NTNU
Kunnskap for en bedre verden

Institutt for datateknikk og informasjonsvitenskap (IDI)

Eksamensoppgave i TDT4117 Informasjonsgjenfinning

Faglig kontakt under eksamen: Heri Ramampiaro
Tlf.: 73591459

Eksamensdato: 07.12.2016
Eksamenstid (fra-til): 09:00-14:00 (4 timer)
Hjelpemiddelkode/Tillatte hjelpemidler: D – Kun godkjent kalkulator tillatt

Annen informasjon:

Målform/språk: SENSURVEILEDNING
Antall sider (uten forside): 2
Antall sider vedlegg: 0

Informasjon om trykking av eksamensoppgave
Originalen er:
1-sidig ☐ **2-sidig** ☐
sort/hvit ☐ **farger** ☐
skal ha flervalgskjema ☐

Kontrollert av:

Dato Sign

*Svar **kort og konsist** på alle spørsmålene.*
Les igjennom alle oppgavesett før du begynner å løse oppgavene. Forklar kort eventuelle antakelser der du mener oppgaveteksten ikke er fullstendig. Lykke til!

I alle regne og tegneoppgaver ***kreves mellomregning og forklaring for å få full pott*** for å vise metodeforståelse. Metodeforståelse er minst like viktig som å vise til detaljkunnskap.

Oppgave I (25%)

1. Her er en påstand: ”Den boolske likhetsmodellen (boolean similarity model) er egentlig ikke en informasjonsgjenfinningsmodell men en datagjenfinningsmodell”. Forklar hvorfor dette kan være sant.

   Basert på at den ***ikke* rangerer** søkeresultater men bruker ”**full match**”, kan det diskuteres om boolske modellen egentlig er en IR-modell.
   Må nevne både om rangering og matching for å få full pott.

2. ”Page Rank” er en rangeringsmetode som kan brukes til å rangere søkeresultat. Forklar når ”Page Rank” ikke kan brukes.

   Page Rank egner seg bare når man har linker mellom dokumenter som for eksempel i websider. Rangeringen er da bl.a. basert på å finne sannsynligheten for at man skal hoppe til en side fra en annen. *Må vise forståelse for Page Rank som metode for å få full pott.*

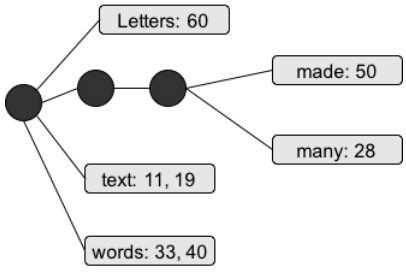
3. Forklar med eksempel og figur forskjellene mellom ”suffix trie” og ”vocabulary trie”.

   Suffix trie er i seg selv en indekseringsmetode mens vocabulary trie brukes i forbindelse med konstruksjonen av indeks i ”inverted index”-indekseringsmetoden.

   Ekempel:

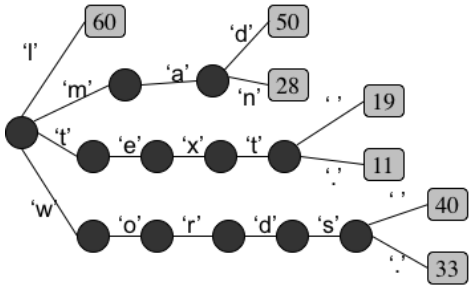
   1 6 9 11 17 19 24 28 33 40 46 50 55 60
   This is a text. A text has many words. Words are made from letters.

   Vocabulary trie lager vi basert på “term occurrencies” og må ha med termene (vocabulary)



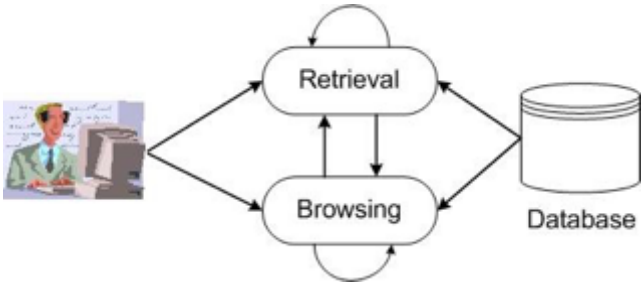
   Suffix trie lager vi basert på følgende suffix-strenger:

String **11**: text. A text has many words. Words are made from letters.
String **19**: text has many words. Words are made from letters.
String **28**: many words. Words are made from letters.
String **33**: words. Words are made from letters.
String **40**: Words are made from letters.
String **50**: made from letters.
String **60**: letters.



Suffix trie

4. Tegn et blokkdiagram (med firkanter og piler) som forklarer hvordan informasjonsgjenfinningsprosessen er bygd opp. Tips: Dette er ikke tekstoperasjoner.



Her forventes at studentene tegner opp en blokkdiagram som forklarer hvordan IR- søkeprosessen fungerer. Forklaringen skal bygges rundt brukerens informasjonsbehov. *Enkel forklaring av prosessen for å få full pott.*

5. To bilder med størrelse 4x4 med inneholder 4 forskjellige pikselfarger (C1, C2, C3, C4) fordelt på følgende måte:
   Bilde 1: 2 av C1, 3 av C2, 6 av C3 og 5 av C4.
   Bilde 2: 4 av C1, 1 av C2, 8 av C3 og 3 av C4.
   Finn forskjellen (eller avstanden) mellom de to bildene ved hjelp av bildenes *histogram*.

   Histogrammene er som følgende:

   H1 = {2, 3, 6, 5} og H2 = {4, 1, 8, 3} for hhv. Bilde 1 og Bilde 2.

   Avstanden beregnes ved å summere forskjellen på hver "bin" i histogrammen, dvs.

   $$d(H_1, H_2) = \sum_{i=0}^{n} |H_{1i} - H_{2i}|, \qquad n = 3$$

   d(H1, H2) = |2-4| + |3-1| + |6-8| + |5-3| = **8**

Oppgave II (30%)

Anta vi har følgende tekst:

"`Political differences can break family ties, but family heals`".

Gjør (og forklar) ellers de antakelsene du finner nødvendig når du svarer på følgende spørsmål.

1. Utfør tekstoperasjonene: leksikalanslyse (lexical analysis), stoppordfjerning, stemming. Hvilke liste av termer sitter man igjen med?

Det forventes at man forklarer først hva disse tekstoperasjonene er. Etter å ha utført tekstoperasjonene sitter vi igjen med: politic, difference, can, break, family, tie, heal. (differ i stedet for difference er også riktig).

2. Konstruer en signaturfil av teksten over. Anta at du kan dele teksten i 3 blokker. Du må ellers gjøre dine egne antakelser angående **hash-funksjon**, og liknende.

   Anta vi har følgende 4-bits hash-funksjon:

   f(political)= 0100 0001,
   f(differences)= 0101 0010,
   f(can)= 0110 0011,
   f(break)= 1101 0100,
   f(family)= 1100 0101,
   f(ties)= 0110 0110,
   f(heals)= 0100 0111.

   Siden vi deler setningen i tre blokker er det nøyaktig tre ord i hver blokk. Vi får dermed følgende *signatur basert på bitvis OR-operasjoner av hash-funksjonene i hver blokk*.

   Signartur for blokk 1: 0100 0001 OR 0101 0010 OR 0110 0011 = **0111 0011**

   Signatur for blokk 2: 1101 0100 OR 1100 0101 OR 0110 0110 = **1101 0111**

   Signatur for blokk 3: 1100 0101 OR 0100 0111 = **1100 0111**

3. Konstruer invertert-indeks (inverted index) av teksten over.

   Antar at man har utført tekstoperasjonene som beskrevet over. Her holder det at studenten viser oppbyggingen av ”inverted file” som følgende:

```
1          11           23  27    33     40    46  50     57
Political differences can break family ties, but family heals
```

Vocabulary	Occurrences
break	27
can	23
difference	11
family	33, 50
heal	57
politic	1
tie	40

4. Lag et ”suffix array” av teksten over. Hvorfor er denne indekseringsmetoden mindre egnet enn ”supra index”?

   Må konstruere suffix strengen først:

```
           1          11           23  27    33     40    46  50     57
Streng 1:  Political differences can break family ties, but family heals
Streng 11: differences can break family ties, but family heals
Streng 23: can break family ties, but family heals
Streng 27: break family ties, but family heals
Streng 33: family ties, but family heals
```

```
String 40: ties, but family heals
String 50: family heals
String 57: heals

Basert på dette blir Suffix Array:
```

27	23	11	33	50	57	1	40

NB: *Må ha med at man konstruere "suffix array" fra "suffix strings" for å få full pott.*

Største problemet med denne indekseringsmetoden er at den fører til mange tilfeldige diskaksesser (random disk access) når array'et blir stort. Dette fører igjen til bl.a. dårlig ytelse. Pga. blokkstrukturen til "supra index" unngår vi dette (så lenge man har plass i minne til å lagre pekere til blokkene). *Forklaring som viser til forståelse av "supra index" forventes for å få full pott.*

Oppgave III (25%)

Ola skal evaluere sitt informasjonsgjenfinningssystem (IR-system). Han tester forskjellige evalueringsmetrikker. Han setter en grense på 20 (rangerte) resultatdokumenter for hver spørring. Han har i alt 4 spørringer (q1 – q4) med fasit han skal bruke til evalueringen. Når han sjekker fasiten finner han relevante dokumenter på følgende plasser i resultatlista for hver spørring:
q1: 1, 3, 4, 5, 6, 10, 12, 15.
q2: 2, 4, 5, 12, 13.
q3: 1, 2, 4, 6, 13, 18.
q4: 1, 4, 7, 8, 9, 10, 14, 20.

For q1 er totalt antall relevante dokumenter 10. For q2 er dette 15. For q3 er det totalt 16 relevante dokumenter og for q4 er det 13.

Siden alle oppgavene her er mest beregningsoppgaver holder ikke at studentene kommer med slutt resultat uten å vise hvordan man kommer til svaret.

1. Beregn presisjon (precision) og recall for hver spørring.

   Her skal man regne ut precision og recall for alle spørringene q1, q2, q3, q4.

   P1 = 8/20=0.4, R1 = 8/10=0.8;
   P2 = 5/20=0.25; R2 = 5/15=0.33;
   P3 = 6/20=0.30, R3 = 6/16=0.38;
   P4 = 8/20=0.40; R4 = 8/13=0.62;

   Formelen på Precision og Recall må være med for å få full pott.

2. Hva blir R-precision for hver av spørringene.

   R-precision er P@R hvor R er antall totale relevante dokumenter for en gitt spørring.

   R-precision1 = 6/10=0.6;
   R-precision2 = 5/15=0.33;
   R-precision3 = 5/16=0.31;
   R-precision4 = 6/13=0.46;

3. Bruk resultatene til q1 til å illustrere hvordan du kan beregne precision- og recall-punktene.

   Her skal studentene beregne

   P@1, P@3, P@4, P@5, P@6, P@10, P@12, P@15; og

   R@1, R@3, R@4, R@5, R@6, R@10, R@12, R@15 som følgende:

Resultatliste	P	R	Utregning
1	**1.00**	**0.10**	**P=1/1, R=1/10**
2			
3	**0.67**	**0.20**	**P=2/3, R=2/10**
4	**0.75**	**0.30**	**P=3/4, R=3/10**
5	**0.80**	**0.40**	**P=4/5, R=4/10**
6	**0.83**	**0.50**	**P=5/6, R=5/10**
7			
8			
9			
10	**0.60**	**0.60**	**P=6/10, R=6/10**
11			
12	**0.58**	**0.70**	**P=7/12, R=7/10**
13			
14			
15	**0.53**	**0.80**	**P=8/15, R=8/10**
16			
17			
18			
19			
20			

   Studentene kan velge å illustrer svaret med tabell (som over) eller graf, men *det er viktig at de viser hvordan de har kommet til punktene*.

4. Hva blir Mean Average Precision (MAP)-verdien for Ola sitt IR-system? NB: For å få poeng må du vise hvordan du kom fram til svaret ditt.

   Svaralternativ 1 (jfr. Læreboka):

Recall Pts	P1	P2	P3	P4
0	1.00	0.60	1.00	1.00
1	1.00	0.60	1.00	0.60
2	0.67	0.60	1.00	0.60
3	0.75	0.38	0.75	0.60
4	0.80	0.38	0.75	0.60
5	0.83	0.00	0.75	0.50
6	0.60	0.00	0.00	0.40

7	0.58	0.00	0.00	0.00
8	0.53	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00
MAP_i	**0.62**	**0.23**	**0.48**	**0.39**

MAP kan beregnes ved først å finne precisions på hvert recall-nivå (recall level) med interpolering (se tabellen over). Deretter beregnes snittet av precisions for hver spørring. Dette gir MAP_i-verdi, i=1, 2, 3, og 4 som i tabellen. MAP for Olas system blir snittet av alle MAP-verdiene som følgende:

$MAP = (MAP_1 + MAP_2 + MAP_3 + MAP_4)/4 = (0.62 + 0.23 + 0.48 + 0.39)/4 =$ **0.43**

Svaralternativ 2:

Avg P (average precision) for en spørring kan regnes ut ved å ta snitte av alle precision-verdi for hvert relevant treff (regnes ut fra precision-verdiene fra precision-recall-punktene). MAP er da snittet av Avg P for alle 4 spørringene (se tabellen under).

	q1		**q2**		**q3**		**q4**	
Resultatliste	P	R	P	R	P	R	P	R
1	1.00	0.10			1.00	0.06	1.00	0.08
2			0.50	0.07	1.00	0.13		
3	0.67	0.20						
4	0.75	0.30	0.50	0.13	0.75	0.19	0.50	0.15
5	0.80	0.40	0.60	0.20				
6	0.83	0.50			0.67	0.25		
7							0.43	0.23
8							0.50	0.31
9							0.56	0.38
10	0.60	0.60					0.60	0.46
11								
12	0.58	0.70	0.33	0.27				
13			0.38	0.33	0.38	0.31		
14							0.50	0.54
15	0.53	0.80						
16								
17								
18					0.33	0.38		
19								
20							0.40	0.62
Avg P	0.72		0.46		0.69		0.56	

MAP: **0.61**

Oppgave IV (20%)

Svar rett/galt med begrunnelse på følgende utsagn. Hvert **riktig** og **begrunnet** svar belønnes med **2** poeng. **Feilsvar**, **ubegrunnet** eller **ingen svar** gir ingen poeng.

1. Fargehistogrammer kan brukes i forbindelse med gjenfinning av både bilder og videosnutter.
   (Rett/Galt)
   Rett: Kan brukes som feature til å finne avstanden mellom to bilder, eller to rammer (frames) i videosnutter.

2. Google bruker ikke ”stemming” fordi stemming ikke passer til web-søk generelt.
   (Rett/Galt)
   Rett: Stemming koster mye å gjennomføre. Stemming øker generelt recall, og kan minke precision. Med web-søk er hovefokuset å få best mulig precision-verdi.

3. Søkemotorer med ”Harvest”-arkitektur er en variant av distribuert web-søkemotor arkitektur.
   (Rett/Galt)
   Rett: Harvest er navnet på et av distribuert arkitektur for web-søkemotor. Denne består av ”gatherers” og distribuerte ”brokers”.

4. Thesaurus-bygging er naturlig del i automatisk lokal analyse (automatic local analysis), og bruker hele dokumentsamlingen til å gjøre dette.
   (Rett/Galt)
   Galt: Local analysis bruker kun de returnerte dokumentene fra en spørring til å bygge thesaurus.

5. Den største forskjellene mellom ”Probabilistic Similarity Model” og ”Language Model” er måten sannsynligheten blir beregnet.
   (Rett/Galt)
   Rett: ”Probabilistic Similarity Model” bruker relevans som basis for å beregne sannsynlighet, mens ”Language Model” bruker dokumentmodell generering som basis til å bergne sannsynlighet.

6. Hvis man ser på tekst som et multimediaobjekt ville indekstermene (index terms) være ”features”.
   (Rett/Galt)
   Rett: Index terms skal representer *innholdet* i et dokument, kan de sees på som ”features”.

7. En SQL-database er for datagjenfinning og derfor kan den ikke brukes til bildegjenfinning.
   (Rett/Galt)
   Galt: Bilde-metadata kan brukes som grunnlag for bildegjenfinning. Metadata er strukturert informasjon som kan lagres i en SQL-database. Derfor kan også SQL-database benyttes til bildegjenfinning også.

8. Komprimering kan ikke alltid brukes i informasjonsgjenfinning siden dataene da alltid må dekorprimeres først.
   (Rett/Galt)
   Galt: Noen komprimeringsalgoritmer tillater søk i komprimert data.

9. Treet for Huffman-kode er en spesial-versjon av et indekskomprimeringstre.
   (Rett/Galt)
   Galt: Huffman-kode er en statistikkbasert generell komprimeringsmetode.

10. Videogjenfinning kan bruke ”features ” fra bildegjenfinning.
    (Rett/Galt)
    Rett: Videoer er sekvenser av bilder og kan derfor bruke features fra bildegjenfinning.