UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Eksamen i: INF3800 Søketeknologi
Eksamensdato: Torsdag 9. juni 2011
Tid for eksamen: 14:30-18:30 (4 timer)
Oppgavesettet er på 4 sider
Vedlegg: Ingen
Tillatte hjelpemidler: Ingen

Kontroller at oppgavesettet er komplett før du begynner å besvare spørsmålene.

Oppgavesettet har 6 deloppgaver. Du skal besvare 5 (og bare 5) av dem. Du kan selv velge hvilke 5. Hver av de 5 valgte deloppgavene vil telle like mye.

OPPGAVE 1: TF-IDF

Kurt har fire dokumenter med følgende innhold (uten anførselstegn):

> d_1: "*jeg liker skilpadder jeg*"
> d_2: "*jeg digger skilpadden din jeg*"
> d_3: "*jeg er en skilpadde*"
> d_4: "*ingen digger elger*"

Anta normal mellomrom-basert tokenisering, at lemmatisering skjer ved at substantiver reduseres til baseform, og at TF-IDF vekter beregnes på følgende måte:

> $\mathrm{TF}(w, d) = occ(w, d) \,/\, \max\{occ(w', d) \text{ der } w' \text{ forekommer i } d\}$
> $\mathrm{IDF}(w) = 1 - (df(w) \,/\, 4)$

Her er $occ(w, d)$ en funksjon som teller antall forekomster av ordet w i dokumentet d, og $df(w)$ er en funksjon som teller antall dokumenter i indeksen som inneholder w.

(a) Kurt vil lage en ikke-posisjonell invertert indeks over de tre dokumentene. Tegn opp alle postinglistene i indeksen. Hvert innslag i postinglistene skal annoteres med en tilhørende TF-IDF verdi.
(b) Kurt gjør en spørring "*digger du skilpadder*" (uten anførselstegn) mot indeksen og har implementert det slik at han får rangert dokumentene i synkende rekkefølge etter deres cosinus likhet med spørringen. Hva evaluerer dette likhetsmålet til for dokument d_2? (I mangel av kalkulator er det tilstrekkelig å angi uttrykket, med tall, som skal evalueres.) Hvordan kan Kurt forenkle beregningen på en måte som garanterer at dokumentene blir rangert i samme rekkefølge?

OPPGAVE 2: INDEKSKOMPRIMERING

Som en del av prosessen med å komprimere en søkeindeks inngår gjerne det å kode verdien av gapene mellom påfølgende postings ved hjelp av en passende komprimeringsteknikk.

(a) Hvordan ville du komprimert verdien 1337 ved hjelp av VB koding? Hvordan ville du komprimert verdien 13 ved hjelp av gamma koding?
(b) Kurt har en større søkeindeks, og lurer på om han skal velge VB koding eller gamma koding. Hvilket råd vil du gi Kurt?
(c) Forklar kort hvordan indekskomprimering innvirker på ytelsen til et søkesystem.
(d) Entropi er et sentralt begrep innenfor komprimering generelt. For hvilken diskret distribusjon over N verdier har entropi sitt maksimum, og hva er da denne maksimumsverdien?

OPPGAVE 3: K-GRAM RANKING

Kurt har implementert nok en søkeindeks, denne gang en *k*-gram indeks med *k*=3. Han ønsker å komme frem til en passende rangeringsfunksjon.

(a) Kurt prøver først å bruke Jaccard koeffisienten som rangeringsfunksjon. For en spørring "*foobar*" (uten anførselstegn) og et dokument "*foodbar*" (uten anførselstegn), hva blir da verdien med *k*=3? Du trenger ikke å innføre spesielle markører for starten eller slutten av strengene når du genererer 3-gram.
(b) Kurt bestemmer seg for å komme opp med en alternativ rangeringsfunksjon. Han vil komme frem til et uttrykk som for et dokument som er *N* tegn langt angir sannsynligheten for at en vilkårlig posisjon i dokumentet er "truffet" av minst ett *k*-gram fra spørringen. For eksempel, 3-grammet "*foo*" (uten anførselstegn) kan sies å "treffe" dokumentet "*foodbar*" (uten anførselstegn) i posisjonene 0, 1 og 2. Hjelp Kurt å komme frem til en slik formel uttrykt ved hjelp av *N*, *k*, og *h*, der *h* er antall *k*-gram fra en spørring som "treffer" et dokument som er *N* tegn langt. For eksempel, for spørringen *"foobar"* (uten anførselstegn) og dokumentet *"foodbar"* (uten anførselstegn) har vi *N*=7 og med *k*=3 har vi *h*=2. Du kan gjøre de noe urealistiske antagelsene at alle *k*-gram er uavhengige, og at sannsynligheten for treff på en gitt posisjon i dokumentet er helt uniformfordelt (inkludert for starten og slutten av dokumentet.)

OPPGAVE 4: EVALUERING AV RELEVANS

Følgende liste med R og N elementer representerer relevante (R) og ikke-relevante (N) dokumenter returnert i rangert rekkefølge fra et søkesystem for en gitt spørring. Listen leses fra venstre mot høyre, med det høyest rangerte dokumentet (det dokumentet systemet mener er mest relevant) lengst til venstre. Listen viser 6 relevante dokumenter. Anta at systemet har indeksert i alt 10000 dokumenter, hvorav 8 er relevante for denne spørringen.

RRNNN NNNRN RNNNR NNNNR

(a) Hva er presisjonen til dette systemet for denne spørringen på de topp 20 resultatene?
(b) Hva er den interpolerte presisjonen ved 33% recall?
(c) NDCG er et populært mål som søkemotorselskaper bruker for å måle relevans. Forklar kort hva dette målet innebærer.

OPPGAVE 5: SUFFIX ARRAYS

(a) Konstruer og tegn et felles suffix array for de to strengene "*hermetisk*" og "*mett*" (uten anførselstegn).
(b) Vis hvordan du kan bruke denne arrayen til å gjøre et effektivt substreng-søk etter strengen "*met*" (uten anførselstegn).

OPPGAVE 6: YMSE OM ORDLISTER

(a) Kurt har en stor ordliste med mange millioner innslag. Som en del av et system for dokumentprosessering vil han så raskt som mulig gjenkjenne alle ord og fraser i et dokument som også finnes i ordlista. For eksempel, dersom "*foobar*" både er i ordlista og dokumentet så skal dette rapporteres. Beskriv hvordan Kurt burde representere ordlista og utføre søket.
(b) Kurt har også en annen stor ordliste som inneholder alle former av de fleste vanlige norske ord. Som en del av et system for å gjenkjenne skrivefeil, ønsker Kurt å gjøre spørringer mot denne slik at han for en tilfeldig spørring q raskt kan få returnert alle innslag q' i ordlista som har edit distanse k eller mindre fra q. Du kan anta at k er liten. Beskriv hvordan Kurt burde representere ordlista og utføre søket.