

1. KOLOKVIJ

1. Kaj je smisel entropijskega kodiranja? kaj potrebujemo, da lahko kodiramo?

Smisel entropijskega kodiranja je v tem da zakodiramo različno dolge kodne besede.

(dolžina kodnih besed povezana z verjetnostjo pojavitve simbola – bolj verjetni simboli imajo krajšo kodno besedo)

Entropija je **povprečna** lastna informacija podatkov. Entropija podaja **spodnjo mejo** števila bitov, s katerim lahko podatke predstavimo.

2. Z Huffmanom zakodiraj "KALAMARCKI"

Gradimo binarno drevo od spodaj navzgor:

1. vse simbole dodamo v vrsto V

2. ponavljamo, dokler v vrsti V ne ostane en sam element:

a) iz vrste vzamemo dva elementa e_1 in e_2 z najmanjšima verjetnostma

b) naredimo nov element drevesa n , ki ima za naslednika e_1 in e_2 in katerega verjetnost je vsota verjetnosti e_1 in e_2

c) vejama drevesa med n in e_1 in e_2 določimo oznaki 0 in 1

d) n dodamo v V

e) iz V izbrišemo e_1 in e_2

Kodo simbola dobimo iz oznak na vejah drevesa, ki vodijo do simbola.

3. Kako izvedemo iskanje Beseda1 OR Beseda2 (ilustriraj s primerom)

4. Podana je bila tabela, 3 besede in v kolikih dokumentih se posamezna pojavi (10, 100, 1000).

Število vseh dokumentov je 10 000. Koliko je idf?

$$\text{idf}_t = \log_{10} (N / \text{df})$$

5. Podano je bilo neko zaporedje cifer, ki se kodira z RLE. Koliko je kompresijsko razmerje?

Podobno kot prej ponovitve simbolov zamenjamo s **simbolom** in **številom pojavitev**

V najslabšem primeru, ko ni ponavljanj, dobimo precej večjo kodo (dve vrednosti namesto ene)

Primer:

111122233333311112222

stisnemo kot:

(1,4),(2,3),(3,6),(1,4),(2,4)

razmerje je 10:21

6. Kaj je BOM?

To uporabljamo za določanje vrstnega reda bytov (big/little endian) pri UTF-16 in UTF-32, lahko na začetku datoteka vsebuje t.i. BOM, preko katerega bralec lahko prepozna vrstni red

7. Narisana je bila shema obiskovanja spletnih strani in podana verjetnost obiska. Izračunaj PageRank.

Računanje PageRank

- obiskovalec naključno izbere neko stran, npr. z verjetnostmi
 $t = [0.2 \ 0.4 \ 0.1 \ 0.3]$
- po eni izbiri povezave ali sprehodu na naključen URL, bodo verjetnosti strani:
 $t' = t * M$
- po dveh izbirah povezav ali sprehodu na naključen URL:
 $t' = t * M * M$
- po n izbirah povezav ali sprehodih na naključen URL
 $t' = t * M^n$
- vrednost t' se z večanjem n ustali na stabilni vrednosti (konvergira)
- to je PageRank – bolj pomembne strani imajo višjo vrednost

8. Zakaj v vektorskem prostoru evklidska razdalja ni dobra?

ni dobra izbira

- dokumenti z različnimi besedami so daleč narazen, čeprav je del besed skupnih
- dokument, pri katerem se ena beseda večkrat pojavi je lahko dlje, kot dokument, ki nima te besede

9. Iskanje v vektorskem prostoru

Dokument predstavimo kot **vektor v vektorskem prostoru**

- dimenzija vektorja je enaka številu simbolov (besed) v vseh dokumentih

V vektorju so neničelne vrednosti pri simbolih, ki jih dokument vsebuje

Vsi vektorji tvorijo matriko

-term-document matrix

Ni važen vrstni red simbolov

-bag of words

Dokumenti pri spletnih indeksih imajo lahko milijone dimenzij

- ker je toliko različnih indeksiranih simbolov

Dokumente lahko razvrstimo glede na **razdaljo** med povpraševanjem in dokumentom

- povpraševanje obravnavamo kot dokument - vektor
- nič več Boolovega povpraševanja

Razdaljo raje merimo glede na **kot** med vektorji

- velikostni red uteži ni tako važen
- npr. če dokument d podvojimo, $d' = dd$, potem je kot med njima še vedno 0

Kosinusna **podobnost**

Povpraševanje predstavimo z tf-idf vektorjem

Dokumente tudi

Izračunamo kosinusno podobnost med povpraševanjem in dokumenti

Razvrstimo rezultate, vrnemo prvih K

V realnosti lahko uporabimo različne uteži za povpraševanja in dokumente

- dokumenti: npr. logaritmični tf, brez idf, normirani vektorji
- povpraševanje: logaritmični tf, z idf, brez normiranja

10. Razlika med text code in text encoding (tole ne vem če sem si prav zapomnila. Se kdo spomni?)

Koda znaka (*code position, code value, code point ...*)

- določa število za vsak znak iz repertoarja
- npr. ISO 10646 določa kode za znake "a", "!", "ä", and "%" kot 97, 33, 228, and 8240
- naboru vseh kod za repertoar lahko rečemo *coded character set (CCS)*

Kodiranje znakov (*character encoding*)

- določa kako se koda znaka dejansko zapiše v obliki bytov, npr niz a!ä% lahko zapišemo kot:

61 21 C3 A4 E2 80 B0 (UTF-8) ali 61 00 21 00 E4 00 30 20 (UTF-16)

11. Podani sta bili dve besedi (ne spomnim se kateri). Koliko je Levenshteinova razdalja? Opiši kako jo izračunamo.

Gradimo matriko vseh možnih pretvorb iz ene besede v drugo

V celicah matrike (i,j) seštevamo cene operacij potrebnih za transformacijo niza (1..j) v drugega (1..i)

če sta znaka (i,j) enaka:

$$m[i,j] = \min(m[i-1,j]+1, m[i,j-1]+1, m[i-1,j-1])$$

če znaka nista enaka

$$m[i,j] = \min(m[i-1,j]+1, m[i,j-1]+1, m[i-1,j-1]+1)$$

		F	A	S	T
	0	1	2	3	4
C	1	1	2	3	4
A	2	2	1	2	3
T	3	3	2	2	2

Razdalja med
fast in cat je 2

- vstavljanje – cena == 1
- brisanje – cena == 1
- zamenjava – cena == 1
- kopiranje – cena == 0

12. Kaj je pozicijski indeks?

v obrnjen indeks shranimo še položaj besed v dokumentu

- V katerem dokumentu je
- “to1 be2 or3 not4 to5 be6”

Pri iskanju moramo upoštevati tudi položaje besed in razdalje med položaji

- počasneje

Pozicijski indeks lahko uporabimo tudi za bolj splošna iskanja, npr. besed, ki so si blizu

- najdi vse pojavitve, ko sta si besedi *employment* in *place* največ 4 besede narazen

Velikost pozicijskega indeksa je precej večja kot običajnega obrnjenega indeksa

- velikost je odvisna od povprečne dolžine dokumentov
- v povprečju 2-4 krat večji od običajnega
- 35-50% velikosti originalnih dokumentov

13. Kako bi iskali po*ka? (nekaj takega)

Kako najdemo ka*

- torej vse simbole, ki se začnejo na ka

V drevesu enostavno, ker je urejeno

- poiščemo vse simbole w: ka <= w < kb

Kaj pa *ka?

- vzdržujemo še eno drevo s simboli v obratnem vrstnem redu

Kaj pa ka*ki?

- obravnavamo kot $ka^* \text{ AND } ^*ki$, torej naredimo presek vseh simbolov, ki jih najdemo v slovarju
- časovno zahtevno

Permuterm indeks omogoča hitrejše približno povpraševanje tipa ka^*ki

Ideja: besedo indeksiramo v vseh permutacijah

Povpraševanje obrnemo tako, da je $*$ na koncu:

Problem je, da se velikost indeksa precej poveča

- obstajajo tudi druge tehnike, npr. bigrami itn.

14. Kaj smisel/bistvo pri LZ77?

Uporablja se npr. v DEFLATE (ZIP)

Ko kodiramo, iščemo preteklo **najdaljšo** pojavitev **zaporedja** simbolov

Če najdemo, **shranimo**:

- razdaljo do trenutnega položaja
- dolžino zaporedja
- naslednji znak

Pretekle pojavitve iščemo znotraj **omejenega okna**, npr. 4096 simbolov

- prvih n znakov ne kodiramo

Slovar je **impliciten** kot referenca na preteklost

Dekodiranje je enostavno, samo sledimo referencam