Capitole Speciale de Informatică

Curs 4: Calculul scorurilor în un sistem complet de extragere a informaţiilor

18 octombrie 2018

Reamintim că în cursul precedent am prezentat modelul de spaţiu vectorial în care fiecare document d din o colecţie D de N documente care are vocabularul de termeni $V = \{t_1, \ldots, t_M\}$, este reprezentat ca un vector

$$\vec{V}(d) = (\mathrm{wf}_{t_1,d}, \ldots, \mathrm{wf}_{t_M,d})$$

unde $\mathrm{wf}_{t_i,d}$ este greutatea termenului t în documentul d. De obicei, $\mathrm{wf}_{t,d}$ se defineşte în unul din următoarele 2 feluri:

1. $\mathrm{wf}_{t,d} := \mathrm{tf}_{t,d}$, adică numărul de apariţii ale termenului t în documentul d, sau
2. $\mathrm{wf}_{t,d} := \text{tf-idf}_{t,d} = \mathrm{idf}_t \cdot \mathrm{tf}_{t,d}$ unde $\mathrm{idf}_t = \log \frac{N}{\mathrm{df}_t}$

Şi cererile se reprezintă ca vectori: o cerere $q = t_{i_1}\ t_{i_2}\ \ldots\ t_{i_n}$ este descrisă de vectorul

$$\vec{V}(q) = (\mathrm{w}_{t_1,q}, \ldots, \mathrm{w}_{t_M,q})$$

unde $w_{t_j,q} = 1$ dacă $j \in \{i_1, \ldots, i_n\}$ şi $w_{t_j,q} = 0$ în caz contrar.

Valorile normalizate ale acestor vectori sunt vectorii de lungime 1 definiţi astfel:

$$\vec{v}(d) := \frac{\vec{V}(d)}{|\vec{V}(d)|} \qquad \text{şi} \qquad \vec{v}(q) := \frac{\vec{V}(q)}{|\vec{V}(q)|} = \frac{|\vec{V}(q)|}{\sqrt{n}}$$

Scorul (sau gradul) de similaritate cosinusoidală dintre d şi q este cosinus unghiului dintre $\vec{v}(q)$ şi $\vec{v}(d)$, adică produsul scalar $\vec{v}(q) \cdot \vec{v}(d)$.

Sistemele complete de extragere a informaţiilor facilitează determinarea rapidă a primelor K documente care se potrivesc cel mai bine cu o cerere de căutare q, adică acele documente d pentru care scorul $\vec{v}(q) \cdot \vec{v}(d)$ este cât mai mare. Pentru performanţă ridicată (adică timp de răspuns scurt, fără a compromite semnificativ calitatea răspunsului), se elimină nişte calcule inutile şi se folosesc metode euristice care nu garantează calculul exact al celor mai bune K potriviri, dar dau K răspunsuri apropiate de cele mai bune K.

În acest curs sunt prezentate (1) tehnici şi metode euristice de calcul al documentelor cu cele mai bune scoruri de potrivire cu o cerere dată, (2) o arhitectură pentru un motor de căutare web, bazat pe generalizări ale indecşilor şi metodelor de calcul al scorului de potrivire, şi (3) extensii ale modului de interogare pentru modelul de reprezentare a documentelor în spaţiu vectorial.

Optimizare: evitarea normalizării vectorului $\vec{V}(q)$

Primele K documente care se potrivesc cel mai bine cu o cerere q sunt documentele $d_1, \ldots, d_K \in D$ astfel încât

- $\vec{v}(q) \cdot \vec{v}(d_i) \geq \vec{v}(q) \cdot \vec{v}(d_{i+1})$ pentru $1 \leq i < K$, şi
- $\vec{v}(q) \cdot \vec{v}(d_K) \geq \vec{v}(q) \cdot \vec{v}(d)$ pentru toţi $d \in D \setminus \{d_1, \ldots, d_K\}$.

Se observă că

$$\begin{aligned} \vec{v}(q) \cdot \vec{v}(d) \geq \vec{v}(q) \cdot \vec{v}(d') &\Leftrightarrow \frac{\vec{V}(q)}{|\vec{V}(q)|} \cdot \vec{v}(d) \geq \frac{\vec{V}(q)}{|\vec{V}(q)|} \cdot \vec{v}(d') \\ &\Leftrightarrow \vec{V}(q) \cdot \vec{v}(d) \geq \vec{V}(q) \cdot \vec{v}(d') \end{aligned}$$

Deci, primele K documente care se potrivesc cel mai bine cu o cerere q sunt documentele $d_1, \ldots, d_K \in D$ pentru care $\vec{V}(q) \cdot \vec{v}(d)$ iau valorile cele mai mari. Deoarece componentele nenule ale lui $\vec{V}(q)$ sunt 1, rezultă că algoritmul de calcul al celor mai bune potriviri din cursul 3 poate fi optimizat astfel:

```
ScorCosinusoidalRapid(q)
 1 float scoruri[N] = [0]
 2 for fiecare d din colecţia D do
 3     iniţializează lungime[d]
 4 for fiecare termen t din cererea q do
 5     obţine lista de postări a lui t
 6     for fiecare pereche (d, tf_{t,d}) din lista de postări a lui t do
 7         scoruri[d] += wf_{t,d}
 8 for fiecare document d ∈ D do
 9     scoruri[d] := scoruri[d]/lungime[d]
10 return primele K componente din scoruri[ ]
```

Figure 1: Un algoritm de calcul mai rapid al scorurilor în modelul vectorial.

Extragerea primelor K componente din $scoruri[\,]$ se poate face în 2 paşi: (1) se sortează documentele din D în ordine descrescătoare a scorurilor (în timp $O(N \log N)$) şi se extrag primele K elemente din lista sortată de documente (în timp constant, fiindcă K este o valoare constantă). O alternativă mai eficientă este să se plaseze documentele d cu $scoruri[d] > 0$ în o structură *heap*. Dacă în colecţie sunt J documente d cu $scoruri[d] > 0$, construcţia structurii heap se face folosind $2 \cdot J$ paşi de comparare, iar apoi extragerea primelor K elementele cu cele mai mari scoruri se poate face folosind $\log J$ paşi de comparare.

1 Calculul inexact al primelor K potriviri

Idee: există tehnici cu timp de calcul mult mai scurt care, pentru o cerere q, extrag K documente din o colecţie, pentru care probabilitatea este mare să fie printre primele K documente cu scor de potrivire maxim.

D.p.d.v. al utilizatorului, aceste tehnici de potrivire inexactă sunt adesea un lucru bun:

- timpul de răspuns este mult mai mic,
- măsura de similaritate cosinusoidală este doar o aproximare a gradului de relevanţă perceput de utilizator.

Vom prezenta metode euristice care au probabilitate mare să returneze K documente cu scoruri cosinusoidale apropiate de cele ale primelor K potriviri cosinusoidale. Aceste metode operează în 2 paşi:

1. Se determină o mulţime de candidaţi A care are mai mult de K elemente, adică $K < |A| \ll N$. Nu este necesar ca A să conţină primele K potriviri pentru cererea q, dar sunt şanse mari ca A să conţină multe documente cu scoruri apropiate de primele K scoruri de potrivire.
2. Se returnează K documente din A care au cele mai mari scoruri de potrivire cu q.

a) Eliminarea indecşilor

Aceast a metodă euristică se aplică pentru cereri $q = t_1 \ \ldots \ t_n$ cu cel puţin 2 termeni de interogare (adică $n > 1$).

- Idee de bază: se consideră doar documente $d \in D$ care conţin cel puţin un termen din q.
- Alte euristici posibile:
  - ▶ Se consideră doar documente $d \in D$ care conţin termeni din t cu idf_t mai mare decât o valoare prestabilită. În general, termenii t cu idf_t mic au liste de postări lungi $\Rightarrow$ mulţimea de documente pentru care se calculează scoruri devine mult mai mică.
  - ▶ Se consideră doar documente $d \in D$ cre conţin mulţi termeni din q.

b) Liste de campioni

Listele de campioni se calculează pentru termeni diin dicţionar şi depind de alegerea unui număr întreg pozitiv $r > 0$: lista de campioni a lui $t \in V$ constă din r documente d care au cele mai mari valori pentru $\text{wf}_{t,d}$.

Idee: construcţia lui A bazată pe liste de campioni se face astfel: A este reuniunea listelor de campioni pentru termenii din cererea q. Calculul scorurilor de potrivire se face doar pentru documente din A.

Valoarea lui r se fixează înainte de construcţia indecşilor, şi poate varia în funcţie de termen: de pildă, r poate fi mai mare pentru termeni care apar mai rar.

c) Scoruri şi ordini statice de calitate

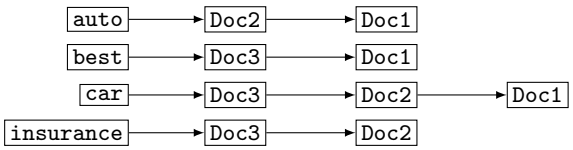
Numeroase motoare de extragere a informaţiilor calculează pentru fiecare document d o măsură $g(d)$ a calităţii documentului, care nu depinde de vreo cerere. Astfel de m]u asuri se numesc scoruri statice de calitate. De obicei, $0 \leq g(d) \leq 1$. Valoarea lui $g(d)$ poate fi determinată în mai multe feluri, ce de exemplu din numărul de comentarii favorabile primite de la vizitatori pe web.

Scorul net de potrivire a unui document d cu o cerere q este o combinaţie între $g(d)$ şi $\text{wf}_{t,d}$. De exemplu, poate fi

$$\text{scor-net}(q,d) = g(d) + \frac{\vec{V}(q) \cdot \vec{V}(d)}{|\vec{V}(q)| \cdot |\vec{V}(d)|}.$$

Pentru procesarea eficientă a listelor de postări, de exemplu pentru intersecţia a două liste de postări, este necesar ca postările tuturor termenilor să fie ordonate la fel, de exemplu:

- în ordine crescătoare a identificatorilor de document, sau
- în ordine descrescătoare a valorilor statice $g(d)$. Exemplul de mai jos este pentru o colecţie de trei documente Doc1, Doc2 şi Doc3 cu scorurile statice de calitate $g(1) = 0.25$, $g(2) = 0.5$, $g(3) = 1$:



Valorile statice $\{g(d) \mid d \in D\}$ ne permit să definim liste globale de campioni pentru o valoare fixată parametrului r: la fiecare termen t asociem ca postări postări r documente care au cele mai mari valori pentru $g(d) + \text{tf-idf}_{t,d}$. Listele de postări ale tuturor termenilor trebuie ordonate după o ordine comună; de exemplu în ordine crescătoare a identificatorului de document, sau în ordine descrescătoare a scorului static de calitate $g(d)$.

d) Ordonare bazată pe impact

Algoritmii de calcul al rangului bazaţi pe gradarea după termeni nu necesită ca postările termenilor să fie ordonate după o ordine comună.

Idee de bază: ordonarea bazată pe impact ordinează documentele din lista de postări a unui termen t în ordine descrescătoare a valorilor lui $\text{tf}_{t,d}$. Prin urmare ordonarea documentelor poate varia de la o listă de postări a unui termen la lista de postări a altui termen.

Acest mod de ordonare a documentelor permite reducerea semnificativă a mulţimii de documente pentru care calculăm scoruri de potrivire:

1. Putem fixa criterii de oprire a traversării unei liste de postări pentru un termen t, de exemplu (1) după ce s-a traversat un număr de r postări; sau (2) după ce $\text{tf}_{t,d}$ a devenit mai mică decât o anumită valoare.

2. Termenii care apar în cererea q se traversează în ordine descrescătoare a idf_t. Această euristică se bazează pe observaţia că termenii t din q care contribuie cel mai mult la scorurile de potrivire a documentelor sunt cei pentru care idf_t are valoare mare.

   $\Rightarrow$ putem fixa criterii de încetare a acumulării de valori pentru scorurile documentelor, pentru termeni din q care contribuie cu valori mici.

e) Reducere bazată pe grupuri

În reducerea bazată pe grupuri (engl. *cluster pruning*) se efectuează un pas de preprocesare care produce grupuri de documente, iar atunci când se primeşte o cerere de informare, se iau în considerare doar documentele dintr-un număr mic de grupuri, pentru care se calculează scorurile de potrivire cu q.

Preprocesarea unei colecţii de N documente cu această metodă se face-n 2 paşi:

1. Se aleg la întâmplare $\sqrt{N}$ documente din colecţie. Documentele alese se numesc **lideri**.

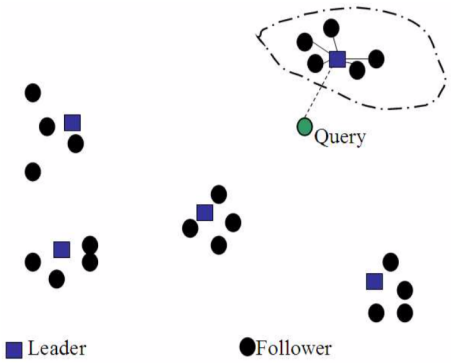
2. Pentru fiecare document d din celelalte documente se calculează liderul cel mai apropiat de d.

Documentele care nu sunt lideri se numesc **urmăritori**. Numărul estimat de urmăritori ai fiecărui lider ales aleator este $N/\sqrt{N} = \sqrt{N}$.

Procesarea unei cereri de interogare q se face astfel:

1. Se detectează documentul lider L cel mai similar cu q (se calculează $\sqrt{N}$ scoruri cosinusoidale)

2. Mulţimea de candidaţi A este mulţimea de urmăritori ai lui L.



2 Componentele unui sistem complet de extragere a informaţiilor

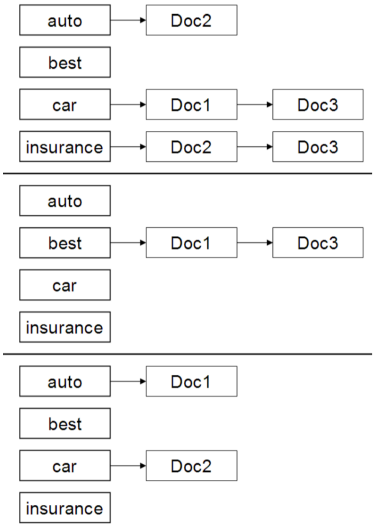
Un sistem rudimentar complet de extragere a informaţiilor se poate dezvolta pornind de la modelul de spaţiu vectorial pentru documente şi de la ideile prezentate de calcul al scorurilor de potrivire cu o cerere de informare.

2.1 Indecşi organizaţi pe niveluri

Idee de bază: Euristicile prezentate pot produce mulţimi de candidaţi A cu $|A| < K$. Indecşii organizaţi pe niveluri rezolvă această problemă, după cum este ilustrat în exemplul de mai jos:

- Pe nivelul 1 reţinem postări de documente d pentru termeni t cu $\text{tf}_{t,d} > 20$,
- Pe nivelul 2 reţinem postări de documente d pentru termeni t cu $\text{tf}_{t,d} > 10$,
- Pe nivelul 3 reţinem restul de postări de documente.

O astfel de situaţie este ilustrată în figura de mai jos.



Dacă eşuăm să obţinem K rezultate de pe nivelul 1, continuăm cu procesarea documentelor de pe nivelul 2, ş.a.m.d. Pe fiecare nivel, postările sunt ordonate după identificatorii de document.

2.2 Proximitatea termenilor din cereri

Idee de bază: sunt mai relevante documentele în care termenii din cerere au apariţii apropiate. De exmplu, fie q o cerere de informare conţine k termeni $t_1, t_2, \ldots, t_k$, şi ω lăţimea (adică, numărul de termeni) celui mai mic fragment

din d care conţine toţi termenii din q. Cu cât ω este mai mic, cu atât este documentul mai relevant petru cererea q.

Numeroase motoare de căutare (inclusiv Google) calculează măsuri de proximitate a termenilor din cereri q, care afectează scorul de potrivire al unui document cu q.

2.3 Construirea funcţiilor de analiză a cererilor şi de calcul al scorului

Stare de fapt: interfeţele majorităţii sistemelor de extragere a informaţiilor au operatori complecşi de exprimare a cererilor de informare.

- Aceşti operatori nu sunt gândiţi să fie folosiţi de utilizatori.
- De obicei, se presupune că un utilizator comunică o cerere de informare printr-un **text liber** (engl. *free text*), care este o secvenţă de termeni.
- Textul liber este analizat şi transformat în una sau mai multe cereri de informare exprimate cu operatorii complecşi al sistemului de extragere a informaţiilor.

De exemplu, cererea prin text liber `rising interest rates` a $K = 10$ potriviri, poate fi procesată astfel:

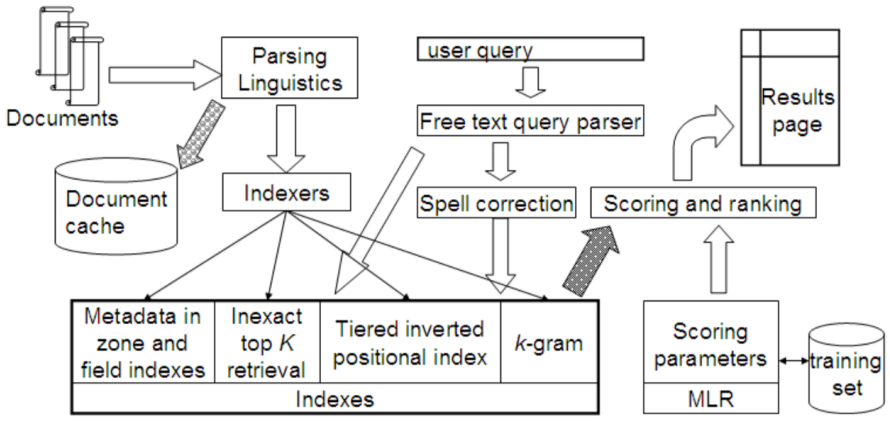
1. Mai întâi se caută documentele în care apare fraza `rising interest rates`, şi li se calculează rangul folosind reprezentarea ca vectori.
2. Dacă sunt mai puţin de K documente care conţin această frază, se caută documentele care conţin frazele `rising interest` şi `interest rates`. Din nou, se poate calcula rangul documentelor găsite folosind reprezentarea ca vectori.
3. Dacă nu am găsit încă K răspunsuri, se continuă cu cererile ce constau din termenii individuali ai cererii iniţiale.

Fiecare din cei 3 paşi produce o listă de documente cu scoruri aferente de potrivire. Fiecare scor calculat trebuie să combine contribuţii ce provin din valoarea scorului în spaţiu vectorial, scorul static de calitate, proximitatea în documente a termenilor din cerere, şi alc ti factori potenţiali — de exemplu, un document poate să apară în listele unor paşi diferiţi. Petnru a rezolva această problemă, trebuie definită o funcţie de agregare a scorurilor de potrivire produse de paşi diferiţi:

- Unele sisteme oferă utilizatorului un toolkit cu care pot să-şi configureze (1) modul de reducere a cererii din text liber în cereri concrete, şi (2) funcţia de agregare a scorurilor calculate de paşii diferic ti. Acest mod de configurare este adecvat pentru colecţii de documente cre nu se modifică frecvent: o dată configurat, este de aşteptat ca sistemul de extragere a informaţiilor să funcţioneze satisfăcător.

- Pentru colecţii de documente care se modifică frecvent, ca de exemplu documentele indexate de motoarele de căutare web, este imposibilă calibrarea manuală a funcţiilor de analiză a cererilor şi de calcul al scorului. În acest caz, se folosesc tehnici de machine learning.

2.4 Arhitectura unui model complet de căutare



Bibliografie

1. Capitolul 7: *Computing scores in a complete search system*
   din
   Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze: *An Introduction to Information Retrieval*. Ediţie online (c) 2009 Cambridge UP.
   `http://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf`