Løsningsforslag til eksamen i IN2110 vår 2023

- Husk på at løsningsforslagene som står her er nettopp det: forslag. Det finnes typisk flere måter å gjøre ting på.

1 Klassifikasjon (21 poeng)

1.1 Vektor-typer (5 poeng)

- Tradisjonelle ordvektorer er gjerne høy-dimensjonale, er såkalt "sparse", dvs. at de har få aktive (ikke-null) trekkverdier, og en gitt dimensjon / trekk tilsvarer en diskré egenskap (f.eks. antall forkomster i nærheten av ordet "bake").
- Embeddings har typisk lavere dimensjonalitet, er såkalt "dense", dvs. at de har mange eller kun aktive trekk (altså verdier != 0), og informasjonen er distribuert, dvs. at et gitt trekk ikke tilsvarer en gitt egenskap ved et ord, i stedet er informasjonen kodet i totaliteten av trekk-aktivering i hele vektoren.
- Mens tradisjonelle vektorer gjerne er basert på opptellinger av manuelt definerte trekk-typer, så er embeddings typisk representasjoner som har blitt lært som del av en annen (typisk nevral) modell, f.eks. en nevral språkmodell for å predikere ord i kontekst.
- Begge representasjoner reflekterer den distribusjonelle likheten til ordene i treningskorpuset, altså hvor like kontekster to gitte ord forekommer i.

1.2 P og R 1 (5 poeng)

Automatisk.
-1 poeng pr feil svar ned til 0
Rett svar gir TP:2, TN:7, FP:1, FN:0

1.3 P og R 2 (6 poeng)

P = (TP) / (TP+FP).
P = 2/(2+1) = 0,67.
(Utregningen skal anses som rett hvis man har brukt sine egne tall rett, selv om man kan ha gjort en feil i forrige oppgave)

R = (TP) / (TP+FN).
R = 2 / (2+0) = 1 .

1.4 P og R 3 (5 poeng)

(TP) / (TP+FP) = (TP) / (TP+FN)
FP må være lik FN
Hverken TP, FP eller FN kan være null
Vi har to grønne.
TP, FP og FN må være 1

2 Klassifikatorer (16 poeng)

- **A)** Rocchio vs kNN: Rocchio baserer seg på avstand til nærmeste centroide og fungerer best for situasjoner der hver klasse er distribuert som sfæriske regioner av tilnærmet lik størrelse. Den vil her dermed kunne risikere å feilaktig klassifisere test-punktet vårt som trekant, siden denne klassen vil ha den nærmeste centroiden. kNN derimot er sensitiv til den lokale distribusjonen av eksempler nærmest test-punktet og vil dermed enklere kunne klassifisere punktet riktig, i hvert fall for en relativt lav verdi for k.

- **B)** Rocchio vs kNN: Begge algoritmer kan her forventes å klassifisere punktet riktig. Som nevnt over antar Rocchio sfæriske regioner av tilnærmet lik størrelse, noe som nettopp er tilfelle her. Videre er kostnaden ved å utføre klassifikasjon med Rocchio lavere enn for kNN: mens vi for Rocchio kun trenger å beregne avstanden mellom test-punktet og de to klasse-centroidene, må vi for kNN beregne de parvise avstandene mellom test-punktet og samtlige treningseksempler. Rocchio er derfor å foretrekke i dette tilfellet.
- **C)** Vi ser at klassene ikke er lineært separerbare. Siden logistisk regresjon er lineær klassifikator er den derfor ikke egnet for dette problemet. kNN derimot kan utføre ikke-lineær klassifikasjon og vil enkelt kunne gjøre riktig prediksjon i dette tilfellet, igjen med antakelse om en relativt lav verdi for k.
- **D)** Log. reg. vs Rocchio: Vi ser at problemet er lineært separerbart, og siden både logistisk regresjon og Rocchio er lineære modeller kunne man kanskje tro at begge vil være like egnet. Men, som nevnt for deloppgave A) så baserer Rocchio seg på avstand til nærmeste centroide og fungerer best for situasjoner der hver klasse er distribuert som sfæriske regioner av tilnærmet lik størrelse. Dette kriteriet oppfylles ikke i dette tilfellet siden regionene er av svært ulik størrelse. Rocchio vil her dermed kunne risikere å feilaktig klassifisere test-punktet vårt som trekant, siden denne klassen vil ha den nærmeste centroiden, selv om punktet naturlig ser ut til å falle innenfor regionen til sirkel-klassen. En logistisk regresjons-modell derimot vil enkelt kunne lære en lineær grense som korrekt separerer klassene og korrekt klassifiserer test-punktet.

3 Logistisk regresjon (8 poeng)

3.1 Spørsmål 1 (4 poeng)

I multinomial (=multi-klasse) logistisk regresjon har man en beslutningsregel per klasse. Vi har en separat vektvektor og skjæringspunkt for hver klasse. Sannsynligheten for å tilhøre en klasse er definert som $e^{\mathbf{w_i x}+b_i}$ normalisert for å få en sannsynlighetsfordeling over alle klassene. Normaliseringen gjøres ved å ta i bruk *softmax* funkjsonen, som er definert slik: $softmax(z) = \frac{e^z}{\sum_{j=1}^{K} e^{z_j}}$ med K = antall klasser.

Formelen for multinomial logistisk regresjon er derfor:

$$P(y = i|\mathbf{x}) = \text{softmax}(\mathbf{w_i x} + b_i) \tag{1}$$

Man må forklare at:

- man har en separat vektvektor og skjæringspunkt per klasse,
- softmax brukes til å normalisere til en sannsynlighetsfordeling.
- I tillegg må de gi hele formelen.

3.2 Spørsmål 2 (4 poeng)

Tapsfunkjsonen uttrykker hvor mye "feil" modellen gjør om den predikerer $\hat{y}$ mens den ekte verdi (fra fasiten i treningsettet) er y. En vanlig tapsfunkjson for klassifisering er cross-entropy. Treningen av modellen handler dermed om å minimere tapet på treningssettet ved å gradvis endre modelparametrene for å redusere tapet. Man bruker optimeringsalgorithmer som Gradient Descent for å finne hvordan disse parametrene skal endres skritt etter skritt, inntil man kommer på en bunn hvor det ikke lenger er mulig å redusere tapet ytterliggere.

Man må kunne forklare at

- tapsfunksjonen handler om å sammenligne outputten fra modellen med fasiten,
- at tapet representer hvor mye "feil" modellen gjør for et bestemt datapunkt hvor vi har tilgang til fasiten – eller, med andre ord, avstanden mellom outputten og fasiten,
- og at den brukes til å optimere parametrene i modellen slik at tapet blir etterhvert så liten som mulig.

4 Hidden Markov Models (12 poeng)

4.1 Transisjonsmodell (4 poeng)

Vi kan estimere transisjonsmodellen ved å telle transisjonene fra en tilstand til en annen:

		Tilstand dag i	
		Kjølig	Varm
Tilstand dag $i-1$	Kjølig	7	3
	Varm	4	4

NB: det er helt greit om man også legger inn et startsymbol i denne transisjonsmodellen (siden vi ikke har spesifisert hva som skulle gjøres med starten på sekvensen).

Og transisjonsmodellen blir da etter normalisering:

		Tilstand dag i	
		Kjølig	Varm
Tilstand dag $i-1$	Kjølig	0.7	0.3
	Varm	0.5	0.5

4.2 Emisjonsmodell (4 poeng)

Vi starter med å telle emisjonene for hver tilstand:

		Observasjon dag i		
		Regn	Overskyet	Sol
Tilstand dag i:	Kjølig	4	8	1
	Varm	0	2	6

og normaliser resultaten for å få emisjonsmodellen:

		Observasjon dag i		
		Regn	Overskyet	Sol
Tilstand dag i:	Kjølig	$4/13$	$8/13$	$1/13$
	Varm	0	0.25	0.75

4.3 Smoothing (4 poeng)

Vi benytter oss av Laplace smoothing med å legge til $\alpha = a$ til alle tallene.

		Observasjon dag i		
		Regn	Overskyet	Sol
Tilstand dag i:	Kjølig	5	9	2
	Varm	1	3	7

og det gir oss en ny emisjonsmodell:

		Observasjon dag i		
		Regn	Overskyet	Sol
Tilstand dag i:	Kjølig	$5/16$	$9/16$	$1/8$
	Varm	$1/11$	$3/11$	$7/11$

5 Dependenssyntaks og parsing (21 poeng)

5.1 Dependenssyntaks 1 (5 poeng)

Dependensgrammatikk beskriver syntaktisk funksjon i en setning ved hjelp av rettede kanter mellom to ord. Kantene har merker som angir syntaktisk funksjon. Frasestrukturen beskriver strukturelle segmenter og undersegmenter i en setning i et frasetrukturtre der elementene har strukturelle kategorier, mens kanten ikke er merket.
Syntaktisk funksjon er sentralt for dependensgrammatikk, mens den ytre oppbygging, struktur, er sentral i frasestrukturgrammatikk.
Eksempler:

- Dependensgrammatikk kjennetegnes ved rettede relasjoner (bileksikale") mellom to ord: et hode og en dependent.
- Relasjonene er merket med funksjoner, som feks subjekt og adverbial.
- Dependensgrammatikk beskriver syntaktisk funksjon snarere enn form.
- I frasestrukturgrammatikk er fraser (sammenhengende setningssegmenter) sentrale. Disse beskriver strukturen i setningen.
- Frasene i frasestrukturgrammatikk merkes med strukturelle kategorier, feks NP, VP, som beskriver frasene.

5.2 Dependenssyntaks 2a (3 poeng)

Automatisk: 3 hvis rett, 0 ellers. Ikke projektiv: Kanten fra *Det* til *fekk*

5.3 Dependenssyntaks 2b (3 poeng)

Automatisk: 3 hvis rett, 0 ellers. Grafen er syklisk: Kanten fra *ganske* til *skrivestil*

5.4 Dependensparsing (10 poeng)

Stack	Buffer	Action	New Relations	Indekser	Ord
ROOT	*Kamsen vert laga med lever og smakar pyton spør du meg*	Shift		1	*Kamsen*
ROOT, Kamsen	*vert laga med lever og smakar pyton spør du meg*	Shift		2	vert
ROOT, Kamsen, vert	*laga med lever og smakar pyton spør du meg*	Left-arc	(2,1)	3	laga
ROOT, vert	*laga med lever og smakar pyton spør du meg*	Shift		4	med
ROOT, vert, laga	*med lever og smakar pyton spør du meg*	Shift		5	lever
ROOT, vert, laga, med	*lever og smakar pyton spør du meg*	Shift		6	og
ROOT, vert, laga, med, lever	*og smakar pyton spør du meg*	Right-arc	(4,5)	7	smakar
ROOT, vert, laga, med	*og smakar pyton spør du meg*	Right-arc	(3,4)	8	pyton
ROOT, vert, laga	*og smakar pyton spør du meg*	Right-arc	(2,3)	9	spør
ROOT, vert	*og smakar pyton spør du meg*	Shift		10	du
ROOT, vert, og	*smakar pyton spør du meg*	Shift		11	meg
ROOT, vert, og, smakar	*pyton spør du meg*	Left-arc	(7,6)		
ROOT, vert, smakar	*pyton spør du meg*	Shift			
ROOT, vert, smakar, pyton	*spør du meg*	Right-arc	(7,8)		
ROOT, vert, smakar	*spør du meg*	Right-arc	(2,7)		
ROOT, vert	*spør du meg*	Shift			
ROOT, vert, spør	*du meg*	Shift			
ROOT, vert, spør, du	*meg*	Right-arc	(9-10)		
ROOT, vert, spør	*meg*	Shift			
ROOT, vert, spør, meg		Right-arc	(9,11)		
ROOT, vert, spør		Right-arc	(2,9)		
ROOT, vert		Right-arc	(0,2)		
ROOT					

Oppgaveteksten angir eksplisitt at det er tre alternative handlinger, og 'REDUCE' som handling er en feil. Relasjonene kan enten angis som nye relasjoner som her, eller som et sett av relasjoner som vokser hver gang. Det er ikke bedt om at relasjonene merkes med syntaktisk form, og hvis det er lagt til så gir det verken pluss eller minus.
En korrekt besvarelse innebærer: Rett valgt handling, rette endringer på stack og buffer, rette relasjoner registrert.

6 Maskinoversettelse (8 poeng)

6.1 Spørsmål 1 (3 poeng)

Et parallellkorpus er et korpus (det vil si en samling tekstdokumenter) hvor det samme innholdet er tilgjengelig på minst 2 språk, og hvor setningene er sammenstilt ("aligned"), slik at man kan se hvordan en setning i språk A er oversatt i språk B. Eksempler av slike parallellkorpus er filmtekstinger, forhandlinger i EU-parlamentet, Bibelen, osv.

Svaret må også nevne at et parallellkorpus har det samme innholdet tilgjengelig på minst 2 språk og at setningene i korpuset er sammenstilt.

6.2 Spørsmål 2 (5 poeng)

En stor fordel med BLEU er at den er en automatisk måling som ikke krever menneskelige vurderinger (men er korrelert med menneskelige vurderinger). Den er lett og beregne og krever kun tilgang til fasitoversettelser.

Men BLEU har også betydelige svakheter: den er veldig overfladisk og tar ikke hensyn til semantikk. For eksempel vil det å oversette en negasjon ha en stor innflytelse på hva som formidles i en setning, men vil ikke påvirke BLEU-skoren i nevneverdig grad. BLEU har også utfordringer med å ta hensyn til synonymer eller morfologiske varianter av et ord, siden den kun baserer seg på overlap mellom n-grams i outputten og fasiten..

(Det kan finnes andre fordeler og ulemper som kan nevnes her.)

7 Dialogsystemer (14 poeng)

7.1 TF-vektorene (5 poeng)

TF-vektorene for de 5 ytringene er:

	Anna	du	er	hei	ja	jeg	sulten	!	?
Ytring 1				1				1	
Ytring 2	1	1	1	1				1	1
Ytring 3	1		1		1	1			
Ytring 4	1	1	1				1		1
Ytring 5					1			1	

Cellene som ikke har noen tall har en verdi $= 0$.

7.2 Besvarelse (9 poeng)

Vi starter med å regne ut TF-vektoren for spørsmålet er du sulten?"

	Anna	du	er	hei	ja	jeg	sulten	!	?
Spørsmål		1	1				1		1

Vi da beregne cosine similarity mellom denne vektoren og vektorene i korpuset:

- Ytring 1: 0
- Ytring 2: $\frac{3}{\sqrt{6}\sqrt{4}}$
- Ytring 3: $\frac{1}{\sqrt{4}\sqrt{4}}$
- Ytring 4: $\frac{4}{\sqrt{5}\sqrt{4}}$
- Ytring 5: 0

Vi kan dermed se at den nærmeste ytringen til spørsmålet er ytring 4. Det betyr at chatboten vil da velge som svar ytringen som kommer etter ytring 4, nemlig ytring 5: **"ja"**.