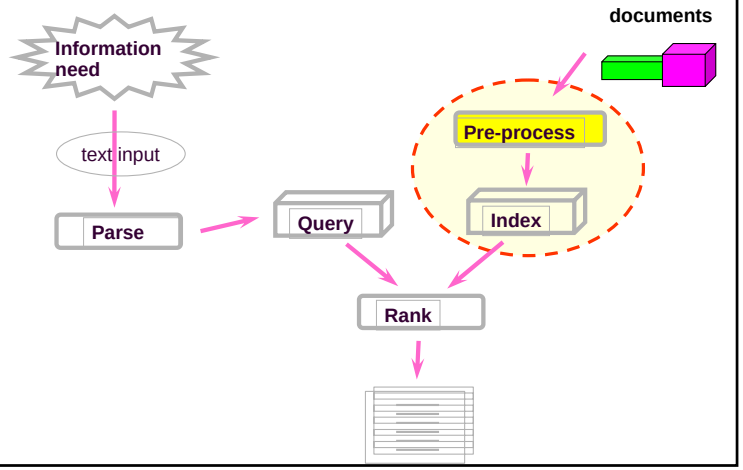
Indicizzazione

Fasi del processo di IR



Indicizzazione: due aspetti

- Costruzione delle viste logiche dei documenti:
  - Per ogni documento si identificano i termini che li caratterizzano
  - Ad ogni coppia termine-documento si associa un peso, che riflette l'importanza del termine rispetto al documento
- Costruzione dell'indice
  - Le viste dei documenti sono organizzate in modo da facilitarne il recupero a fronte di una query
  - Struttura dati che conserva le associazioni fra termini e documenti
    - Dato un termine *t*, permette di recuperare in modo efficiente tutti i documenti per i quali *t* è rilevante

Indicizzazione:

Terminologia

- Corpus:
  - insieme di documenti
- Termine indice
  - parola (o radice di parole) presente in qualche documento del corpus
- Vocabolario:
  - lista di *termini indice* estratti dal corpus
- Indice:
  - struttura dati che memorizza l'associazione fra termini o argomenti e documenti

Esempio

- Corpus
  - Documento A: “The quick brown fox”
  - Documento B: “crazy like a fox”
- Vocabolario:
  - t_1= **brown**
  - t_2= **crazy**
  - t_3= **fox**
  - t_4= **quick**
- Indice:
  - t_1 (brown): **A**
  - t_2 (crazy): **B**
  - t_3 (fox): **A, B**
  - t_4 (quick): **A**

Linguaggio di indicizzazione

- Come scegliere i termini del linguaggio di indicizzazione:
  - Linguaggio controllato: limitato ad un vocabolario predefinito
  - Linguaggio libero: termini estratti liberamente dal testo del documento e non definiti a-priori
    - **full-text**: tutte le parole contenute nel documento sono utilizzate per la sua vista logica
    - selezione di un sottoinsieme di termini rilevanti, mediante operazioni di elaborazione del testo (normalizzazione lessicale, stemming, …)
- Forma dei termini del linguaggio
  - Termini singoli (es. “recupero”, “informazione”, “sistema”…)
  - Termini in contesto: composti da diverse parole (es. “sistemi di recupero dell’informazione”)

Processo di indicizzazione

- Manuale:
  - Un esperto sceglie quali termini meglio caratterizzano il contenuto di un documento
  - Più “semantico” e quindi migliore
- Automatico:
  - fatto da un’applicazione software
  - Più sintattico, su base statistica e quindi “peggiore”
- Indicizzazione automatica o manuale?
  - In media c’è un accordo del 60% con le due tecniche, paragonabile all’accordo che esiste tra due indicizzatori “umani”
  - L’approccio manuale, anche se qualitativamente superiore, non è scalabile (richiede tempo e competenze)
  - In certi domini (es. Web) l’approccio automatico è l’unica soluzione

Indicizzazione: configurazioni tipiche

- Indicizzazione manuale
- Linguaggio controllato
- Linguaggio a termini in contesto

- Indicizzazione automatica
- Linguaggio libero
- Linguaggio a termini singoli

Qualità dell'indicizzazione

- Esaustività:
  - il grado in cui i termini indice rappresentano i vari concetti trattati
- Specificità:
  - il grado di specificità della rappresentazione dei concetti
- Indicizzazione profonda:
  - alto grado di esaustività e specificità
  - più costosa
  - precisione e richiamo più alti
- Indicizzazione superficiale:
  - uso di alcuni termini generici
  - meno costosa
  - prestazioni inferiori

Struttura del linguaggio di indicizzazione

- Dizionario:
  - termini ordinati alfabeticamente
- Schema di classificazione:
  - codici che organizzano i termini gerarchicamente
- Thesaurus:
  - termini organizzati in una "rete semantica"

Schema di classificazione

- Esempio (schema di classificazione Dewey)

15 psicologia
  152 psicofisiologia
    1521 percezioni sensoriali
  153 processi mentali
  154 subconscio

Thesaurus

- Un thesaurus consiste di:
  - un insieme di vocaboli ed espressioni-chiave rilevanti per un particolare dominio
  - un insieme di **relazioni semantiche** fra vocaboli
- I thesauri sono di solito definiti da esperti del settore
- L'utilizzo di thesauri è vantaggioso per domini in cui è possibile la standardizzazione dei termini di ricerca
  - esempio: ambito medico, legale,ecc.

Thesaurus: tipi di relazioni semantiche

- Sinonimie:
  - Es. calcolatore "è sinonimo di" elaboratore, calcolatrice
- Ipernimie / Iperonimie:
  - Es. "felini" è un termine più generale di "gatti", "leoni"
- Associazioni: vari tipi, dipendenti dal contesto
  - antinomia: vittoria--sconfitta
  - concomitanza: sintomo--malattia
  - proprietà: trampolino--altezza
  - inclusione: contenuto--contenente
  - localizzazione: partita--stadio

Uso di un thesaurus: esempi

- In fase di indicizzazione:
  - Rimando a "forme ufficiali" per aumentare l'oggettività dell'indicizzazione
  - Espansione dell'indice con sinonimi (costoso), es:

Indice iniziale		*Indice "espanso"*
quick: **A**	→	*fast:* ***A***
		rapid: ***A***
		...

- … oppure in fase di recupero:
  - Uso dei sinonimi per aumentare il richiamo e rendere le ricerche meno "sintattiche"
  - Recupero per tematismi:
    - Posate per forchette, coltelli, cucchiai
    - Europa per Italia, Francia, Spagna …

Indicizzazione automatica

- Lo scopo del processo di indicizzazione è duplice:
  - Estrarre da un documento un insieme di termini, che ne caratterizzano il contenuto
  - Assegnare a ciascuno di questi termini un peso, che ne riflette l'importanza per il documento
- Assunzione di base:
  - la frequenza di un termine in un documento è indicativa della sua importanza nel caratterizzarne il contenuto

> *"… Uno scrittore normalmente ripete certe parole nell'elaborare aspetti di un certo argomento. L'enfasi è considerata un indicatore di significatività …".* [Luhn]

Studi statistici

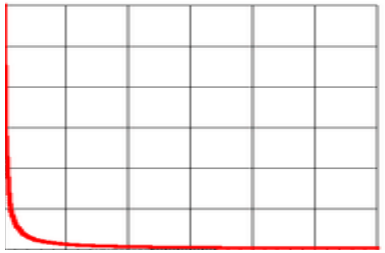
- Le occorrenze delle parole nei testi non sono distribuite in modo uniforme (o gaussiano), ma seguono una distribuzione Zipf
  - Poche parole con alta frequenza, molte parole con bassa frequenza
  - Per la lingua inglese il 20% delle parole usate in un testo copre il 70% del testo stesso
  - Le 250 parole più comuni coprono in media il 40-50% di un testo
- Spiegazione:
  - Principio dello "sforzo minimo": è più facile ripetere parole di uso comune che usarne di nuove
  - Le parole più usate sono quelle di lunghezza breve e di uso comune

Studi statistici: la legge di Zipf

- Il prodotto fra la frequenza (f) di una parola ed il suo rango (r) è quasi costante: $f = C / r$
  - Il termine di rango 1 occorre C volte
  - Il secondo termine più frequente occorre C/2 volte
  - Il terzo termine occorre C/3 volte

Rango	Termine	Frequenza	(Rango × Frequenza) /1.000.000
1	the	69.971	0,070
2	of	36.411	0,073
3	and	28.852	0,086
4	to	26.149	0,104
5	a	23.237	0,116
6	in	21.341	0,128
7	that	10.595	0,074
8	is	10.099	0,081
9	was	9.816	0,088
10	he	9.543	0.095

La legge di Zipf

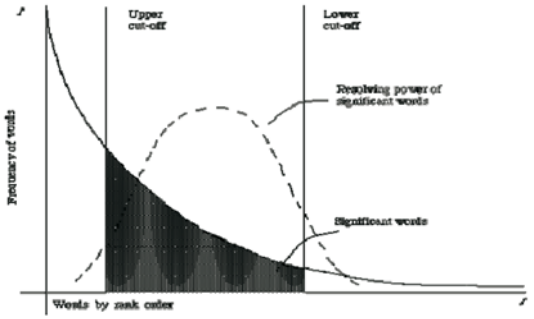


Distribuzione delle parole nei testi, in ordine di frequenza decrescente (rango)

Frequenza vs. Potere discriminante

- Le parole in assoluto più frequenti sono anche poco significative, in quanto non sono buoni discriminatori
  - "stopwords" (avverbi, articoli, preposizioni ecc.)
  - tuttavia non basta solo la frequenza assoluta: bisogna considerare la frequenza rispetto all'intera collezione
    - Es: 'computer' non è in grado di caratterizzare (discriminare) un documento all'interno di biblioteca di informatica
- Esiste un grande numero di parole che appaiono pochissime volte e possono essere trascurate
  - es: *Precipitevolissimevolmente*
- Assunzione tipica
  - Parole mediamente frequenti = Parole discriminanti

Frequenza vs. Potere discriminante



Come calcolare i pesi dei termini: schema di pesatura TF-IDF

- Peso w_{ij} = tf(i,j) * idf(i)
- TF (Term frequency)
  - Più un termine è frequente nel documento più è importante per il documento, in quanto è maggiormente indicativo dell'argomento trattato
- IDF (Inverse Document Frequency)
  - Termini che appaiono in molti documenti diversi sono meno indicativi dell'argomento trattato in un documento
  - Indicatore del potere di discriminazione di un termine

Schema di pesatura TF-IDF

- Siano:
  - N il numero totale di documenti
  - n_i il numero di documenti che contengono k_i
  - $\mathbf{freq}_{i,j}$ la frequenza di k_i in d_j
- Fattore TF (term frequency):

$$tf(i,j) = \frac{freq_{i,j}}{\max_k (freq_{k,j})}$$

- Fattore IDF (inverse document frequency):

$$idf_i = \log \frac{N}{n_i}$$

Schema di pesatura TF-IDF: esempio

- Collezione di documenti e termini contenuti:
  - La collezione consiste di 10.000 documenti: $N = 10.000$
  - I documenti contengono solo i seguenti 3 termini:

    A(50), B(1300), C(250)
  - per ogni termine è riportato in parentesi il numero di documenti che lo contengono: $n_A = 50$, $n_B = 1300$, $n_C = 250$
- Documento *d:*
  - Frequenze dei termini in *d:* A(3), B(2), C(1)
    - $freq_{A,d} = 3$, $freq_{B,d} = 2$, $freq_{C,d} = 1$
  - I pesi dei termini secondo lo schema tf-idf:

    A: tf = 3/3; idf = log(10000/50) = 5.3; → tf-idf= 5.3

    B: tf = 2/3; idf = log(10000/1300) = 2.0; → tf-idf= 1.3

    C: tf = 1/3; idf = log(10000/250) = 3.7; → tf-idf= 1.2
  - Rappresentazione vettoriale: (5.3,1.3,1.2))

Strutture dati per l' indicizzazione (indici)

Indici

- Problema
  - Come organizzare l'informazione disponibile nell'archivio per rispondere velocemente ed efficacemente alle interrogazioni poste dall'utente ?
- Indice
  - struttura dati utilizzata per recuperare i documenti sulla base di termini indice
  - Associa ad ogni termine indice t l'insieme dei documenti correlati a t
- Vari tipi di indici:
  - Liste invertite o File Invertiti (Inverted files)
  - Array di suffissi
  - Signature files

Calcolo di query nel modello booleano

Indice naive = matrice (binaria) documenti-termini

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	1	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

1 se l'opera contiene il termine, 0 altrimenti

Per rispondere a Q=[***Brutus*** *AND* ***Caesar*** AND *NOT* ***Calpurnia***]
- Prendiamo i vettori di ***Brutus, Caesar, Calpurnia***
- Complementiamo il vettore di ***Calpurnia***
- Eseguiamo l'***AND*** logico bit-a-bit

Efficienza ?

- Problema: occupazione di memoria eccessiva
  - Tutte le celle della matrice devono essere memorizzate, anche se il loro valore è 0
  - Esempio: se ci sono 1000 termini distinti e 1 milione di documenti, la matrice occupa $10^3 \times 10^6 \times$ 1byte = 1Gb
- Obiettivo: ottenere una struttura dati più efficiente
  - Minore quantità di memoria occupata
  - Accesso veloce alle occorrenze dei termini nei documenti

Liste invertite

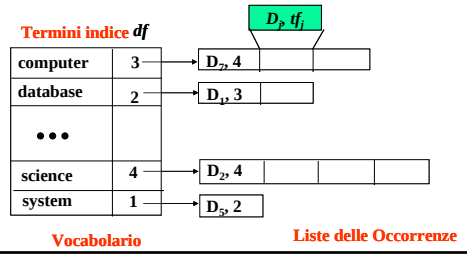
Il nome indica l'inversione della matrice documenti-termini:
- termini sulle righe, documenti sulle colonne
- si accede ai documenti a partire dai termini

docs	*t1*	*t2*	*t3*
D1	1	0	1
D2	1	0	0
D3	0	1	1
D4	1	0	0
D5	1	1	1
D6	1	1	0
D7	0	1	0
D8	0	1	0
D9	0	0	1
D10	0	1	1

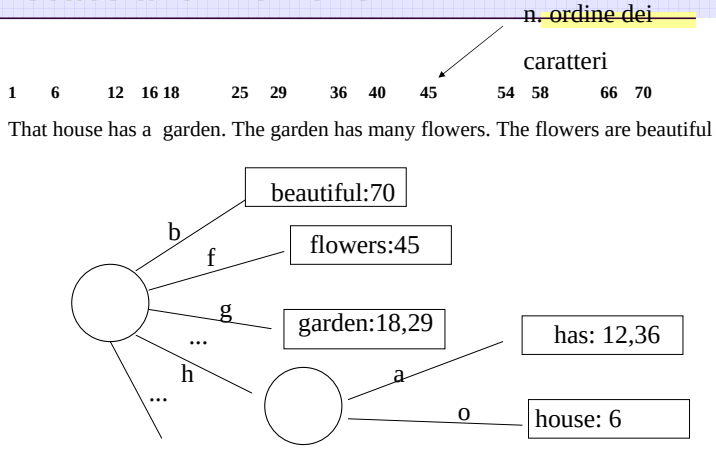
Terms	D1	D2	D3	D4	D5	D6	D7	…
t1	1	1	0	1	1	1	0	
t2	0	0	1	0	1	1	1	
t3	1	0	1	0	1	0	0	

Liste invertite

- Vocabolario *V*:
  - l'insieme di termini indice
  - la taglia di *V* dipende dalle operazioni effettuate sui testi
- Liste delle occorrenze (o *posting lists*):
  - Ad ogni termine del vocabolario è associata la lista dei documenti in cui occorre: **<documento + tf>**



Memorizzazione del vocabolario in un **trie**



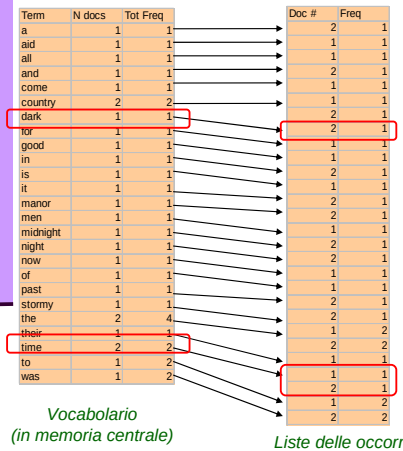
Ricerca con liste invertite

1. Ricerca nel vocabolario: i termini presenti nella query sono ricercati nel vocabolario
   - uso di strutture dati per ricerche efficienti (es., *tabelle hash, B-trees*)
   - alternativamente, i termini sono tenuti in un vettore ordinato (per ordine alfabetico), su cui si fanno ricerche binarie
2. Recupero delle occorrenze: si estraggono le liste delle occorrenze di tutti i termini della query
3. Manipolazione delle occorrenze: si scandiscono le liste delle occorrenze per generare la risposta alla query

Esecuzione di query booleane su un indice a liste invertite

- Caso base: query congiuntive (es. *madding* AND *ignoble*)
  1. per ogni termine della query, si recupera dall'indice la lista dei documenti che lo contengono
  2. si calcola l'intersezione delle liste
- Caso generale: query in forma normale congiuntiva, ad esempio: (*madding* OR *crowd*) AND (*ignoble* OR *strife*)
  1. si recuperano le lista dei documenti per ognuno dei termini
  2. si fondono le liste dei termini in OR (componente disgiuntiva)
  3. si calcola l'intersezione delle liste ottenute al passo 2
- Ottimizzazioni possibili:
  - Le liste sono tenute ordinate
  - Altrimenti, si calcola l'intersezione a partire dalle liste più corte
    - La lunghezza di una componente disgiuntiva si stima sommando le lunghezze delle liste dei singoli termini che la costituisce

Esecuzione di query booleane: Esempio



Term	N docs	Tot Freq
a	1	1
aid	1	1
all	1	1
and	1	1
come	1	1
country	2	2
dark	1	1
for	1	1
good	1	1
in	1	1
is	1	1
it	1	1
manor	1	1
men	1	1
midnight	1	1
night	1	1
now	1	1
of	1	1
past	1	1
stormy	1	1
the	2	4
their	1	1
time	2	2
to	1	2
was	1	2

Doc #	Freq
2	1
1	1
1	1
2	1
1	1
1	1
2	1
2	1
1	1
1	1
2	1
1	1
2	1
2	1
1	1
2	1
2	1
1	1
2	1
1	2
2	2
1	1
2	1
1	2
2	2

Vocabolario (in memoria centrale)

Liste delle occorrenze (su disco)

- Query: “dark” AND “time”
- Calcolo della query
  1. Cerco “dark” nel vocabolario
  2. Recupero la lista di “dark”:
     [2]
  3. Cerco “time” nel vocabolario
  4. Recupero la lista di “time”
     [1, 2]
  5. Calcolo l’intersezione delle liste:
     [2]
- Solo il doc. 2 soddisfa la query

Liste invertite per query posizionali

- La coppia <docID, tf> non basta per query che riguardano le posizioni di occorrenza dei termini
  - Phrase queries
    - *“Paolino Paperino”*
  - Proximity queries
    - *apache* NEAR *lucene* (nella sintassi di Lucene: *“apache lucene”* ~2)
- Bisogna aggiungere nell’indice informazioni sulle posizioni che i termini occupano nei documenti

< *doc1*: pos1, pos2, pos3, … ;
doc2: pos1, pos2, pos3, … ;
…
>

Liste invertite posizionali: Esempio

- Un corpus di 2 documenti:
  - documento *D1:*

1	2	3	4	5	6	7	8	9	10	11	12	13	14
That	house	has	a	garden.	The	garden	has	many	flowers.	The	flowers	are	beautiful

  - documento *D2:*

1	2	3	4	5	6	7
A	beautiful	house	must	have	a	garden

- Indice a liste invertite:

beautiful	→	(*D1*: 14; *D2*: 2)
flowers	→	(*D1*: 10, 12)
garden	→	(*D1*: 5, 7; *D2*: 7)
house	→	(*D1*: 2; *D2*: 3)

Vocabolario **Liste delle occorrenze (posting lists)**

Esecuzione di “phrase query” (2)

- Phrase query: *“Paolino Paperino”*
- Calcolo della query:
  - Si estraggono le posting list dei termini: ***Paolino, Paperino***
  - Troviamo i documenti che contengono tutte le parole presenti nell’interrogazione
  - Esaminiamo le posizioni delle occorrenze e garantiamo che siano consecutive nel documento
    - ***Paolino***:
      - 2:1,17,74,222,551; 4:8,27,101,429,433; 7:13,23; ...
    - ***Paperino***:
      - 1:17,19; 4:17,191,291,430,436; 5:14,19,101; ...

Costruzione di un indice a liste invertite (1)

Doc 2

So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious

Doc 1

I did enact Julius Caesar I was killed i' the Capitol; Brutus killed me.

I documenti sono analizzati per estrarre i termini e questi sono memorizzati insieme al corrispondente DocID

Term	DocID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Costruzione di un indice a liste invertite (2)

Term	DocID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Ordiniamo tutti i termini in modo lessicografico per formare il Dizionario

Term	DocID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Costruzione di un indice a liste invertite (3)

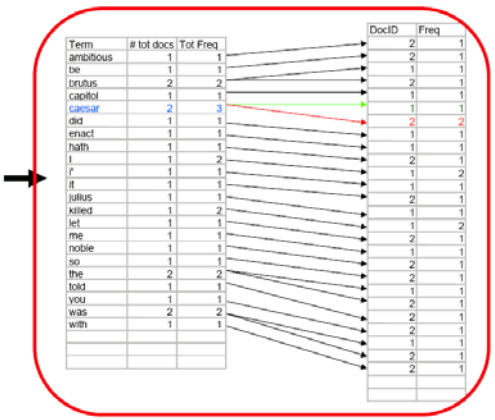
Term	DocID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Le occorrenze dello stesso termine nello stesso documento sono "fuse insieme" incrementando opportunamente il valore della frequenza del termine

Term	DocID	Freq
ambitious	2	1
be	2	1
brutus	1	1
brutus	2	1
capitol	1	1
caesar	1	1
caesar	2	2
did	1	1
enact	1	1
hath	2	1
I	1	2
i'	1	1
it	2	1
julius	1	1
killed	1	2
let	2	1
me	1	1
noble	2	1
so	2	1
the	1	1
the	2	1
told	2	1
you	2	1
was	1	1
was	2	1
with	2	1

Costruzione di un indice a liste invertite (4)

Term	DocID	Freq
ambitious	2	1
be	2	1
brutus	1	1
brutus	2	1
capitol	1	1
caesar	1	1
caesar	2	2
did	1	1
enact	1	1
hath	2	1
I	1	2
i'	1	1
it	2	1
julius	1	1
killed	1	2
let	2	1
me	1	1
noble	2	1
so	2	1
the	1	1
the	2	1
told	2	1
you	2	1
was	1	1
was	2	1
with	2	1



Term	# tot docs	Tot Freq
ambitious	1	1
be	1	1
brutus	2	2
capitol	1	1
caesar	2	3
did	1	1
enact	1	1
hath	1	1
I	1	2
i'	1	1
it	1	1
julius	1	1
killed	1	2
let	1	1
me	1	1
noble	1	1
so	1	1
the	2	2
told	1	1
you	1	1
was	2	2
with	1	1

DocID	Freq
2	1
2	1
1	1
2	1
1	1
1	1
2	2
1	1
1	1
1	1
2	1
1	2
1	1
2	1
1	1
1	2
2	1
1	1
2	1
2	1
1	1
2	1
2	1
1	1
2	1
2	1