right]$$

Dunque nella word prediction

- si possono "`filtrare`" tutte le parole suggerite e mostrare solo le più probabili secondo la grammatica statistica dell'*n*-gram[a]
- il sistema può "`apprendere`" aggiornando le occorrenze usate nel calcolo delle probabilità.
- c'è perdita di informazione "`semantica`", dovuta al fatto che si considerano le POS anziché le parole distinte.

[a] Tipicamente non si supera il valore n=4, altrimenti l'algoritmo richiederebbe un tempo eccessivo (nelle tesi è n=3).

Le catene di tag

⇒ la miglior sequenza di tag $\widehat{t}_{1,n}$ che approssimi la frase $\omega_1, \dots \omega_n$ è:

$$\widehat{t}_{1,n} = arg\ max_{\{t_{1,n}\}}\left[\mathbb{P}(t_{1,n}|\omega_{1,n})\right] = arg\ max_{\{t_{1,n}\}}\left[\prod_{i=1}^{n}\mathbb{P}(\omega_i|t_i)\mathbb{P}(t_i|t_{i-1})\right]$$

Dunque nella word prediction

- si possono "`filtrare`" tutte le parole suggerite e mostrare solo le più probabili secondo la grammatica statistica dell'*n*-gram[a]
- il sistema può "`apprendere`" aggiornando le occorrenze usate nel calcolo delle probabilità.
- c'è perdita di informazione "`semantica`", dovuta al fatto che si considerano le POS anziché le parole distinte.

[a] Tipicamente non si supera il valore n=4, altrimenti l'algoritmo richiederebbe un tempo eccessivo (nelle tesi è n=3).

Le catene di tag

⇒ la miglior sequenza di tag $\widehat{t}_{1,n}$ che approssimi la frase $\omega_1, \dots \omega_n$ è:

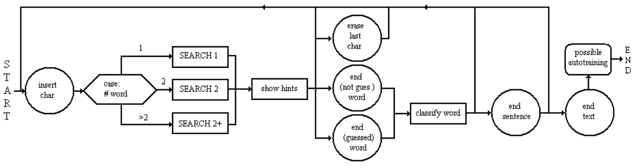
$$\widehat{t}_{1,n} = arg\ max_{\{t_{1,n}\}} \left[\mathbb{P}(t_{1,n}|\omega_{1,n})\right] = arg\ max_{\{t_{1,n}\}} \left[\prod_{i=1}^{n} \mathbb{P}(\omega_i|t_i)\mathbb{P}(t_i|t_{i-1})\right]$$

Dunque nella word prediction

- si possono "`filtrare`" tutte le parole suggerite e mostrare solo le più probabili secondo la grammatica statistica dell'*n*-gram[a]
- il sistema può "`apprendere`" aggiornando le occorrenze usate nel calcolo delle probabilità.
- c'è perdita di informazione "`semantica`", dovuta al fatto che si considerano le POS anziché le parole distinte.

---

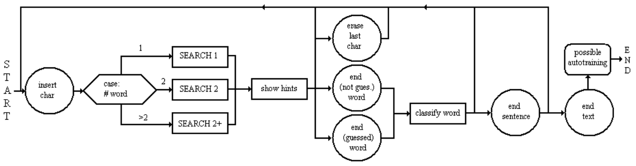
[a] Tipicamente non si supera il valore n=4, altrimenti l'algoritmo richiederebbe un tempo eccessivo (nelle tesi è n=3).



- SEARCH 1: per la prima parola della frase non si possono sfruttare le triple, la predizione consiste nella sola ricerca all'interno del lessico[3];
- SEARCH 2: conoscendo la/e POS della prima parola, la ricerca della seconda si restringe considerando le triple concordanti con le classificazioni della prima parola;
- SEARCH 2+: si sfruttano appieno le triple e si filtrano le parole cercate nel lessico in base alla/e POS più probabili.

---

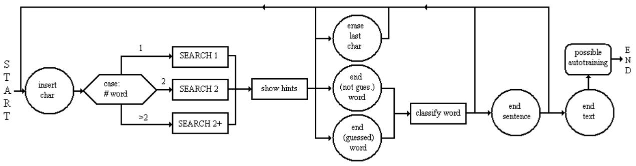
[3] Si può personalizzare l'algoritmo creando un **dizionario privato** come pure delle **triple private**.



- SEARCH 1: per la prima parola della frase non si possono sfruttare le triple, la predizione consiste nella sola ricerca all'interno del lessico[3];
- SEARCH 2: conoscendo la/e POS della prima parola, la ricerca della seconda si restringe considerando le triple concordanti con le classificazioni della prima parola;
- SEARCH 2+: si sfruttano appieno le triple e si filtrano le parole cercate nel lessico in base alla/e POS più probabili.

---

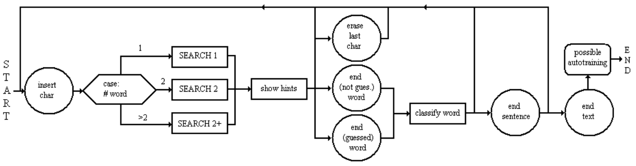
[3] Si può personalizzare l'algoritmo creando un **dizionario privato** come pure delle **triple private**.



- SEARCH 1: per la prima parola della frase non si possono sfruttare le triple, la predizione consiste nella sola ricerca all'interno del lessico[3];
- SEARCH 2: conoscendo la/e POS della prima parola, la ricerca della seconda si restringe considerando le triple concordanti con le classificazioni della prima parola;
- SEARCH 2+: si sfruttano appieno le triple e si filtrano le parole cercate nel lessico in base alla/e POS più probabili.

---

[3] Si può personalizzare l'algoritmo creando un **dizionario privato** come pure delle **triple private**.

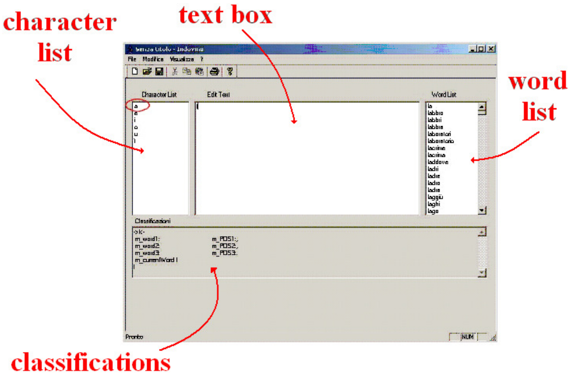


- SEARCH 1: per la prima parola della frase non si possono sfruttare le triple, la predizione consiste nella sola ricerca all'interno del lessico[3];
- SEARCH 2: conoscendo la/e POS della prima parola, la ricerca della seconda si restringe considerando le triple concordanti con le classificazioni della prima parola;
- SEARCH 2+: si sfruttano appienno le triple e si filtrano le parole cercate nel lessico in base alla/e POS più probabili.

---

[3] Si può personalizzare l'algoritmo creando un **dizionario privato** come pure delle **triple private**.

"La ragazza. . . "

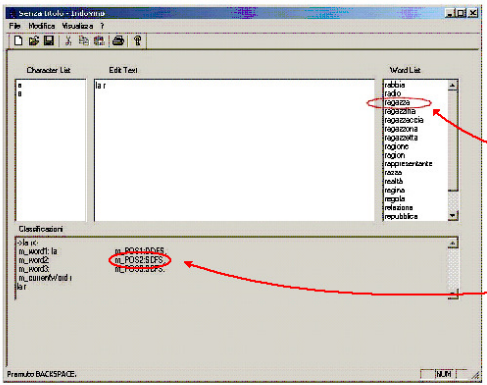


a
e
i
o
u
'

la
labbro
labbri
labbra
laboratori
laboratorio
lacrime
lacrima
laddove
ladri
ladre
ladro
ladra
laggiù
laghi
lago

->k<-
m_word1: m_POS1:,
m_word2: m_POS2:,
m_word3: m_POS3:.
m_currentWord l

"La ragazza. . . "



a
e

la r

rabbia
radio
ragazza
ragazzina
ragazzaccia
ragazzona
ragazzetta
ragione
ragion
rappresentante
razza
realtà
regina
regola
relazione
repubblica

->la r<-
m_word1: la m_POS1:DDFS,
m_word2: m_POS2:SDFS,
m_word3: m_POS3:SDFS.
m_currentWord r
la r

la parola compare tra i suggerimenti

per la word2:
sostantivo S
comune C
femminile F
singolare S

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Le tre caratteristiche

- *copertura del lessico* (quasi) totale[a]
- *concordanze sintattiche* (genere, numero, forme verbali)
- *autoapprendimento*

---

[a] Non totale in quanto ricavato da un corpus, che è evidentementemente limitato.

Preferenze lessicali

- Dizionario generale: *italbase*[4];
- **Dizionario privato**: dizionario di classificazioni dell'utente.

Preferenze sintattiche

- Triple generali: dovute ai corpora per l'addestramento di inizializzazione;
- **Triple private**: individuano lo stile di composizione dell'utente.

---

[4] 43.800 lemmi, 876.000 forme, 1.165.000 classificazioni

Sviluppi

Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\ldots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\ldots,\omega_{n-1}) + \ldots \alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\ldots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Sviluppi

Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\dots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\dots,\omega_{n-1}) + \dots\alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\dots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Sviluppi

Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\dots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\dots,\omega_{n-1}) + \dots\alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\dots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Sviluppi

Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\dots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\dots,\omega_{n-1}) + \dots\alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\dots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Sviluppi

Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\dots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\dots,\omega_{n-1}) + \dots\alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\dots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Sviluppi

Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\dots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\dots,\omega_{n-1}) + \dots\alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\dots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Sviluppi

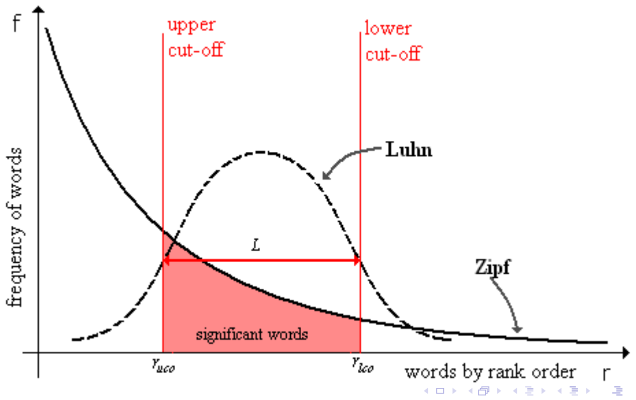
Limitazioni:

- non si tiene conto delle occorrenze delle singole parole;
- si usano solo i 3-gram;
- si perde in semantica.

Possibili soluzioni:

- nell'autoapprendimento (quindi una tantum) si possono calcolare le frequenze (relative) delle varie forme (*1-gram*), offrendo un ulteriore filtro decisionale;
- si possono pesare in "modo opportuno" le probabilità di 3-gram, 2-gram e 1-gram (in generale $\alpha_{n-1}\mathbb{P}(\omega_n|\omega_1,\dots,\omega_{n-1}) + \alpha_{n-2}\mathbb{P}(\omega_n|\omega_2,\dots,\omega_{n-1}) + \dots\alpha_1\mathbb{P}(\omega_n|\omega_{n-2},\dots,\omega_{n-1}) + \alpha_0\mathbb{P}(\omega_n)$)
- utilizzo del *potere discriminante* calcolabile tramite la regola di *Luhn* (per cui è essenziale conoscere le frequenze delle diverse forme) per ricercare solo in una sezione dei corpora.

Proposta semantica



Proposta semantica

- *Legge di Zipf*: mette in relazione la **frequenza** (le occorrenze) delle parole con il loro **rango**[5]
- Si introducono[6] due *valori di soglia* di r: le parole sono considerate troppo rare per $r > r_{lco}$ e troppo comuni per $r < r_{uco}$
- *Regola di Luhn*: il **resolving power** delle parole[7] (curva gaussiana) ha un picco per $r \approx L/2$ e tende a zero vicino ai due valori di soglia

⇒ conoscendo le parole discriminanti dei differenti testi nel *training corpus* e ricercando quelle nel testo in via di composizione, possono essere scelte le sezioni del corpus che meglio rispecchiano l'area semantica trattata dall'utente. Questo implicherebbe, oltre ad una **velocizzazione della ricerca**, la necessità di avere un *training corpus* (all'occorrenza estendibile dall'utente) costituito da testi che coprano le diverse aree della conoscenza.

---

[5] La parola con freq. maggiore ha rango=1, la seconda in ordine di freq. ha rango=2, e così via, finché non si arriva alla parola con la minor freq., che ha rango massimo.

[6] Seguendo i lavori di alcuni studiosi, e in particolare di H.P.Luhn.

[7] La loro capacità di discriminare il contenuto dei documenti: questa resta un'ipotesi di lavoro ben fondata.

Proposta semantica

- *Legge di Zipf*: mette in relazione la **frequenza** (le occorrenze) delle parole con il loro **rango**[5]
- Si introducono[6] due *valori di soglia* di r: le parole sono considerate troppo rare per $r > r_{lco}$ e troppo comuni per $r < r_{uco}$
- *Regola di Luhn*: il **resolving power** delle parole[7] (curva gaussiana) ha un picco per $r \approx L/2$ e tende a zero vicino ai due valori di soglia

⇒ conoscendo le parole discriminanti dei differenti testi nel *training corpus* e ricercando quelle nel testo in via di composizione, possono essere scelte le sezioni del corpus che meglio rispecchiano l'area semantica trattata dall'utente. Questo implicherebbe, oltre ad una **velocizzazione della ricerca**, la necessità di avere un *training corpus* (all'occorrenza estendibile dall'utente) costituito da testi che coprano le diverse aree della conoscenza.

---

[5] La parola con freq. maggiore ha rango=1, la seconda in ordine di freq. ha rango=2, e così via, finché non si arriva alla parola con la minor freq., che ha rango massimo.

[6] Seguendo i lavori di alcuni studiosi, e in particolare di H.P.Luhn.

[7] La loro capacità di discriminare il contenuto dei documenti: questa resta un ipotesi di lavoro ben fondata.

Proposta semantica

- *Legge di Zipf*: mette in relazione la **frequenza** (le occorrenze) delle parole con il loro **rango**[5]
- Si introducono[6] due *valori di soglia* di r: le parole sono considerate troppo rare per $r > r_{lco}$ e troppo comuni per $r < r_{uco}$
- *Regola di Luhn*: il **resolving power** delle parole[7] (curva gaussiana) ha un picco per $r \approx L/2$ e tende a zero vicino ai due valori di soglia

⇒ conoscendo le parole discriminanti dei differenti testi nel *training corpus* e ricercando quelle nel testo in via di composizione, possono essere scelte le sezioni del corpus che meglio rispecchiano l'area semantica trattata dall'utente. Questo implicherebbe, oltre ad una **velocizzazione della ricerca**, la necessità di avere un *training corpus* (all'occorrenza estendibile dall'utente) costituito da testi che coprano le diverse aree della conoscenza.

[5] La parola con freq. maggiore ha rango=1, la seconda in ordine di freq. ha rango=2, e così via, finché non si arriva alla parola con la minor freq., che ha rango massimo.

[6] Seguendo i lavori di alcuni studiosi, e in particolare di H.P.Luhn.

[7] La loro capacità di discriminare il contenuto dei documenti: questa resta un ipotesi di lavoro ben fondata.

Proposta semantica

- *Legge di Zipf*: mette in relazione la **frequenza** (le occorrenze) delle parole con il loro **rango**[5]
- Si introducono[6] due *valori di soglia* di r: le parole sono considerate troppo rare per $r > r_{lco}$ e troppo comuni per $r < r_{uco}$
- *Regola di Luhn*: il **resolving power** delle parole[7] (curva gaussiana) ha un picco per $r \approx L/2$ e tende a zero vicino ai due valori di soglia

⇒ conoscendo le parole discriminanti dei differenti testi nel *training corpus* e ricercando quelle nel testo in via di composizione, possono essere scelte le sezioni del corpus che meglio rispecchiano l'area semantica trattata dall'utente. Questo implicherebbe, oltre ad una **velocizzazione della ricerca**, la necessità di avere un *training corpus* (all'occorrenza estendibile dall'utente) costituito da testi che coprano le diverse aree della conoscenza.

[5] La parola con freq. maggiore ha rango=1, la seconda in ordine di freq. ha rango=2, e così via, finché non si arriva alla parola con la minor freq., che ha rango massimo.

[6] Seguendo i lavori di alcuni studiosi, e in particolare di H.P.Luhn.

[7] La loro capacità di discriminare il contenuto dei documenti: questa resta un ipotesi di lavoro ben fondata.

Conclusioni

> L'Occidente si rifà in genere al Vangelo secondo Giovanni, pur spezzando in traduzione la circolarità dell'originale, che recita: *"In principio era la Parola, e la Parola era presso Dio, e Dio era la Parola"*. La distinzione fra linguistica e aritmetica era però labile, per i greci. Da un lato, il logos stava a significare anche il rapporto numerico, oltre alla parola e alla ragione. Dall'altro lato, i greci non avevano simboli specifici per le cifre, e usavano allo scopo le lettere dell'alfabeto. Fondare il mondo sulle parole o sui numeri non doveva dunque apparire loro incompatibile.
>
> (*La tela di Pitagora*, di *Piergiorgio Odifreddi*)

BIBLIOGRAFIA

- `Barsocchi Daniele`, *Disabilità, Informatica, Linguistica: un'istanza del trinomio* (Tesi di laurea 2001/2001)
- `Carmignani Nicola`, *Progetto di un sistema di Word Prediction per Persoe Disabili basato su Part-Of-Speech Tagging* (Tesi di laurea 2003/2004)

1. Utilità della WP
   - Perché farlo
   - Alcuni prodotti
   - Lo scopo
2. Linguistica computazionale e statistica
   - Il dizionario
   - Il parser
   - Le sestuple
   - Grammatica statistica
3. Funzionamento della WP
   - Lo schema
   - Un esempio
   - Punti chiave
4. Possibili sviluppi. . . e conclusioni