Luonnollisen kielen tilastollinen käsittely

T-61.281 (3 ov) L

Kevät 2004

Luennot: **Timo Honkela ja Krista Lagus**

Laskuharjoitukset: **Vesa Siivola**

1. Markov-mallit

1.1 Näkyvät Markov-mallit

Olkoon satunnaismuuttuja X jolla ajanhetkillä $1 \dots t$ arvot $X = (X_1, \dots, X_t)$. Satunnaismuuttujan arvo jokin tiloista $S = s_1, \dots, s_N$.

Satunnaismuuttujan arvot peräkkäisinä ajanhetkinä ovat toisistaan riippuvaisia. Yleisessä tapauksessa (ei-Markov) tn:t riippuvat koko (äärettömästä) historiasta, eli siitä miten tämänhetkiseen tilaan tultiin.

Markov-ominaisuudet:

1. **Äärellinen horisontti:**

$$P(X_{t+1} = s_k | X_1, \dots, X_t) = P(X_{t+1} = s_k | X_t) \tag{1}$$

2. **Aikariippumattomuus (stationaarisuus):**

$$P(X_2 = s_k | X_1) = P(X_3 = s_k | X_2) = P(X_{t+1} = s_k | X_t) \tag{2}$$

Markov-ketjun esitys taulukkona

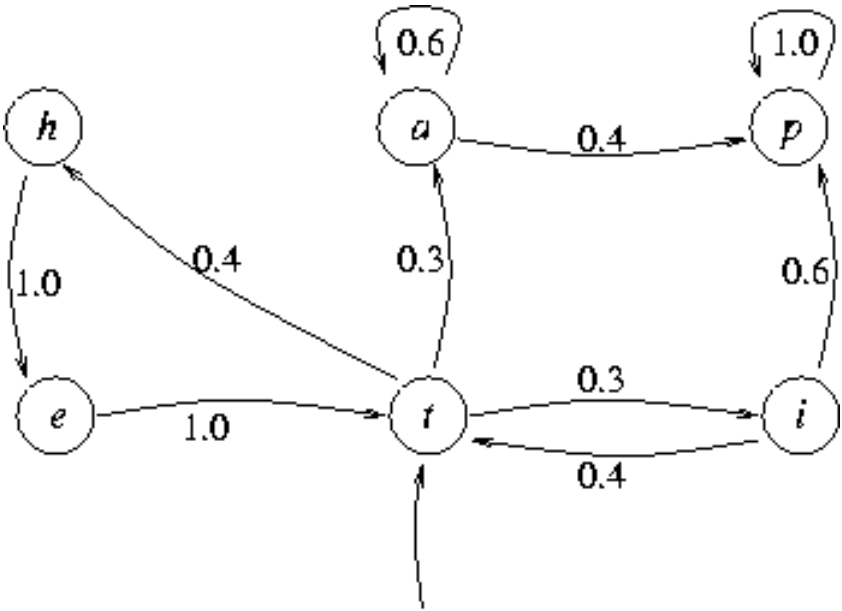
Tilojen joukko $S = \{a, e, i, h, t, p\}$

Tilasiirtymätn:t $a_{ij} = P(X_{t+1} = s_j | X_t = s_i)$ voidaan esittää 2-ulotteisena taulukkona:

$a_{i,j}$	a	e	i	h	t	p
a	0.6	0	0	0	0	0.4
e	...					...
i					0.6	0.4
h	...					...
t	0.3	0	0.3	0.4	0	0
p	...					...

Lisäksi määriteltävä tilojen aloitustodennäköisyydet: esim. ylimääräisen alkutilan s_0 avulla josta tilasiirtymätn muihin tiloihin taulukossa.

Esittäminen probabilistisena äärellisenä tila-automaattina



- Äärellinen tila-automaatti eli Finite-State-Automaton, FSA: tilat piirretään ympyröinä, tilasiirtymät kaarina.

- Mikäli automaattia käytetään lukemaan jotain syötejonoa (ja esim. tarkastamaan onko se sallittu), tai tulostamaan kielessä sallittuja symbolijonoja, voidaan kaariin piirtää ko. tilasiirtymässä luettava/tulostettava symboli.

- epädeterministinen automaatti: yhdestä tilasta johtaa ulos monta kaarta (joissa sama symboli, ts. ei ko. tilassa tiedetä mihin tulisi mennä)

- Painotettu tai probabilistinen automaatti: jokaisella kaarella tn, ja yksittäisestä tilasta ulosmenevien kaarien tn:ien summa $=1$.

- Näkyvä Markov-ketju vastaa äärellistä probabilistista automaattia jossa tilajono on tunnettu. Tilajonoa voidaan siis pitää automaatin tulosteena.

Tilasekvenssin todennäköisyys:

$$
\begin{aligned}
P(X_1 \dots X_T) &= P(X_1)P(X_2|X_1)P(X_3|X_1X_2)\dots P(X_T|X_1\dots X_{T-1}) && (3)\\
&= P(X_1)P(X_2|X_1)P(X_3|X_2)\dots P(X_T|X_{T-1}) && (4)\\
&= \pi_{X_1}\prod_{t=1}^{T-1} a_{X_tX_{t+1}} && (5)
\end{aligned}
$$

jossa $a_{X_tX_{t+1}}$ on siirtymätodennäköisyys tilasta hetkellä t tilaan hetkellä $t+1$ (katsotaan tn:t taulukosta tai tilakoneen kaarista) ja π_{X_1} on tilan X_1 aloitustodennäköisyys.

N-grammimalli Markov-mallina:

- Esim. tri-grammimallissa $P(w_3|w_1w_2)$ merkitään tilojen joukko S on kaikkien 2-grammien joukko $S = \{w_1w_1, w_1w_2, \dots w_1w_N, w_2w_1, \dots w_2w$
- Rajallisen pituiset historiat voidaan aina esittää äärellisenä joukkona tiloja, jolloin 1. Markov-oletus täyttyy.

- Markov-mallit erotellaan historian pituuden m mukaan, ja puhutaan m:nnen asteen Markov-mallista.
- $n$grammimalli on siis $n-1$:nnen asteen näkyvä Markov-malli.

1.2 Piilo-Markov-Mallit

Hidden Markov Models, HMMs, kätketty Markov-malli

Mallin rakennuspalat:

- Tilat: $S = \{s_1, \ldots s_N\}$
- Tilasiirtymätodennäköisyydet: $A = \{a_{ij}\}, 1 \leq i, j \leq N$
- Havainnot: $\{o_1 \ldots o_t\}$
- Havaintotodennäköisyydet kussakin tilasiirtymässä: $B = \{b_{ijk}\}$

Ero näkyvään Markov-malliin: Vaikka havaintojono tunnetaan, tilasekvenssi ei yksikäsitteisesti tunnettu (kuitenkin tunnetaan tilasekvenssien tn-jakauma).

Esimerkki:

Tilat: Hyvä tuuli / Ärsyyntynyt
Havainnot: 'Upeaa!', 'Lähdetäänkö leffaan?', 'Miksei kukaan ole tyhjentänyt roskista?'

HMM:n variantteja

- mukana nollatransitioita: jotkin tilasiirtymät eivät emittoi mitään (merkitään esim. epsilonilla ϵ)
- arc-emission-HMM: havainnot emittoidaan kaarista: b_{ijk}
- state-emission-HMM: havainnot emittoidaan tiloista: b_{ik}

1.3 HMM-mallin käyttäminen

Output-sekvenssin tuottaminen

Jos HMM-malli on annettu, havaintojonojen tuottaminen siitä on triviaalia, seuraavasti:

t:=1
Arvotaan alkutila X_1 tilojen aloitustn:ien perusteella: $P(s_i) = \pi_i$
while (1)
. . . . . Jos ollaan tilassa i, arvotaan tilasiirtymä $s_i \rightarrow s_j$ tn:llä a_{ij}
. . . . . Arvotaan siirtymässä ij tulostettava symboli o_t=k tn:llä b_{ijk}
end

Havaintojonon tn:n laskeminen

Jos on annettuna tietty havaintojono $O = o_1 \dots o_t$ ja malli $\mu = (A, B, \Pi)$, miten saadaan tehokkaasti laskettua havaintojonon O todennäköisyys mallissa, $P(O|\mu)$?

Suoraviivainen, tehoton ratkaisu: summataan havaintojonon tn kaikkien tilasekvenssien yli (edellyttää $(2T+1)N^T$ kertolaskua, jossa T havaintojonon pituus)

$$P(O|X,\mu) = \prod_{t=1}^{T} P(o_t|X-t, X-t+1, \mu) \quad (6)$$

$$P(O|\mu) = \sum_{X} P(O,X|\mu) = \sum_{X} P(O|X,\mu)P(X|\mu) \quad (7)$$

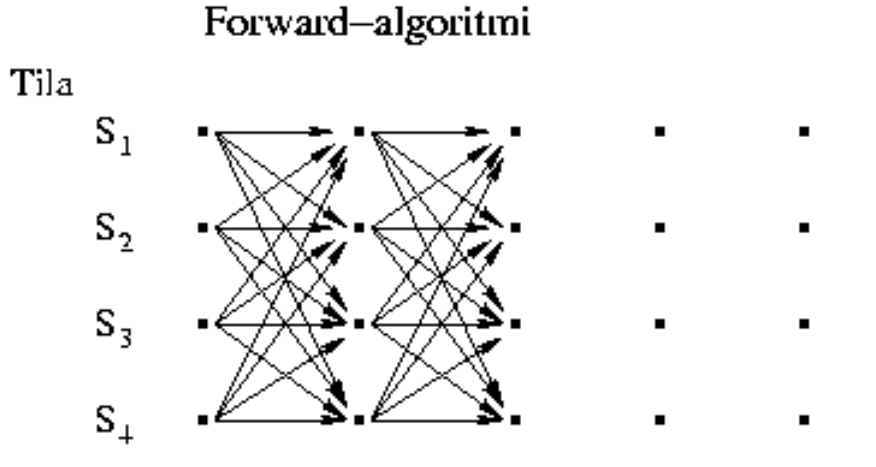
$$= \sum_{X_1 \dots X_{T+1}} \pi_{X_1} \prod_{t=1}^{T} a_{X_t X_t+1} \quad (8)$$

Huomattavasti tehokkaampi ratkaisu saadaan *dynaamisella ohjelmoinnilla* kun huomataan että sekvensseillä on yhteisiä alisekvenssejä, ja lasketaan tn:t kunkin alisekvenssin osalta vain kerran, eli

Forward-algoritmi: (eteenpäinlaskenta)

Kerätään tähänastisia tn:iä tilakohtaisiin apuparametreihin α_i, kulkien hilaa ajassa eteenpäin:
$\alpha_i(t) = P(o_1 o_2 \dots o_t - 1, X_t = i|\mu)$

- Initialisoi tilojen alkutodennäköisyyksillä: $\alpha_i(1) = \pi_i$
- Induktioaskel: summaa i:n tulokaarista saapuva tn-masssa: $\alpha_i(t+1) = \sum_{i=1}^{N} \alpha_i(t) a_{ij} b_{ijo_t}$
- Lopuksi: $P(O|\mu) = \sum_{i=1}^{N} \alpha_i(T+1)$



Huom: **Backward-algoritmi** (taaksepäinlaskenta) samoin mutta ajassa päinva

suuntaan.

Eteenpäin- ja taaksepäin-laskenta voidaan myös yhdistää missä kohdassa aikajanaa tahansa:

$$P(O|\mu) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t), \text{ jossa } 1 \leq t \leq T+1 \tag{9}$$

Dekoodaus eli todennäköisimmän tilajonon etsiminen

Tehtävä: etsi tilajono joka parhaiten selittää havaintojonon O.

Eräs mahdollinen tulkinta: maksimoidaan oikeinarvattujen tilojen *lukumäärä*. Kuitenkin tämä saattaa johtaa hyvin epätodennäköisiin tilasekvensseihin.

Siksi toinen tulkinta: etsitään *todennäköisin tilasekvenssi* joka on tuottanut havaintojonon.

Viterbi-algoritmi

(Tunnetaan myös nimillä DP alignment, Dynamic Time Warping, one-pass decoding)

- etsii havainnoille todennäköisimmän kokonaisen tilasekvenssin tehokkaasti
- Tallettaa jokaiseen hilapisteeseen siihenastisen todennäköisimmän po-

lun:

$$\delta_j(t) = \max_{X_1 \ldots Xt-1} P(X_i \ldots X_{t-1}, o_1 \ldots o_{t-1}, X_t = j|\mu) \qquad (10)$$

- Initialisoi tilojen alkutodennäköisyyksillä: $\delta_j(1) = \pi_j$
- Induktioaskel: valitse edellinen tila josta tulee suurin sekvenssin tn:
  talleta tn: $\delta_j(t+1) = \max_{1 \leq i \leq N} \delta_i(t) a_{ij} b_{ijo_t}$
  talleta pointteri ko. tilaan: $\phi_j(t+1) = \arg\max_{1 \leq i \leq N} \delta_i(t) a_{ij} b_{ijo_t}$
- Lopuksi: Lue hilaa lopusta taaksepäin seuraten ϕ_j:n talletamia kaaria ja kerää todennäköisin tilajono.
- Tasatilanteet voidaan ratkaista arpomalla.
- Keskeinen ero forward-algoritmiin: *tddennäköisyyksien summan* sijaan talletetaan *maksimi-tn*.
- Joskus yhden parhaan sekvenssin sijaan halutaan n parasta sekvenssiä (n-best-list). Tällöin talletetaan jokaisessa hilapisteessä $m < n$ parasta tn:ää ja tilaa.

- Viterbi tietyssä mielessä approksimaatio forward-algoritmille, joka ei laske *havainnon* tn:ää, ainoastaan selvittämään todennäköisimmän *tilasekvenssin* ja sen tn:n.

A*-dekoodausalgoritmi

Dekoodausalgoritmi joka soveltaa täydellistä forward-algoritmia (summaus), ja käy hilaa läpi pinon avulla prioriteettijärjestyksessä.

Kaksivaiheinen dekoodausalgoritmi (viterbi oli one-pass), jossa ensin nopeasti valitaan potentiaaliset seuraavat sanat, sitten lasketaan tarkemmin näiden tn:t hilassa

Lisäksi on muita useampivaiheisia dekoodausalgoritmeja.

1.4 Parametrien estimointi

Jos annettuna tietty havaintojono O (opetusdata), etsitään HMM-mallin parametreille $\mu = A, B, \pi$ arvot jotka uskottavimmin selittävät havainnot. Sovelletaan MLE-estimointia:

$$\arg\max_{\mu} P(O_{\text{opetusdata}}|\mu) \tag{11}$$

Ei analyytistä ratkaisua.

Baum-Welch eli forward-backward-algoritmi

- Iteratiivinen optimointimenetelmä, EM-algoritmin sovellus. Periaate: vuorotellaan seuraavia kahta askelta:
  1. Pidä mallin parametrit vakiona, laske $P(O|\mu)$ ja kerää tiedot siitä miten O:n tn-massa jakautui mallin hilaan (eli miten monta kertaa kuljettiin mitäkin reittiä)
  2. Uudelleenestimoi mallin parametrit $\mu \to \hat{\mu}$ siten että O olisi mahdollisimman todennäköinen

- Taatusti joka kierroksella $P(O|\hat{\mu}) \geq P(O|\mu)$ (kuten EM yleensäkin)
- Malli voidaan initialisoida satunnaisesti tai jollain nopealla raa-alla menetelmällä
- Iteroidaan kunnes datan tn ei enää muutu merkittävästi.
- Löytää lokaalin maksimin, ei välttämättä globaalia (kustannusfunktio on suuren joukon parametrejä luultavasti epälineaarinen funktio).

Parametrien estimaatit sanallisesti:

$$\hat{\pi}_i = \text{ odotettu frekvenssi tilassa } i \text{ hetkellä } t = 1$$

$$\hat{a}_{ij} = \frac{\text{odotettu transitioiden lkm tilasta } i \text{ tilaan } j}{\text{odotettu transitioiden lkm tilasta } i}$$

$$\hat{b}_{ijk} = \frac{\text{odotettu transitioiden lkm tilasta } i \text{ tilaan } j \text{ kun havainto oli } k}{\text{odotettu transitioiden lkm tilasta } i \text{ tilaan } j}$$

Käytännön implementointi sekä joitain ongelmia

Ongelma: Kerrotaan paljon hyvin pieniä tn:iä keskenään → ylivuoto-ongelmia.

Viterbissä vain kertolaskua ja max-operaatiota → voidaan toteuttaa kokonaan logaritmoituna, jolloin kertolaskut summia.

Baum-Welchissä: voidaan käyttää skaalauskertomia joita kasvatetaan ajan t funktiona.

Ongelma: Parametrien suuri määrä

- tarvitaan paljon dataa TAI
- oletetaan että osa parametreistä jaetaan verkon eri osien kesken ('parameter tying': sidottuja tiloja tai sidottuja tilasiirtymiä)
- oletetaan että verkossa on 'rakenteisia nollia', ts. rakenteellisesti mahdottomia tilasiirtymiä

Estimointialgoritmi löytää lokaalin maksimin, ei globaalia

- Parametrien fiksu initialisointi
- Erityisen tärkeää initialisoida havaintotn:t $B = b_{ijk}$

1.5 Soveltamistavoista

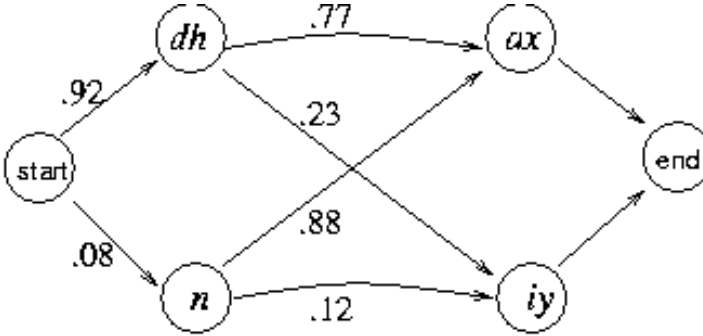
Soveltaminen puheentunnistukseen

Yksinkertaistettu esimerkki: **englannin sanojen ääntämisen mallinnus** Jurafsky & Martin (2000) kuvat 7.5–7.8.

Havainnot fooneja. Sanaa vastaa joukko tiloja joita voidaan käyttää ko. sanan ääntämisessä.

Lasketaan kullekin sanalle oma HMM joka mallintaa sanan eri lausumistavat ja näiden tn:t.

Havaintojono: foonijono, esim. [dh iy]
Sana: 'the', jonka yksinkertaistettu tilamalli (kuva)



Hyvin yksinkertaistettu ääntämismalli sanalle 'the'

Jos tehtävänä on tunnistaa mikä yksittäinen sana lausuttiin, yhdistetään sanamallit *rinnakkain* yhdeksi isoksi HMM:ksi

Lisätietoa: äärellisten automaattien yhdistäminen, ks. esim. Jurafsky&Martin.

Yhdistetty segmentointi + tunnistus

Viterbi-algoritmilla voidaan suorittaa samalla kertaa sekä segmentointi sanoiksi että sanojen tunnistus.

Havaintojono: foonijono, esim. [aa n iy dh ax]
Tuloste: sanajono 'I need the'

- Jokaiselle sanalle oma HMM
- Poimitaan sanaparien esiintymistn:t bi-grammimallista, ja käytetään näitä sanojenvälisiä tn:iä yhdistettäessä sanakohtaiset HMM:t *peräkkäin* (Jurafsky-Martin, kuva 7.8).
- Huom: näin tehdessä oletetaan että edellisen sanan ja seuraavan sanan lausuntatavat eivät riipu toisistaan. Oikeasti sanojen välillä riippuvuuksia lausumistavoissa, esim. savolainen viäntää, ja helsinkiläinen sihauttaa s-kirjainta sanasta riippumatta.
- Riippumattomuusoletusten tarkoitus helpottaa estimointia (vähemmän parametreja). Voidaan myös soveltaa pelkästään mallin initialisointiin.

Viterbin ongelma:

- Etsii todennäköisimmän tilajonon, mikä ei ole sama asia kuin todennäköis sanajono.
- Kukin tilajono vastaa jotain sanojen lausuntavaihtoehtoa. Löytää siis todennäköisimmän *lausuntavaihtoehdon*.
- Kuitenkin sanajonon tn on summa sen sanajonon eri lausuntavaihtoehtojen tn:istä.
- Ei voida soveltaa kaikkien kielimallien kanssa (lähinnä bigrammin kanssa, jo trigrammien kanssa ongelmia)
- Eräs vaihtoehto: multi-pass-decoding: Viterbi nopeaan dekoodaukseen joka tuottaa n-best-listan, sitten sovelletaan parempaa kielimallia näistä valitsemiseen.

Aidompi HMM:ien sovellus puheentunnistukseen:

- Havainnot puheen spektristä laskettuja piirrevektoreita.
- Tilat vastaavat fooneja
- Aika ei kulje fooni kerrallaan vaan tietynkestoisina segmentteinä.
- Erikestoiset foonit hoidetaan esim. siten että foonista on kaari myös siihen itseensä.

Muita HMM:ien sovelluksia

- sanaluokkien taggaus
- geenisekvenssien analyysi / taggaus