Luonnollisten kielten tilastollinen käsittely

T-61.281 (3 ov) L

Kevät 2002

Luennot: **Krista Lagus**
Laskuharjoitukset: **Vesa Siivola**

1. YLEISTÄ KURSSISTA

1.1 Kurssin suorittaminen

Kurssi suoritetaan tekemällä harjoitustyö ja läpäisemällä tentti.

1.2 Ilmoittautuminen

Ilmoittautukaa kurssille www-topin avulla. Kieliteknologian opetuksen verkoston (KIT) opiskelijat voivat ilmoittautua sähköpostitse kurssin luennoijalle, kunnes saavat opintokirjannumeron TKK:lle.

1.3 Tiedotukset

Kurssista tiedotetaan webissä http://www.cis.hut.fi/Opinnot/T-61.281, ryhmässä news://nntp.tky.hut.fi/opinnot.tik.informaatiotekniikka sekä CIS-laboratorion ilmoitustaululla 3.krs aulassa B-käytävän suulla.

1.4 Luennot

Luennot pidetään keskiviikkoisin kello 10–12 salissa T2.

Luennoitsija opettava tutkija TKT Krista Lagus (mailto:krista.lagus@hut.fi).

Luentokalvot ovat luennon jälkeen nähtävillä osoitteessa http://www.cis.hut.fi/Opinnot/T-61.281/kalvot.pdf.

Luennoitsijan vastaanotto on tarvittaessa välittömästi luennon jälkeen T-talossa huoneessa C310.

1.5 Laskuharjoitukset

Laskuharjoitukset tiistaisin kello 8–10 salissa U264 alkaen 22.1.2001.

Harjoitukset pitää DI Vesa Siivola (mailto:vesa.siivola@hut.fi).

Tehtävät laitetaan edellisenä perjantaina nähtäville osoitteeseen http://www.cis.hut.fi/Opinnot/T-61.281/laskarit.html.

1.6 Kirja

Kurssi seuraa kirjaa:

Christopher D. Manning, Hinrich Schütze:
Foundations of statistical natural language processing,
MIT Press, 1999.

Kirja löytyy TKK:n pääkirjastosta ja tietotekniikan kirjastosta.

Yliopiston kirjakauppa (TKK) tilaa kirjaa haluttaessa sitovilla etukäteistilauksil
(hinta n. 600 mk).

Kirjan hinta muutamassa web-kaupassa:
http://www.uk.bol.com/ noin 74 euroa (sis. kuljetus)
http://www.amazon.com/ noin 87 euroa (sis. kuljetus)
http://www.amazon.co.uk/ noin 79 euroa (sis. kuljetus)

Tutustumiskappale on nähtävillä laboratorion sihteerin Tarja Pihamaan huo-
neessa B326 olevassa harmaassa peltisessä vetolaatikostossa.

1.7 Luentomonisteet

Laskuharjoitukset ratkaisuineen ja luentokalvot ilmestyvät opetusmonisteina.

Materiaalia voi myös lainata luennoitsijalta ja assarilta itse kopioitavaksi.

Vapaaehtoinen kurssitoimittaja?

1.8 Suhde muihin opintoihin

Kurssi soveltuu osaksi seuraavia opintoja

- Kieliteknologian pää- ja sivuaine TKK:lla (Tik, Sähkö)
- Informaatiotekniikan pää- ja sivuaineen valinnaiset opinnot
- KIT-verkoston opinnot (mm. Helsingin Yliopistolla)
- Muut aiheeseen liittyvät jatko-opinnot TKK:lla ja muualla (hyväksytettä
  erikseen)

1.9 Tentti

Tentti järjestetään 14.toukokuuta klo 13-16. Lisäksi syksyn tenttikausilla järjestetään yksi tai kaksi tenttiä.

Tentissä on 5 tehtävää à 6 pistettä, maksimi 30 pistettä.

Tentissä pyritään mittaamaan sitä kuinka hyvin opiskelija on perehtynyt toisaalta tilastollisen kielenkäsittelyn sovellusongelmiin ja toisaalta alan keskeisiin menetelmiin.

Tehtävät tulevat painottumaan luentomonisteiden ja laskarien käsittelemiin aiheisiin. Kuitenkin myös kirjan lukeminen näiden aiheiden osalta on suositeltavaa.

Tehtävät voivat olla esseetehtäviä, pieniä sanallisia tehtäviä kuten termien tai kaavojen selittämistä, ja laskutehtäviä. Laskutehtävät ovat samantyyppisiä kuin laskareissa.

Tehtävinä voi olla esim. tietyn sovellusongelman selostaminen (mistä on kysymys), mitä menetelmiä ongelmaan on käytetty tai voidaan käyttää, jonkin (tietyn) menetelmän selostaminen yksityiskohtaisesti, tai eri menetelmien hyvien ja huonojen puolien vertaaminen.

Voidaan myös edellyttää kykyä tulkita mitä oletuksia jossain mallissa (annettu esim. kaavana) tehdään, ja arvioida kuinka paikkansapitäviä ne ovat ko. sovellusongelman kannalta.

Tentissä saa olla mukana matemaattinen kaavakokoelma ja tavallinen funktiolaskin.

Tenttiin ilmoittaudutaan normaalisti eli Topin kautta viimeistään 2 päivää etukäteen.

1.10 Harjoitustyö

Kurssin suoritukseen kuuluu pakollinen harjoitustyö.

Harjoitustyön tehtävänanto, arvostelu ja aiheet esitellään luennolla myöhemmin keväällä, jolloin aiheet laitetaan myös esille osoitteeseen http://www.cis.hut.fi/Opinnot/T-61.281/harjtyo.html.

2. JOHDANTO

2.1 Mitä on tilastollinen luonnollisen kielen käsittely?

- Kieliteknologian osa-alue
- Sovelletaan informaatiotekniikan, tilastomatematiikan, ja tietojenkäsittelytieteen menetelmiä kieliteknologisiin ongelmiin.
- Malliperheet sisältävät todennäköisyyksiä, jotka estimoidaan hyvin suurista aineistoista (*korpuksista*).
- Sovelluskohteita: tiedonhaku, tekstien järjestäminen ja luokittelu, puheentunnistus, luonnollisen kielen käyttöliittymät tietokantoihin ja varauspalveluihin
- Menetelmäaloja: koneoppiminen, hahmontunnistus, tilastotiede, todennäköisyyslasku, signaalinkäsittely
- Lähialoja: lingvistiikka, korpuslingvistiikka, fonetiikka, keskusteluntutkimus, tekoälyntutkimus, kognitiotiede

- Nykykäsitys: Lingvististä tietoa kielestä hyödynnetään prioritietona, sen sijaan että lähdettäisiin täysin puhtaalta pöydältä.

2.2 Mallinnuksen peruskäsitteitä

- Malli — Jonkin ilmiön tai datajoukon kattava kuvaus.
  Esim: sääntökokoelma joka kuvaa suomen morfologian.
- Malliperhe, malliavaruus — joukko potentiaalisia malleja joita harkitaan ilmiön kuvaamiseen. Esim. niiden sääntöjen kokoelma jota voitaisiin periaatteessa käyttää kielen syntaksin kuvaamiseen.
- Mallin valinta — prosessi jonka kautta päädytään johonkin tiettyyn malliin. Algoritmit usein tämäntyyppisiä: vuorotellaan mallin evaluointia ja mallin muuttamista, pyrkien kohti parempaa mallia.
- Oppiminen — ks. mallin valinta.
- Probabilistinen malli(perhe) — representoi ilmiöiden todennäköisyyksiä.
- Algoritmi — menetelmä jolla edetään malliavaruudessa, kokeillen eri malleja.
- Iteratiivinen — vähän kerrassaan, toiston kautta tapahtuva

2.3 Yleisestä kielitieteestä

Tavoiteena kuvata ja selittää toisaalta kielen (kielten) säännönmukaisuudet, toisaalta kielen (kielten) monimuotoisuus.

Oppivien menetelmien terminologialla ilmaistuna: tavoiteena on *konstruoida malli kielestä*.

Kielen ilmenemismuotoja mm. keskustelut visuaalisella kontaktilla ja ilman, viitomalla, yksinpuhelut, kirjoitetut artikkelit, kirjat, luennot, ja muut kielelliset viestit eri viestinvälineitä ja -ympäristöjä käyttäen.

Laajemmin nähtynä kielen mallinnuksen tavoitteena on selvittää ja kuvata:

- Miten ihmiset käyttävät kieltä, mitä todella sanotaan?
- Mitä kielenkäyttäjä tahtoo tai mihin pyrkii sanoessaan jotain?

Kuvattavan ilmiön osa-alueita:

- Kogitiiviset mekanismit, eli miten kielikyky syntyy ja muotoutuu ihmisessä (ja muissa olennoissa), ja miten tuotamme ja ymmärrämme kieltä.
- Kielen ilmausten ja maailman väliset yhteydet (entä *maailmantiedon* kuvaus?)
- Kielessä esiintyvät säännönmukaisuudet.

2.4 Perinteinen lähestymistapa kielitieteessä

Ominaisuus 1: Perinteisen lähestymistavan mukaan kieli on kuvattavissa *joukkona 'kovia' sääntöjä*, esim. produktiosääntöjä.
<= MALLIPERHE

Esimerkki: Englannin substantiivilauseke NP koostuu valinnaisesta artikkelista DET=[a, the, an], valinnaisesta määrästä adjektiiveja ADJ=[brown, beautiful,...] ja substantiivista N=[flower, building, thought...].

NP => (Det)? (ADJ)* N

Ominaisuus 2: Sääntöjen avulla pyritään kuvaamaan mitkä lauseet ovat hyvinmuodostettuja (sallittuja, kieliopin mukaisia) ja mitkä väärinmuodostettuja (kiellettyjä, kieliopin vastaisia).
<= MALLINNUKSEN TAVOITE

Mallinnuksen tavoite aina kahtalainen: *kattavuus* ja *tarkkuus*.

'Kaikki kieliopit vuotavat' (Edward Sapir, 1921)

Täydellisen kuvauksen saavuttamisen esteinä ainakin kielellinen variaatio (yksilöiden ja kieliyhteisöjen välillä), luovuus, kielen muuttuminen.

Kritiikki 1: Onko kovan kieliopillinen-eikieliopillinen rajan etsiminen hyvin määritelty ongelma, ts., onko sellaista rajaa edes olemassa, vai onko kyse aidosti sumeasta ilmiöstä?

On paljon lauseita joiden kieliopillisuudesta voidaan olla montaa mieltä, ja ollaankin. $=>$ Todellisuudessa kovaa rajaa ei ehkä ole.

Kritiikki 2: Onko kieliopillisuus relevantti ja riittävä kielen kuvauksen taso?

Esim. lause 'Colourless green ideas sleep furiously.' (Chomsky) on syntaktisesti ok, mutta semanttisesti ei kovin mielekäs tai ainakaan tavanomainen.

Ratkaisuyritys: määritellään myös semanttisia sääntöjä. Ongelmia kuitenkin tulee, mm. sanojen metaforisen käytön kanssa. Ehkä 'kovat' säännöt ylipäänsä eivät ole oikea malliperhe?

Esimerkki:

Sääntö: niellä-sanan subjektina täytyy olla elävä olento
Lause: Supernova nielaisi planeetan.

2.5 Kategoriset (diskreetit) vs. jatkuvat representaatiot

- a/ä p/b: äänisignaalissa jatkuva muutos, foneemitasolla havainto on kategorinen: havaitaan joko a tai ä, ja havainto muuttuu yhtäkkisesti jossain kohti signaalin muuttuessa vähitellen.

Havainttaessa puhetta muutos jatkuvalta representaation tasolta (äänisignaali) diskreetiksi tai kategoriseksi (foneemi). Puhetta tuotettaessa päinvastainen muutos.

Todellisissa systeemeissä eroa diskreetin ja jatkuvan välillä ei ole, koska:

- Kaikki todelliset systeemit ovat kohinaisia (fysiikan perusteet)
- kohinainen kommunikaatikanava aina diskretoi signaalin (Shannonin informaatiteoria)

Sen sijaan aidosti relevantti kysymys on, onko representaatioavaruuden pisteiden välille määritelty etäisyysrelaatio (metriikka) vai ei. Usein tarkoitetaankin tätä silloin kun puhutaan jatkuvista representaatioista.

2.6 Probabilistinen esitystapa

- Probabilistisessa mallissa malliperheenä todennäköisyydet. (vertailukohta: kaksiarvoinen esitystapa jossa asiat ovat joko-tai, tosia tai epätosia
- Esitystapa mahdollistaa tiedon esittämisen silloinkin kun ei voida muodostaa kategorista sääntöä, mutta on olemassa preferenssi: Subjekti on ennen predikaattia 90% tapauksista P(A)=0.9.
- 'Kova' sääntö: P(A)=1 tai P(A)=0.
- Probabilistisessa representaatiossa tiedon kerääminen ja mallin päivittäm voi tapahtua iteratiivisesti, vähitellen. Lisäesimerkit tarkentavat aiemmin muodostettua alustavaa kuvaa.

2.7 Datasta oppiminen

Periaatteellinen lähtökohta mallinnukseen

Mallit eivät ole tosia tai epätosia. Ne ovat parempia tai huonompia. Paremmuus mitataan suurista joukoista todellista dataa kokonaisuutena. Vertailukohta tai onnistumisen mitta ei ole vastaavuus lingvistisen intuition kanssa, vaan ilmiön/datan optimaalinen kuvaus.

Perustelu oppimiselle

Miksi kannattaa muodostaa malleja automaattisesti, datasta oppimalla tai estimoimalla (eli automaattisesti), eikä asiantuntijatietoa kirjaamalla?

- Data on halpaa ja sitä on paljon, myös sähköisesti.
- Voidaan saada mallit aikaan nopeammin / vähemmällä ihmistyövoimalla / pienemmin kustannuksin.
- Kielen muuttuessa mallit voidaan estimoida uudestaan helposti.

- Asiantuntijatietämys hankalaa tuottaa tai kerätä (mm. konsistenssiongelmat).
- Asiantuntijatietoa käytettäessä malliperhettä rajoittaa 'ihmisbias'.
- Koneiden 'kognitiiviset ominaisuudet' eroavat ihmisen vastaavista.
- Toteutettaessa kielikykyä koneille ei tarvitse rajoittua ihmiselle helposti ymmärrettäviin malleihin.
- Aineistolähtöinen keskittää resurssit niihin ilmiöihin jotka todella esiintyvät. Resurssien käyttö suhteessa ilmiön keskeisyyteen aineistossa.

Onnistuneen oppivan mallinnuksen seurauksia

- Resurssien käytön tehostuminen: Voidaan ulottaa mallinnus laajempaan kielijoukkoon, ja yksittäisen kielen sisällä eri osa-alueisiin.
- Laadullinen parannus, koska koneellisesti pystytään käymään läpi suuri joukko malleja ja koska mallin valinnassa ei ole inhimillistä biasta (ainakaan samassa määrin kuin käsin muodostetuissa malleissa).

Riskejä ja haasteita

- Datan valinta ja kattavuus,
- sopivien malliperheiden määrittely,
- optimointimenetelmien tehokkuus.

2.8 Ihmisen kielikyky ja kielen oppiminen

Miten kielikyky ihmisellä syntyy ja muotoutuu? Mikä osa on synnynnäistä, mitä opitaan?

Rationalistinen näkemys: Kielikyky on synnynnäinen, ja oma erillinen kielimodulinsa

Keskeisiltä osin ihmismielen ja kielen rakenne on kiinnitetty (oletettavasti geneettisesti määrätty). Perustelu: argumentti stimuluksen vähyydestä (mm. Chomsky 1986). Kannattajia mm: Chomsky, Pinker.

Vrt. tekoälyntutkimus 1970-luvulla: tietämyksen koodaaminen käsin. Saatiin aikaan pienimuotoisia älykkään oloisesti käyttäytyviä systeemejä (mm. Newell & Simon: Blocks world). Systeemit usein käsin koodattuja sääntöpohjaisia järjestelmiä. Näiden laajentaminen on kuitenkin osoittautunut hyvin hankalaksi.

Empiristinen näkemys: Kieli opitaan, kielikyky toteutuu osana yleistä kognitiivista laitteistoa

Amerikkalaiset strukturalistit. Zellig Harris (1951) jne: tavoitteena kielen rakenteen löytäminen automaattisesti analysoimalla suuria kieliaineistoja. Ajatus siitä että hyvä rakennekuvaus (grammatical structure) on sellainen joka kuvaa kielen kompaktisti.

Nykyisin melko yleisen näkemyksen mukaan mieli ei ole täysin tyhjä taulu, vaan oletetaan että tietyt 1. rakenteelliset preferenssit yhdessä 2. yleisten kognitiivisten oppimisperiaatteiden ja 3. sopivanlaisen stimulin kanssa johtavat kielen oppimiseen.

Vrt. adaptiivisten menetelmien tutkimus, havaintopsykologia ja laskennallinen neurotiede, ihmisen havaintomekanismien ja piirreirroittimien muotoutuminen aistisyötteen avulla (*plastisiteetti*).

Avoimia kysymyksiä:

- Tarvittavan prioritiedon määrä ja muoto?
- Mitä ovat tarvittavat oppimisperiaatteet?
- Minkälaista syötettä ja missä järjestyksessä tarvitaan?

Käytännöllinen lähestymistapa

Tavoite voi olla puhtaasti käytännöllinen: kehittää toimivia, tehokkaita kieliteknologisia menetelmiä ja järjestelmiä.

Eri menetelmiä sovellettaessa ei välttämättä oteta rationalismi-empirismi-vastakkainasetteluun lainkaan kantaa.

Aineistoihin (korpuksiin) pohjautuvat ja tietämysintensiiviset mallit ovat tällöin samalla viivalla.

Vertailukriteerit:

- lopputuloksen laatu
- lopullisen mallin tilankäytön tehokkuus ja riittävä nopeus (esim. reaaliaikaiset sovellukset)
- mallin konstruoinnin tai oppimisen tehokkuus (tarvittava ihmistyö, prosessointitila ja -aika)

Usein kohteena jokin spesifi kieliteknologinen sovellusongelma, jonka ratkaisemiseksi riittää vain osittainen kielen mallinnus.

Koko kielikyvyn implementointi luultavasti edellyttäisi koko kognitiivisen väline ja tekoälyn toteuttamista, mukaanlukien maailmantiedon kerääminen ja esittär

3. MATEMAATTISIA PERUSTEITA

3.1 Todennäköisyyslasku

Peruskäsitteitä

Todennäköisyysavaruus (*probability space*):
Tapahtuma-avaruus Ω — diskreetti tai jatkuva
Todennäköisyysfunktio P
Kaikilla tapahtuma-avaruuden pisteillä A on todennäköisyys: $0 \leq P(A) \leq 1$

Todennäköisyysmassa koko avaruudessa on $\sum_A P(A) = 1$

Esimerkki 1

Jos tasapainoista kolikkoa heitetään 3 kertaa, mikä on todennäköisyys että saadaan 2 kruunaa?

Mahdolliset heittosarjat Ω: { HHH, HHT, HTH, HTT, THH, THT, TTH, TTT }

Heittosarjat joissa 2 kruunaa: A={ HHT, HTH, THH}

Oletetaan tasajakauma: jokainen heittosarja yhtä todennäköinen, $P = 1/8$

$$P(A) = \frac{|A|}{|\Omega|} = \frac{3}{8}$$

3.2 Ehdollinen todennäköisyys

A= asiaintila jonka todennäköisyyden haluamme selvittää
B= meillä oleva ennakkotieto tilanteesta, ts. tähän asti tapahtunutta

Ehdollinen todennäköisyys, A:n todennäköisyys ehdolla B:

$$P(A|B) = \frac{P(A,B)}{P(B)} \tag{1}$$

Palataan esimerkkiin 1: Oletetaan että on jo heitetty kolikkoa kerran ja saatu kruuna. Mikä nyt on todennäköisyys että saadaan 2 kruunaa kolmen heiton sarjassa?

Alunperin mahdolliset heittosarjat: {HHH, HHT, HTH, HTT, THH, THT, TTH, TTT}

Prioritiedon B perusteella enää seuraavat sarjat mahdollisia: { HHH, HHT, HTH, HTT }

$P(A|B) = 1/2$

Ketjusääntö

$$P(A_1, ...A_n) = P(A_1)P(A_2|A_1)P(A_3|A_1, A_2)\dots P(A_n|A_1, \dots, A_{n-1})$$

Riippumattomuus

Tilastollinen riippumattomuus:

$$P(A, B) = P(A)P(B) \tag{2}$$

Sama ilmaistuna toisin: se että saamme lisätiedon B ei vaikuta käsitykseemme A:n todennäköisyydestä, eli:

$P(A) = P(A|B)$

Huom: tilastollinen riippuvuus $\neq$ kausaalinen riippuvuus!

Esim. jäätelön syönnin ja hukkumiskuolemien välillä on tilastollinen riippu-

vuus. (Yhteinen kausaalinen tekijä ehkä lämmin kesäsää.)

Ehdollinen riippumattomuus

$$P(A, B|C) = P(A|C)P(B|C) \tag{3}$$

A ja B ovat riippumattomia ehdolla C mikäli on niin että jos jo tiedämme C:n, tieto A:sta ei anna mitään lisätietoa B:stä (ja päinvastoin).

3.3 Bayesin teoreema

Koska kyse ei ole kausaalisesta riippumisesta, tapahtumien järjestystä voidaan vaihtaa:

$$P(A, B) = P(B)P(A|B) = P(A)P(B|A) \tag{4}$$

Eli $P(B|A)$ voidaan laskea $P(A|B)$:n avulla.

$$P(B|A) = \frac{P(A, B)}{P(A)} = \frac{P(B)P(A|B)}{P(A)} \tag{5}$$

Jos $A =$ lähtötilanne, joka ei muutu (esim. jo tapahtuneet asiat), ja haluamme ainoastaan tietää, mikä tulevista tapahtumista B on todennäköisin, $P(A)$ on normalisointitekijä joka voidaan jättää huomiotta:

$$\arg\max_B P(B|A) = \arg\max_B \frac{P(B)P(A|B)}{P(A)} = \arg\max_B P(B)P(A|B) \quad (6)$$

Toisaalta, $P(A)$ voidaan myös laskea:

$P(A) = \sum_i P(A|B_i)P(B_i)$

3.4 Satunnaismuuttuja

Periaate: satunnaismuuttuja on se asia josta ollaan kiinnostuneita, ja joka kussakin kokeessa saa jonkin arvon.

- Jatkuva-arvoinen satunnaismuuttuja: $X : \Omega => \mathbb{R}^n$, jossa $\mathbb{R}$ on reaalilukujen joukko ja n on avaruuden dimensio. Jos $n > 1$ puhutaan myös satunnaisvektorista.

- Diskreetti satunnaismuutuja: $X : \Omega => S$, jossa S on numeroituva $\mathbb{R}$:n osajoukko.

- Indikaattorimuuttuja: $X : \Omega => 0, 1 (Bernoulli - jakautunut)$.

Todennäköisyysjakauma *probability mass function pmf* $p(x)$ kertoo miten todennäköisyysmassa jakautuu satunnaismuuttujan eri arvojen kesken. Jakauman massa aina $= 1$ (muussa tapauksessa ei ole tn-jakauma).

3.5 Odotusarvo ja varianssi

Odotusarvolle $E(X) = \sum_x xp(x)$

(diskreetissä tapauksessa; jatkuvassa tapauksessa summan korvaa integraali)

Ts. odotusarvo on keskiarvo kussakin näytteessä (kokeessa) saadun satunnaismuuttujan arvon yli.

Varianssi kuvaa muuttujan arvon vaihtelua keskiarvon ympärillä:

$$\begin{aligned} Var(X) &= E((X - E(X))^2) \\ &= E(X^2) - E^2(X) \end{aligned}$$

3.6 Yhteisjakauma

Yhteistodennäköisyys

P(X,Y) = kahden tapahtuman tai väitteen yhteistodennäköisyys, ts. että molemmat toteutuvat (esim. X=jonain tiettynä ajanhetkenä kuultu sananmuoto on 'siitä' ja Y=samana ajanhetkenä kuullun sanan sanaluokka on 'NOM'.)

Luetaan: X ja Y

Yhteistodennäköisyysjakauma

p(x,y) = kahden satunnaismuuttujan yhteisjakauma. Kuvaa x:n ja y:n kunkin arvokombinaation todennäköisyydet.

Kaavoja kootusti:

$$p(x,y) = p(X=x, Y=y) \quad \text{Yhteisjakauma} \quad (7)$$

$$p_X(x) = \sum_y p(x,y) \quad \text{Marginaalijakauma} \quad (8)$$

$$p(x,y) = p_X(x)p_Y(y) \quad \text{Riippumattomuus} \quad (9)$$

$$p_{X|Y}(x|y) = \frac{p(x,y)}{p_Y(y)} \quad \text{Ehdollinen jakauma} \quad (10)$$

$$p(x,y,z,w) = p(x)p(y|x)p(z|x,y)p(w|x,y,z) \quad \text{Ketjusääntö} \quad (11)$$

3.7 P:n laskeminen

- Yleisesti P on tuntematon, ja estimoitava datasta, tyypillisesti erilaisten tapahtumien frekvenssejä laskemalla.
- Koko todennäköisyysjakauman estimoinnin sijaan on mahdollista käyttää *parametrisia malleja* todennäköisyysjakaumille: tällöin estimoidaan vain jakauman parametrit.
- Bayeslaisessa estimoinnissa datan lisäksi huomioidaan prioritieto.

3.8 Tavallisia (parametrisiä) jakaumia

Notaatio: Jakauma(satunnaismuuttuja; jakauman parametrit)

Diskreettejä jakaumia: Binomijakauma

Satunnaismuuttujalla 2 mahdollista arvoa, onnistuu/ei, tai tarkasteltava ominaisuus (esim. tietty sana) joko on tai ei ole läsnä jossain tietyssä näytteessä (esim. lauseessa).

$p =$ Onnistumistodennäköisyys yksittäisessä kokeessa
$r =$ onnistumisten lukumäärä kun kokeita yhteensä n

$$b(r; n, p) = \frac{n!}{(n-r)!r!} p^r (1-p)^{n-r}, \text{jossa } 0 \leq r \leq n \tag{12}$$

Odotusarvo: np, varianssi: $np(1-p)$

Oletus: riippumattomat kokeet. Kieleen sovellettaessa kuitenkin edellinen ja

seuraava lause yleensä riippuvat toisistaan (samoin sanat), joten kokeet eivät ole todella riippumattomia.

Muita diskreettejä jakaumia

Multinomijakauma: Binomijakauman yleistys kun lopputuloksia voi olla useampi kuin kaksi.

Poisson-jakauma: Kiinteän kokoinen tapahtumaikkuna, jossa jakauma kuvaa tietyn, tutkittavan, asian tapahtumislukumäärän todennäköisyydet. Satunnaismuuttuja x on siis tapahtumien lukumäärä tietyssä aikaikkunassa.

Jatkuvia jakaumia

Normaalijakauma (gaussinen jakauma)

Märitelty jos tunnetaan keskiarvoμ ja varianssi σ^2 :

$$N(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{(x-\mu)^2/2\sigma^2}$$

Yleisesti: todennäköisyysjakauma voi olla mikä tahansa funktio jonka integraali $= 1$ välillä $[0, 1]$

3.9 Bayesläisestä tilastotieteestä

Tähänasti tarkasteltu todennäköisyyttä *frekventistisestä* näkökulmasta.

Bayeslainen tulkinta: todennäköisyys kuvastaa *uskomuksen astetta*. Bayeslaisessa mallinnuksessa myös prioritieto eli uskomukset *ennen* datan näkemistä kyetään ilmaisemaan eksplisiittisesti.

Esimerkki 1: ainutkertaiset tapahtumat

Mikä on todennäköisyys sille että maailmankaikkeus loppuu huomenna?

Frekventisti: ei vastausta, koska koetta ei voi toistaa N kertaa.
Bayeslainen: subjektiivinen todennäköisyys (uskomus) on olemassa.

Esimerkki 2: taskussani olevien kolikoiden rahallinen arvo

Arvo on jokin täsmällinen luku, mutta tietoni siitä ovat vajavaiset. Uskomukseni: Arvo on varmasti positiivinen, ja lähes varmasti alle tuhat mk.

3.10 Bayeslainen päätösteoria

Optimaalinen tapa mallin (teorian) valintaan: valitaan malli (teoria), joka uskottavimmin selittää jonkin havaintojoukon.

Ts. maksimoidaan mallin todennäköisyys kun tunnetaan data ts. mallin *posterioritodennäköisyys*: $P(\text{Malli}|\text{data})$

Esimerkki 1: Mallin parametrien valinta

Kolikonheitto. Olkoon malli M_m joka sanoo $P(\text{kruuna}) = m, 0 \leq m \leq 1$. Olkoon s jokin heittojono jossa i kruunaa, j klaavaa.

$$P(s|M_m) = m^i(1-m)^j \tag{13}$$

Frekventistisestä näkökulmasta, valitaan malli joka maksimoi datan todennäköisyyden (*MLE, maximum likelihood estimaatti*):

$$\arg\max_m P(s|M_m) \tag{14}$$

Havainnot: 10 heittoa joista 8 kruunaa.
Frekventistinen lähestymistapa (MLE): $m = \frac{i}{i+j} = 0.8$

Bayesläinen lähestymistapa: kolikkoa tarkastelemalla näyttäisi siltä että kolikko on tasapainoinen, siis dataa katsomatta vaikuttaisi todennäköiseltä että $m = 1/2$ tai niillä main. Tämä uskomus voidaan liittää malliin *priorijakaumana mallien yli*.

Valitaan prioriuskomuksiamme kuvastava priorijakauma

Eräs sopiva priorijakauma olisi gaussinen jakauma jonka keskipiste (ja siis maksimi) on $1/2$:ssa. Valitaan kuitenkin prioriksi polynominen jakauma, jonka keskipiste (korkein kohta) $1/2$ ja pinta-ala 0 ja 1 välillä on 1:

$$p(M_m) = 6m(1-m)$$

Posterioritodennäköisyys Bayeslaisessa lähestymistavassa:

$$\begin{aligned} P(M_m|s) &= \frac{P(s|M_m)P(M_m)}{P(s)} \\ &= \frac{m^i(1-m)^j \times 6m(1-m)}{P(s)} \end{aligned}$$

jossa $P(s)$ on datan prioritodennäköisyys. Oletetaan ettei se riipu mallista M_m joten voidaan jättää huomiotta mallia valittaessa.

Maksimoidaan osoittaja etsimällä derivaatan nollakohta m:n suhteen kun $i = 8$ ja $j = 2$. Tämä on

$$\arg\max_m P(M_m|s) = \frac{3}{4} \tag{15}$$

Mallin estimointi on-line

Aloitetaan pelkällä priorimallilla, ja aina uuden havainnon tultua päivitetään malli posteriorimalliksi (ns. MAP-estimointi).

Taustaoletus: peräkkäiset havainnot ovat riippumattomia.

Esimerkki 2: Teorioiden tai malliperheiden vertailu

Havainnot: joukko aidan takaa kuultuja "kruuna" ja "klaava"- sanoja.

Malli/Teoria $M1(\theta)$: joku heittää yhtä kolikkoa, joka saattaa olla painotettu, ja mallin vapaa parametri θ on painotuksen voimakkuus.

Malli/teoria $M2$: joku heittää kahta tasapainoista kolikkoa, ja sanoo "kruuna" jos molemmat kolikot ovat kruunia, ja "klaava" muuten. Mallin $M2$ mukaan heittojonon, jossa on i kruunaa ja j klaavaa todennäköisyys on siis:

$$P(\text{data}|M2) = (\frac{3}{4})^i(\frac{1}{4})^j$$

Tehdään oletus: molemmat teoriat/mallit yhtä todennäköisiä *a priori* (ts. ennen kuin on saatu yhtään havaintoa): $P(M1) = P(M2) = 0.5$

Bayesin kaavasta:

$$P(M1|data) = \frac{P(data|M1)P(M1)}{P(data)} P(M2|data) = \frac{P(data|M2)P(M2)}{P(data)}$$

Halutaan selvittää kumpi malleista on uskottavampi. Lasketaan niiden uskottavuuksien välinen suhde:

$$\frac{P(M1|data)}{P(M2|data)} = \frac{P(data|M1)P(M1)}{P(data|M2)P(M2)}$$

Jos suhdeluku on > 1, valitaan malli $M1$, jos < 1, malli $M2$

(Vastaukset eri heittosarjoilla: taulukko 2.1 kirjan sivulla 58)

3.11 Shannonin informaatioteoria

- Claude Shannon, 1948
- Tavoitteena maksimoida informaation siirtonopeus kohinaisella kommunikaatikanavalla
- Teoreettinen maksimi datan pakkaamiselle $=$ Entropia H (tai Kolmogorov kompleksisuus K)
- Kanavan kapasiteetti C: jos kapasiteettia ei ylitetä, virheiden todennäköisyys saadaan niin alhaiseksi kuin halutaan.
- Nykyiset tiedonpakkausmenetelmät hyödyntävät näitä teoreettisia tuloksia.

Entropia

Olkoon $p(x)$ satunnaismuuttujan X jakauma yli diskreetin symbolijoukon (aakkoston) A:

$$p(x) = P(X = x), x \in A$$

$$H(p) = H(X) = -\sum_{x \in A} p(x) \log_2 p(x) \tag{16}$$

(Määritellään $0 \log 0 = 0$).

Entropia ilmaistaan tavallisesti biteissä (kaksikantainen logaritmi), mutta muunkantaiset logaritmit yhtä lailla ok.

Esimerkki: 8-sivuisen nopan heittäminen, kommunikoitava yksittäisen heiton tulos.

$$\begin{aligned} H(X) &= -\sum_{i=1}^{8} p(i) \log p(i) = -\sum_{i=1}^{8} \tfrac{1}{8} \log \tfrac{1}{8} \\ &= -\log \tfrac{1}{8} = log 8 = 3\text{bittiä} \end{aligned}$$

Pätee yleisesti: Jos viestin todennäköisyys on $p(i)$, sen optimaalinen koodinpituus on $-\log p(i)$ bittiä.

Vaihtoehtoinen kirjoitustapa Entropian kaavalle:

$$\begin{aligned} H(X) &= -\sum_{x \in A} p(x) \log p(x) = \sum_{x \in A} p(x) \log \tfrac{1}{p(x)} \\ &= E(\log \tfrac{1}{p(x)}) \end{aligned}$$

ts. entropia = optimaalisen koodinpituuden odotusarvo, eli montako bittiä keskimäärin on käytettävä yhden viestin välittämiseen.

Yhteisentropia ja ehdollinen entropia

Kahden muuttujan X ja Y (aakkostot A ja B) yhteisentropia, eli paljonko informaatiota keskimäärin tarvitaan kummankin arvon kommunikointiin:

$$H(X,Y) = -\sum_{x \in A} \sum_{y \in B} p(x,y) \log p(X,Y) \tag{17}$$

Ehdollinen entropia: Jos X on jo kommunikoitu, paljonko lisäinformaatiota keskimäärin tarvitaan Y:n kommunikoimiseen:

$$\begin{aligned} H(Y|X) &= \textstyle\sum_{x \in A} p(x) H(Y|X=x) && (18) \\ &= -\textstyle\sum_{x \in A} \sum_{y \in B} p(x,y) \log p(y|x) && (19) \end{aligned}$$

Entropian ketjusääntö: $H(X,Y) = H(X) + H(Y|X)$

Erikoistapaus: Jos muuttujat riippumattomia toisistaan, kumpikin voidaan kommunikoida erikseen ja laskea koodinpituudet yhteen:
$H(X,Y) = H(X) + H(Y)$

Vrt. todennäköisyyksien ketjusääntö $P(X,Y) = P(X)P(Y|X)$
ja riipumattomille muuttujille $P(X.Y) = P(X)P(Y)$.

Yhteisinformaatio (Mutual Information, MI)

Yhteisinformaatio I muuttujien X ja Y välillä on

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{20}$$

Kohinainen kanava-malli

Binäärinen kommunikointikanava, lähetetään 1 tai 0.
$p =$ todennäköisyys jolla kanavalla lähetetty bitti kääntyy päinvastaiseksi.

Kanavan kapasiteetti C on tällöin:

$$C = \max_{p(X)} I(X;Y) = 1 - H(p) \tag{21}$$

(kaavan johto kirjassa)

Relevanssi kielen mallintamisessa

(Kuva 2.9 sivulla 70, taulukko 2.2. sivulla 71)

Esimerkki: kääntäminen englannista ranskaksi

Kuvitellaan että alun perin teksti oli ranskaa ('vache noire'), mutta kohinainen kanava sotki sen, tuloksena englantia ('black cow')

$p(i)$ = kielimalli l. sanajonojen todennäköisyydet ranskassa
$p(o|i)$ = käännösmalli l. sanajonojen kääntymistodennäköisyydet ranskasta englanniksi
$p(o)$ = sanajonojen todennäköisyydet englannissa
I' = sanajonolle valittava ranskannos

$$I' = \arg\max_{I'} p(i|o) = \frac{p(i)p(o|i)}{p(o)} = \arg\max_{I'} p(i)p(o|i) \qquad (22)$$

($p(o)$ on sama riippumatta siitä mikä I' valitaan, joten voidaan jättää huomiotta.)

3.12 Minimun Description Length (MDL) -periaate

- Lähestymistapa mallin valintaan
- Rissanen et. al.
- Tavoite: pyritään löytämään datalle sellainen koodi että koko datajoukon koodauspituus minimoituu
- Koodinpituus = mallin kuvauspituus + datan kuvauspituus mallin avulla koodattuna + virheiden koodauspituus
- Koodinpituutta (todellista tai laskennallista) käytetään kustannusfunktiona mallia optimoitaessa
- Teoreettinen alaraja koodinpituudelle: entropia
- Suora yhteys myös Bayeslaiseen mallinnukseen

4. Lingvistiikan perustietoja

Itse luettavaa: kirjan luku 3, WWW-sivun linkit

4.1 Kielellisen analyysin eri tasoista

Käsiteltäviä kielellisiä yksiköitä

foneemi, morfeemi, sananmuoto, lekseemi, käsite, lause, virke, kappale, dokumentti, korpus

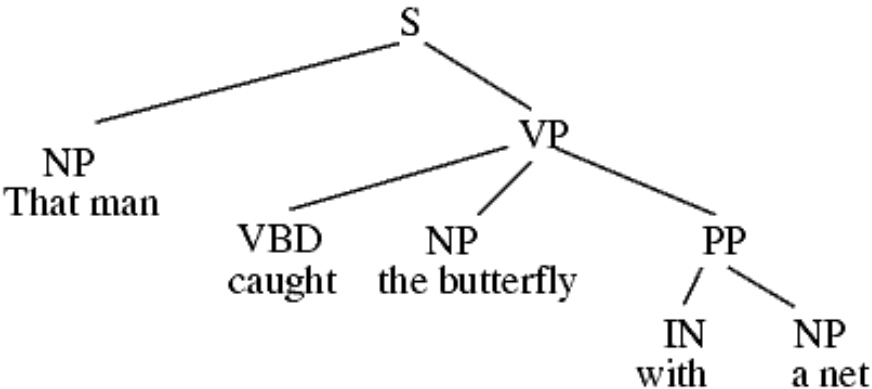
Tiedon lajeja, eri yksiköiden tasoilla

foneettinen ja fonologinen, morfologinen, syntaktinen, semanttinen, pragmaattinen, diskurssitieto, maailmantieto

Esimerkki syntaktisesta analyysista

Käsitteitä: Sanakategoriat, Lauserakennekielioppi, Dependenssikielioppi

Lauserakennekieliopin jäsennyspuu



Esimerkki morfosyntaktisesta analyysistä

Tuotettu Conexorin FDG:llä
FDG=Functional Dependency Grammar

Kääpiösilkkiapina	kääpiö#silkki#apina	&NH N SG NOM
on	olla	&+MV V ACT IND PRES SG3
niin	niin	&ADV ADV &AD> ADV
pieni	pieniä	&+MV V ACT IND PAST SG3
,	,	PUNCT
että	että	&CS CS
se	se	&NH PRON SG NOM
mahtuu	mahtua	&+MV V ACT IND PRES SG3
kämmenelle	kämmen	&NH N SG ALL
.	.	PUNCT

5. Korpustyöskentely

5.1 Tarvittavia välineitä ja tekniikoita

- Ohjelmointikieliä: Perl, Python, C/C++, Prolog, Awk
- Tekstihahmojen tunnistus: säännölliset lausekkeet (regexps)
- Sanojen koodaustapoja: sanojen korvaus numeroilla (taulukointi, hajautus (hash table))
- frekvenssi- tai lukumäärätiedon keräys

5.2 Tekstin esikäsittely

- "roskan" poistaminen
- paloittelu käsiteltäviin yksiköihin (segmentointi tai tokenisointi)
- normalisointi: variaation eliminointi, monitulkintaisuuksien ratkominen

Roskan poistaminen

Kaikki mikä ei ole varsinaista tekstiä poistetaan, esim.

- Samoina toistuvat tiedot dokumenteissa (headerit, footerit)
- Kenttien nimet
- sähköpostiviestien headerit, signaturet

Paloittelu

Mikä on sana? Kahden tyhjän tilan (blanko, space) erottama alue? Puheessa kahden tauon erottama äänisignaali? Entä

- 'database' vs. 'data base'
- yhdyssanat: kesäilta, koti-ilta, venäläissotilaat, sivukonttori, elinkeinoelämä: elin keino elämä, elinkeino elämä, elin keinoelämä?
- rikas taivutus: huomaamattomuudellaansakaankohan huomaa mattomuu dellaan sa kaan ko han ?
- "John's": 1,2 vai 3 sanaa?
- puhelinnumerot, e-mail-osoitteet
- puheessa äänisignaalin tauot eivät osu sananrajoille vaan tietyille konsonanteille

Mikä on virke? (pisteen monitulkintaisuus, muut tavat lopettaa virke)

Mikä on dokumentti? Miten käsitellään pitkät dokumentit esim. indeksoitaessa tiedonhakua varten, paloina vai kokonaisina?

5.3 Variaatio informaation koodaustavoissa

Esim. käsite ITSEORGANISOIVA KARTTA; ilmauksia SOM, SOFM, self-organizing map, self-organising map, self organizing map, Kohonen map, Kohonen SOM, Kohonen net.

Suomeksi: itseorganisoiva kartta, itseorganisoiTUva kartta, itsejärjestyvä kartta, Kohosen kartta, Kohosen verkko

Puhelinnumerojen koodaustapoja mm.:

040 123 4567	Suomi
040-123 4567	
040-1234567	
+358-40-1234567	
(040) 123 4567	
+411/284 3797	Sveitsi
(44.171) 830 1007	UK
+44 (0) 171 830 1007	
1-925-225-3000	USA
212.995.5402	

Information extraction, informaation ekstrahointi: yritetään oppia eri tavat ilmaista sama semanttinen informaatio. Tämä on *hahmontunnistusta*, pyrkimyksenä tunnistaa tietyn semanttisen informaation tyypit.

Morfologisen monimuotoisuuden käsittely

Sanan katkaisu 'juurimuotoon' (stemming)
'istuimme', 'istuttiin' .. → 'istu'
'yöt'→ 'yö'
'öisin'→ 'öi'

sanan perusmuotoistus
etsimme' → 'etsiä'
'lying' → 'lie'/'lay'
'istui', 'istuu', 'istunut', 'istumme' → 'istua'
'istuisitko', 'istuttaisiinko', 'ISTU!' → 'istua'

Kannattaako morfologinen variaation normalisointi tehdä, tai missä määrin?

Riippuu tavoitteesta, esim. halutaanko analysoida keskusteluja yksityiskohtaisesti (esim. interaktiivinen keskustelujärjestelmä) vai representoida pääasiallise puheenaiheet (esim. tiedonhaku).

5.4 Monitulkintaisuus (ambiguity)

Samalla sanalla tai ilmauksella voi olla monta erilaista tulkintaa:

- Englannin *title*: kirjan otsikko, elokuvan nimi, henkilön nimen etuliite tai arvonimi, omistusoikeus, jne
- *Kun lakkaa satamasta, hae lakkaa satamasta.*
- *'...pääsi perille...'* Montako mahdollista tulkintaa?

Edellisen kalvon monitulkintaisuuskysymykseen liittyen
(Lingsoftin TWOL)

<*pääsi*> "pää" N NOM SG 2SG
"pää" N GEN SG 2SG
"pää" N NOM PL 2SG
"päästä" V PAST ACT SG3

<*perille*> "perä" N ALL PL
"perille" ADV ALL
"per" PROP N ALL SG

Mahdollisia tulkintoja ainakin:

- saapui sinne minne oli menossa
- saapui Per:in luo
- 'paina tämä asia pääsi perimmäiseen nurkkaan'
- 'näytitkö pääsi Perille?' (jos siinä oli vaikkapa haava ja Per on lääkäri)

Disambiguointi I. yksikäsitteistäminen

- Valitaan potentiaalisista tulkinnoista oikea tai todennäköisin
- Disambiguointi on tyypillinen kieliteknologinen tehtävä. Esim. morfologian yksikäsitteistäminen, lauserakenteen yksikäsitteistäminen, sananmerkitysten yksikäsitteistäminen
- Voidaan periaatteessa tehdä kontekstin perusteella; edellyttää *mallia* kontekstin ja vaihtoehtoisten tulkintojen välisistä riippuvuuksista.
- Tilastollinen malli: ehdollinen todennäköisyysjakauma $p(\text{tulkinta}|\text{konteksti})$
- Ei-tilastollinen malli: kategoriset säännöt muotoa 'JOS konteksti NIIN tulkinta'.

5.5 Taggaus

Syntaktisia tagijoukkoja Englannille

Brown, Penn, Claws 1-3

Taggauksen ääripäät eri tarkoituksiin

Puheentunnistus: Pelkästään puheessa esiintyvät sanat (ei välttämättä edes välimerkkejä)

Keskusteluanalyysi: vuoronvaihdot, vuorotyypit, epäröinnit, hiljaisuuden kesto jne.

Category	Examples	Claws c5	Brown	Penn
Adjective	happy, bad	AJ0	JJ	JJ
Adjective, ordinal number	sixth, 72nd, last	ORD	OD	JJ
Adjective, comparative	happier, worse	AJC	JJR	JJR
Adjective, superlative	happiest, worst	AJS	JJT	JJS
Adjective, superlative, semantically	chief, top	AJ0	JJS	JJ
Adjective, cardinal number	3, fifteen	CRD	CD	CD
Adjective, cardinal number, one	one	PNI	CD	CD
Adverb	often, particularly	AV0	RB	RB
Adverb, negative	not, n't	XX0	*	RB
Adverb, comparative	faster	AV0	RBR	RBR
Adverb, superlative	fastest	AV0	RBT	RBS
Adverb, particle	up, off, out	AVP	RP	RP
Adverb, question	when, how, why	AVQ	WRB	WRB
Adverb, degree & question	how, however	AVQ	WQL	WRB
Adverb, degree	very, so, too	AV0	QL	RB
Adverb, degree, postposed	enough, indeed	AV0	QLP	RB
Adverb, nominal	here, there, now	AV0	RN	RB
Conjunction, coordination	and, or	CJC	CC	CC
Conjunction, subordinating	although, when	CJS	CS	IN
Conjunction, complementizer *that*	that	CJT	CS	IN
Determiner	this, each, another	DT0	DT	DT
Determiner, pronoun	any, some	DT0	DTI	DT
Determiner, pronoun, plural	these, those	DT0	DTS	DT
Determiner, prequalifier	quite	DT0	ABL	PDT
Determiner, prequantifier	all, half	DT0	ABN	PDT
Determiner, pronoun or double conj.	both	DT0	ABX	DT (CC)
Determiner, pronoun or double conj.	either, neither	DT0	DTX	DT (CC)
Determiner, article	the, a, an	AT0	AT	DT
Determiner, postdeterminer	many, same	DT0	AP	JJ
Determiner, possessive	their, your	DPS	PP$	PRP$
Determiner, possessive, second	mine, yours	DPS	PP$$	PRP
Determiner, question	which, whatever	DT0	WDT	WDT

6. Kollokaatiot

6.1 Mitä on kollokaatio

- Kahdesta tai useammasta sanasta koostuva konventionaalistunut ilmaus (Manning & Schutze)
- Collocations of a given word are statements of the habitual or customary places of that word (Firth, 1957)
- Esimerkkejä:
  - 'weapons of mass destruction', 'disk drive', 'part of speech' (suomessa yhdyssanoina 'joukkotuhoaseet', 'levyasema', 'sanaluokkatieto')
  - 'bacon and eggs'
  - verbin valinta: 'prendre une décision', mutta 'make a decision' ei 'take a decision'.

  - adjektiivin valinta: 'strong tea' mutta ei 'powerful tea'; 'vahvaa teetä', harvemmin 'voimakasta teetä' (valinnat voivat heijastaa kulttuurin asenteita: strong → tea, coffee, cigarettes powerful → drugs, antidote)
  - 'kick the bucket', 'heittää veivinsä' (kiertoilmaus, sanonta, idiomi)

- Olentoja, yhteisöjä, paikkoja tai tapahtumia yksilöivät nimet: 'White House' Valkoinen talo, 'Tarja Halonen', 'Persianlahden sota' (viittaa tiettynä ajankohtana käytyyn sotaan)

- Kollokaation kanssa osittain päällekkäisiä käsitteitä: termi, tekninen termi, terminologinen fraasi. Huom: tiedonhaussa sanalla 'termi' laajempi merkitys: 'sana tai kollokaatio'.

Kollokaatioiden ominaisuuksia

- Ei-kompositionaalisuus: kollokaation merkitys ei ole (täysin) selitettävissä osiensa summana
- Ei-vaihdettavuus: 'white wine' mutta ei 'yellow wine' ('valkoviini', ei 'keltainen viini', todellisesta väristä huolimatta)
- (Ei-muutettavuus: 'Ihmiset seurasivat tilannetta *pala kurkussa*', ei 'pieni pala kurkussa' tai 'palat kurkuissa'. 'Pitäkää päät pystyssä' vs. 'pitäkää pää pystyssä'.)

6.2 Sovelluskohteita

- Luonnollisen kielen generointi: vältetään kummallisia sanavalintoja.
- Lauseenjäsennys: suositaan konventionaalisia sanayhdistelmiä rakenteellisina yksiköinä rakenteen disambiguoinnissa.
- Leksikografia (sanakirjojen tekeminen): mitkä ilmaukset valitaan sanakirjassa selitettäviin.
- Tiedonhaku ja indeksointi: sanaparien valinta indeksointitermeiksi.
- Tietokoneavusteinen kielenkääntäminen: tekniset termit käännettävä systemaattisesti samaa ilmausta käyttäen.
- Korpuslingvistiikka: kulttuuristen asenteiden ja arvostusten tutkiminen, mm. eri nautintoaineita tai eri sukupuolia kohtaan.

6.3 Sanan frekvenssi ja sanaluokkasuodatus

Pelkän frekvenssin käyttö

Esimerkki: Onko luontevampaa sanoa 'strong tea' vai 'powerful tea'?
Ratkaisu: Etsitään altavistasta: 'strong tea' 799, 'powerful tea' 17

Joihinkin täsmällisiin kysymyksiin riittävä tapa. Kuitenkin järjestettäess bigrammeja frekvenssin mukaan, parhaita ovat 'of the', 'in the', 'to the', ...

Frekvenssi + sanaluokka

Jos tunnetaan kunkin sanan sanaluokka, sekä osataan kuvailla kollokaatioiden 'sallitut' sanaluokkahahmot:

- Järjestetään sanaparit tai -kolmikot yleisyyden (lukumäärä) mukaan
- Hyväksytään vain tietyt sanaluokkahahmot:
  AN, NN, AAN, ANN, NAN, NNN, NPN (Justeson & Katz's POS filter)

$C(w^1\ w^2)$	w^1	w^2	Tag Pattern
11487	New	York	A N
7261	United	States	A N
5412	Los	Angeles	N N
3301	last	year	A N
3191	Saudi	Arabia	N N
2699	last	week	A N
2514	vice	president	A N
2378	Persian	Gulf	A N
2161	San	Francisco	N N
2106	President	Bush	N N
2001	Middle	East	A N
1942	Saddam	Hussein	N N
1867	Soviet	Union	A N
1850	White	House	A N
1633	United	Nations	A N
1337	York	City	N N
1328	oil	prices	N N
1210	next	year	A N
1074	chief	executive	A N
1073	real	estate	A N

Table 5.3 Finding Collocations: Justeson and Katz' part-of-speech filter.

6.4 Sanojen etäisyyden keskiarvo ja varianssi

Entä joustavammat kollokaatiot joiden keskellä on kollokaatioon kuulumattomia sanoja?

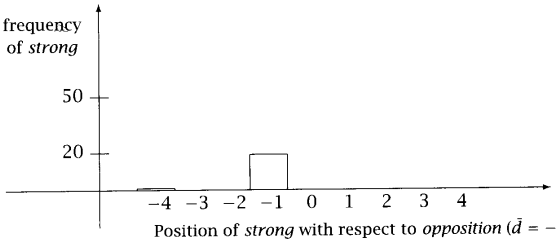
Lasketaan etäisyyden keskiarvo ja varianssi. Jos keskiarvo nollasta poikkeava ja varianssi pieni, potentiaalinen kollokaatio (Huom: oletetaan siis etäisyyden jakautuvan Gaussisesti).

Esim. '*knock ... door*' (ei 'hit', 'beat', tai 'rap'):
a) 'She *knocked* on his *door*'
b) 'They *knocked* at the *door*'
c) '100 women *knocked* on Donaldson's *door*'
d) 'a man *knocked* on the metal front *door*'

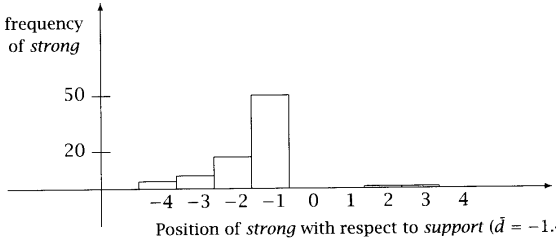
- Liu'uta kiinteän kokoista ikkunaa tekstin yli (leveys esim. 9) ja kerää kaikki sanaparin esiintymät koko tekstissä
- Sanojen etäisyyksien keskiarvo:

$\bar{d} = 1/n \sum_{i=1}^{n} d_i = 1/4(3+3+5+5) = 4.0$
(jos heittomerkki ja 's' lasketaan sanoiksi)

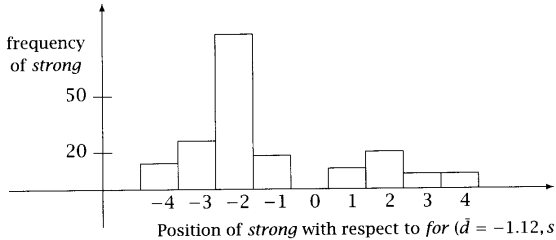
- Varianssin s^2 estimointi pienillä näytemäärillä:
  $s^2 = \frac{\sum_{i=1}^{n}(d_i - \bar{d})^2}{n-1} = 1/3((3-4.0)^2 + (3-4.0)^2 + (5-4.0)^2 + (5-4.0)^2)$
  $s = 1.15$



Position of *strong* with respect to *opposition* ($\bar{d} = -1.15, s = 0.67$).



Position of *strong* with respect to *support* ($\bar{d} = -1.45, s = 1.07$).



Position of *strong* with respect to *for* ($\bar{d} = -1.12, s = 2.15$).

s	$\bar{d}$	Count	Word 1	Word 2
0.43	0.97	11657	New	York
0.48	1.83	24	previous	games
0.15	2.98	46	minus	points
0.49	3.87	131	hundreds	dollars
4.03	0.44	36	editorial	Atlanta
4.03	0.00	78	ring	New
3.96	0.19	119	point	hundredth
3.96	0.29	106	subscribers	by
1.07	1.45	80	strong	support
1.13	2.57	7	powerful	organizations
1.01	2.00	112	Richard	Nixon
1.05	0.00	10	Garrison	said

Table 5.5 Finding collocations based on mean and variance. Sample deviation s and sample mean $\bar{d}$ of the distances between 12 word pairs.

Pohdittavaksi:

1. Mitä tapahtuu jos sanoilla on kaksi tai useampia tyypillisiä positioita suhteessa toisiinsa?
2. Mikä merkitys on ikkunan leveydellä?

6.5 Hypoteesin testaus

Onko suuri osumamäärä yhteensattumaa (esim. johtuen siitä että jommankumman perusfrekvenssi on suuri)? Osuvatko kaksi sanaa yhteen useammin kuin sattuma antaisi olettaa?

1. Formuloi *nollahypoteesi* H_0: assosiaatio on sattumaa
2. Laske tn p että sanat esiintyvät yhdessä jos H_0 on tosi
3. Hylkää H_0 jos p liian alhainen, alle merkitsevyystason, esim $p < 0.05$ tai $p < 0.01$.

Nollahypoteesia varten sovelletaan riippumattomuuden määritelmää, kaava 2. Oletetaan että sanaparin todennäköisyys, jos H_0 on tosi, on kummankin sanan oman todennäköisyyden tulo:
$P(w^1 w^2) = P(w^1)P(w^2)$

T-testi

Tilastollinen testi sille eroaako havaintojoukon odotusarvo oletetun, datan generoineen jakauman odotusarvosta. Olettaa että todennäköisyydet ovat suunnilleen normaalijakautuneita.

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}}, \text{ jossa} \tag{23}$$

$\bar{x}, s^2$: näytejoukon keskiarvo ja varianssi, $N =$ näytteiden lukumäärä, ja $\mu =$ jakauman keskiarvo. Valitaan haluttu p-taso (0.05 tai pienempi). Luetaan tätä vastaava t:n yläraja taulukosta. Jos t suurempi, H_0 hylätään.

Soveltaminen kollokaatioihin: Nollahypoteesina että sanojen yhteisosumat ovat satunnaisia (oletetaan sanoille Bernoullijakauma): Esimerkki: H_0 : $P(new\ companies) = P(new)P(companies)$

$\mu = P(new)P(companies)$
$\bar{x} = \frac{c(new\ companies)}{c(\cdot,\cdot)} = \hat{p}$

$s^2 = p(1-p) = \hat{p}(1-\hat{p}) \approx \hat{p}$ (pätee Bernoulli-jakaumalle)
$N = c(\cdot, \cdot)$

- Järjestetään sanat paremmuusjärjestykseen mitan mielessä TAI
- Hypoteesin testaus: valitaan merkittävyystaso (p=0.05 tai p=0.01) ja katsotaan t-testin taulukosta arvo jonka ylittäminen tarkoittaa nolla-hypoteesin hylkäystä.

Vertaillaan yhtä suuren frekvenssin omaavia bigrammeja keskenään t-testillä:

t	$C(w^1)$	$C(w^2)$	$C(w^1\ w^2)$	w^1	w^2
4.4721	42	20	20	Ayatollah	Ruhollah
4.4721	41	27	20	Bette	Midler
4.4720	30	117	20	Agatha	Christie
4.4720	77	59	20	videocassette	recorder
4.4720	24	320	20	unsalted	butter
2.3714	14907	9017	20	first	made
2.2446	13484	10570	20	over	many
1.3685	14734	13478	20	into	them
1.2176	14093	14776	20	like	people
0.8036	15019	15629	20	time	last

Table 5.6 Finding collocations: The t test applied to 10 bigrams that occur with frequency 20.

Esimerkki soveltamisesta muuhun ongelmaan: Vertailu mitkä lähikonteks sanat parhaiten erottelevat sanoja 'strong' ja 'powerful'

t	$C(w)$	$C(strong\ w)$	$C(powerful\ w)$	Word
3.1622	933	0	10	computers
2.8284	2337	0	8	computer
2.4494	289	0	6	symbol
2.4494	588	0	6	machines
2.2360	2266	0	5	Germany
2.2360	3745	0	5	nation
2.2360	395	0	5	chip
2.1828	3418	4	13	force
2.0000	1403	0	4	friends
2.0000	267	0	4	neighbor
7.0710	3685	50	0	support
6.3257	3616	58	7	enough
4.6904	986	22	0	safety
4.5825	3741	21	0	sales
4.0249	1093	19	1	opposition
3.9000	802	18	1	showing
3.9000	1641	18	1	sense
3.7416	2501	14	0	defense
3.6055	851	13	0	gains
3.6055	832	13	0	criticism

Table 5.7 Words that occur significantly more often with *powerful* (the first ten words) and *strong* (the last ten words).

6.6 Pearsonin Chi-toiseen-testi χ^2

- χ^2-testi mittaa kahden muuttujan välistä riippuvuutta. Perustuu riippumattomuuden määritelmään: mikäli muuttujat ovat riippumattomia, yhteisjakauman arvon tietyssä jakauman pisteessä on marginaalijakaumien tulo.
- Kahden muuttujan jakauma voidaan kuvata 2-ulotteisena kontingenssitaulukkona ($r \times c$).
- Lasketaan *kussakin taulukon pisteessä* (i, j) erotus havaitun jakauman O (todellinen yhteisjakauma) ja odotetun jakauman E (marginaalijakaumien tulo) välillä, ja summataan, skaalattuna jakauman odotusarvolla:

$$\chi^2 = \sum_{i,j} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}} \tag{24}$$

  jossa siis $E_(i, j) = O(i, \cdot) * O(\cdot, j)$.
- χ^2 on *asymptoottisesti* χ^2-jakautunut.Ongelma kuitenkin: herkkä harvalle datalle.

- Nyrkkisääntö: älä käytä testiä jos $N < 20$ tai jos $20 \leq N \leq 40$ ja jokin $E_{i,j} \leq 5$

Soveltaminen kollokaatioiden tunnistamiseen

Formuloidaan ongelma siten että kumpaakin sanaa vastaa yksi satunnaismuuttuja joka voi saada kaksi arvoa (sana joko esiintyy tai ei esiinny yksittäisessä sanaparissa).

Sanojen yhteistnjakauma voidaan tällöin esittää 2×2 taulukkoina. Esim.

	$w_1 = new$	$w_1 \neq new$
$w_2 = companies$	8	4667
$w_2 \neq companies$	15280	14287173

2×2-taulukon tapauksessa kaava 24 voidaan johtaa muotoon:

$$\chi^2 = \frac{N(O_{11}O_{22} - O_{12}O_{21})^2}{(O_{11} + O_{12})(O_{11} + O_{21})(O_{12} + O_{22})(O_{21} + O_{22})}$$

- Järjestetään sanat paremmuusjärjestykseen mitan mielessä TAI
- Hypoteesin testaus: valitaan merkittävyystaso (p=0.05 tai p=0.01) ja katsotaan χ^2-taulukosta arvo jonka ylittäminen tarkoittaa nollahypo-

teesin hylkäystä.

Ongelmallisuus kollokaatioiden tunnistamisen kannalta

Tässä soveltamistavassa ei erotella negatiivista ja positiivista riippuvuutta. Ts. jos sanat vierastavat toisiaan, testi antaa myös suuren arvon, koska tällöin sanojen esiintymisten välillä todellakin on riippuvuus. Kollokaatioita etsittäessä ollaan kuitenkin kiinnostuttu vain positiivisista riippuvuuksista.

Johtopäätös: Ainakaan näin soveltaminen ei välttämättä kannata.

Muita (parempia?) sovelluksia χ^2-testille:

- Konekäännös: Linjattujen korpusten sana-käännösparien tunnistaminen (cow, vache yhteensattumat johtuvat riippuvuudesta)
- Metriikka kahden korpuksen väliselle samankaltaisuudelle: $n \times 2$-taulukko jossa kullekin tutkittavalle sanalle $w_i, i \in (1 \ldots n)$ kerrotaan ko. sanan lukumäärä korpuksessa j

6.7 Uskottavuuksien suhde

Kuinka paljon uskottavampi H_2 on kuin H_1? Lasketaan hypoteesien uskottavuuksien suhde λ:

$$\log \lambda = \log \frac{L(H_1)}{L(H_2)}$$

Esimerkki:
H_1: w_1 ja w_2 riippumattomia: $P(w_2|w_1) = p = P(w_2|\ \not w_1)$
H_2: w_1 ja w_2 eivät riippumattomia: $P(w_2|w_1) = p1 \neq p2 = P(w_2|\ \not w_1)$

Oletetaan selvä positiivinen riippuvuus, eli $p_1 \ll p_2$.

Käytetään ML-estimaatteja (keskiarvoja) laskettaessa p, p_1 ja p_2:
$p = \frac{c_2}{N}$, $p_1 = \frac{c_{12}}{c_1}$, $p_2 = \frac{c_2 - c_{12}}{N - c_1}$

Oletetaan binomijakaumat. Esim. $p(w_2|w_1) = b(c_{12}; c_1, p)$. Ilmaistaan kunkin mallin yhtäaikaa voimassa olevat rajoitteet tulona. Lopputulos: kirjan kaava 5.10.

$\log \lambda$ on asymptoottisesti χ^2-jakautunut. On lisäksi osoitettu että harval-

la datalla uskottavuuksien suhteella saadaan parempi approksimaatio χ^2-jakaumalle kuin χ^2-testillä.

6.8 Suhteellisten frekvenssien suhde

Etsitään kollokaatioita jotka ovat *ominaisia tietylle keskustelunaiheelle* (subject). Verrataan frekvenssejä korpuksissa A ja B joista toinen on yleisaiheinen, toinen erityisaiheinen:

$$r = \frac{c_1^A / N_A}{c_1^B / N_B}$$

jossa c_1^A on sanan 1:n lukumäärä korpuksessa A jne.

$-2 \log \lambda$	$C(w^1)$	$C(w^2)$	$C(w^1 w^2)$	w^1	w^2
1291.42	12593	932	150	most	powerful
99.31	379	932	10	politically	powerful
82.96	932	934	10	powerful	computers
80.39	932	3424	13	powerful	force
57.27	932	291	6	powerful	symbol
51.66	932	40	4	powerful	lobbies
51.52	171	932	5	economically	powerful
51.05	932	43	4	powerful	magnet
50.83	4458	932	10	less	powerful
50.75	6252	932	11	very	powerful
49.36	932	2064	8	powerful	position
48.78	932	591	6	powerful	machines
47.42	932	2339	8	powerful	computer
43.23	932	16	3	powerful	magnets
43.10	932	396	5	powerful	chip
40.45	932	3694	8	powerful	men
36.36	932	47	3	powerful	486
36.15	932	268	4	powerful	neighbor
35.24	932	5245	8	powerful	political
34.15	932	3	2	powerful	cudgels

Table 5.12 Bigrams of *powerful* with the highest scores according to Dunning's likelihood ratio test.

6.9 Pisteittäinen yhteisinformaatio

Muistellaan entropian $H(x)$ ja yhteisinformaation $I(x;y)$ kaavoja:

$$\begin{aligned} H(x) &= -E(\log p(x)) \\ I(X;Y) &= H(Y) - H(Y|X) = (H(X) + H(Y)) - H(X,Y) \\ &= E_{X,Y}(\log \tfrac{p(X,Y)}{p(X)p(Y)}) \end{aligned}$$

joka kuvastaa *keskimääräistä* informaatiota jonka sekä x että y sisältävät.

Määritellään *pisteittäinen yhteisinformaatio* joidenkin tiettyjen tapahtumien x ja y välillä (Fano, 1961):
$I(x;y) = \log \frac{p(x,y)}{p(x)p(y)}$

Voidaanko käyttää kollokaatioiden valintaan? Motivaationa intuitio: jos sanojen välillä on suuri yhteisinformaatio (ts. niiden kummankin kommunikoiman informaation yhteinen osuus on suuri), voisi olettaa että kyse on kollokaatiosta.

$I(w^1, w^2)$	$C(w^1)$	$C(w^2)$	$C(w^1\ w^2)$	w^1	w^2
18.38	42	20	20	Ayatollah	Ruhollah
17.98	41	27	20	Bette	Midler
16.31	30	117	20	Agatha	Christie
15.94	77	59	20	videocassette	recorder
15.19	24	320	20	unsalted	butter
1.09	14907	9017	20	first	made
1.01	13484	10570	20	over	many
0.53	14734	13478	20	into	them
0.46	14093	14776	20	like	people
0.29	15019	15629	20	time	last

Table 5.14 Finding collocations: Ten bigrams that occur with frequency 20, ranked according to mutual information.

Taulukosta 5.16 huomataan että jos jompikumpi sanoista on harvinainen, saadaan korkeita lukuja.

Täydellisen riippuville sanoille yhteisinformaatio:

$$I(x;y) = \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x)}{p(x)p(y)} = \log \frac{1}{P(y)}$$

kasvaa kun sanat muuttuvat harvinaisemmiksi. Ääritilanne: kaksi sanaa esiintyy kumpikin vain kerran, ja tällöin yhdessä. Kuitenkin tällöin evidenssiä kollokaationa toimimisesta on vähän, mikä jää huomiotta.

Johtopäätös: Ei kovin hyvä mitta tähän tarkoitukseen, harhaanjohtava etenkin pienille todennäköisyyksille. Seurauksena kärsii datan harvuudesta erityisen paljon.

I_{1000}	w^1	w^2	w^1w^2	Bigram	I_{23000}	w^1	w^2	w^1w^2	Bigram
16.95	5	1	1	Schwartz eschews	14.46	106	6	1	Schwartz eschew
15.02	1	19	1	fewest visits	13.06	76	22	1	FIND GARDEN
13.78	5	9	1	FIND GARDEN	11.25	22	267	1	fewest visits
12.00	5	31	1	Indonesian pieces	8.97	43	663	1	Indonesian piece
9.82	26	27	1	Reds survived	8.04	170	1917	6	marijuana growi
9.21	13	82	1	marijuana growing	5.73	15828	51	3	new converts
7.37	24	159	1	doubt whether	5.26	680	3846	7	doubt whether
6.68	687	9	1	new converts	4.76	739	713	1	Reds survived
6.00	661	15	1	like offensive	1.95	3549	6276	6	must think
3.81	159	283	1	must think	0.41	14093	762	1	like offensive

Table 5.16 Problems for Mutual Information from data sparseness. The tab shows ten bigrams that occurred once in the first 1000 documents in the r erence corpus ranked according to mutual information score in the first 10 documents (left half of the table) and ranked according to mutual informati score in the entire corpus (right half of the table). These examples illustrate th a large proportion of bigrams are not well characterized by corpus data (even l large corpora) and that mutual information is particularly sensitive to estimat that are inaccurate due to sparseness.

7. Sananmerkitysten yksikäsitteistäminen (word sense disambiguation)

Ongelman määrittely: Oletetaan sana w jolle olemassa k erillistä merkitystä $s_1 \ldots s_k$. Tehtävänä on päätellä yksittäisen esiintymän osalta mikä merkitys on kyseessä. Kyse on siis *hahmontunnistuksesta* tai *luokittelusta*.

Hyödyllisiä aineistotyyppejä

- Sense-tagged corpora: aineisto johon on jokaisen sanan w esiintymän kohdalle tagattu kyseisen esiintymän merkitys s_i.
  Esim. Senseval (englanninkielinen).
- Sanakirjat ja thesaurukset, ks. esim. 'shake' http://www.britannica.com
- Kaksikielinen 'linjattu' aineisto: sama aineisto molemmilla kielillä, johon on merkattu toisiaan vastaavat kohdat, esim. sana tai lause kerrallaan.

- Yksikielinen aineisto jossa on 'siemeneksi' sense-tagattu pieni osa sanan esiintymistä
- Yksikielinen aineisto jossa paljon sanan esiintymiä kontekstissa

Korpukset voivat lisäksi olla syntaktisesti tagattuja (sanaluokat jne) tai sisältää pelkän tekstin.

7.1 Eri oppimisperiaatteista yleensä

(Lainausta hahmontunnistuskurssilta)

Tunnistusmenetelmät voidaan jakaa ryhmiin mm. oppimisperiaatteen mukaan:

Ohjaamaton oppiminen

- Hahmojen *luokkia ei tiedetä* etukäteen
- Tavoitteena on muodostaa hahmoista ryhmiä, joiden sisällä hahmot ovat samankaltaisia ja joiden välillä on selkeitä eroja (klusterointi)
- Optimoitava funktio on klusteroinnin onnistumista kuvaava mitta
- Aina ei tiedetä edes ryhmien lukumäärää

Ohjattu oppiminen

- Hahmojen *luokat tunnetaan* etukäteen
- Tavoitteena on muodostaa kuvaus piirreavaruudesta luokka-avaruuteen
- Optimoitava funktio perustuu kuvauksessa tapahtuviin virheisiin, ts. pyritään minimoimaan tapahtuvien *luokitteluvirheiden todennäköisyys* tai, mikäli virheisiin liittyy toisistaan poikkeavia kustannuksia, virheiden kokonaiskustannuksen odotusarvo.

Kolmas oppimisperiaate on *vahvistettu oppiminen* jota ei kuitenkaan käsitellä tällä kurssilla.

Luonnollisen kielen aineistoilla relevanttia on lisäksi ns. 'bootstrapping'-oppimi Pieni osa aineistosta on luokiteltua, jonka avulla päästään alkuun. Tämän jälkeen oppiminen tapahtuu ohjaamattomasti.

7.2 Menetelmien onnistumisen mittaaminen

Keinotekoinen data: pseudosanat

- Menetelmiä voidaan kehittää ja testata keinotekoisella datalla, jonka ominaisuudet varmasti tunnetaan.
- Esim. korvataan kaikki sanojen 'banaani' ja 'ovi' esiintymät pseudosanalla 'banaaniovi'. Mitataan kuinka hyvin onnistutaan tunnistamaan kutakin 'banaaniovea' vastaava oikea sana.
- Helppo, halpa ja nopea tuottaa laajoja testiaineistoja joissa tunnetaan sekä disambiguoimaton data että alkuperäinen, oikea, data, joka menetelmän pitäisi löytää.

Onnistumisen laskennalliset ylä- ja alarajat

Mikäli yhteisiä testiaineistoja ei ole, pelkkä numeerinen tulos ei riitä menetelmien onnistumisen mittaamiseen: jotkut ongelmat ovat luonnostaan vaikeampia kuin toiset.

Pyritään siksi hahmottamaan ongelman vaikeus:

- Yläraja 'ground truth': Paras mahdollinen tulos. Usein käytetään mittana ihmisen suoriutumista samasta tehtävästä (harvoin 100%).
- Ylärajan määritys tärkeää esim. jos verrataan menetelmien suoriutumista rajallisen mittaisella kontekstilla. Harvoin 100%, esim. jos ikkuna kovin kapea.
- Alaraja, 'baseline': yksinkertaisin mahdollinen perusmenetelmä. Esim. luokan valinta satunnaisesti, tai luokan taustafrekvenssin perusteella.

7.3 Ohjattu disambiguointi

Käytetään notaatiota:

w	monimerkityksinen sana
$s_1 \ldots s_K$	sanan eri merkitykset (senses)
$c_1 \ldots c_I$	sanan w kontekstit korpuksessa
$v_1 \ldots v_J$	piirrejoukko (esim. joukko sanoja) jota käytetään disambiguoinni

Seuraavaksi esitellään joitain ohjatun oppimisen lähestymistapoja joita on todella sovellettu merkitysten disambiguointiongelmaan.

Piirteiden valinta

- Yleisesti piirrejoukko vaikuttaa suuresti luokittelun onnistumismahdollisuuksiin.
- Hyvä piirrejoukko on riippuvainen käytetyn luokittimen (tai mallin) ominaisuuksista, eli piirrejoukon valintaa ja mallinnusta ei voida täysin erottaa toisistaan.

Esimerkkejä mahdollisista piirteistä:

- tietyn sanan esiintyminen jonkin etäisyyden päässä disambiguoitavasta sanasta,
  esim. etäisyys(w,'avasi') < 3
- tiettyjen kahden sanan esiintyminen kontekstissa yhdessä
- tietyn sanaluokan tai morfologisen luokan esiintymisfrekvenssi kontekstiikkunassa (jos voidaan olettaa että data on POS- tai morfol. tagattua)
- tietyn sanan tai sanaluokan esiintyminen tietyssä täsmällisessä positiossa suhteessa disambiguoitavaan sanaan (esim. edeltävänä sanana)
- Tieto jonkin semanttisen muuttujan, esim. keskustelunaihe, arvosta (jos sellainen syystä tai toisesta tunnetaan)
- Jokin ylläolevien funktio tai yhdistelmä

Bayesläinen luokitin

- Luokitin ei tee piirrevalintaa, vain yhdistää evidenssin eri piirteistä
- Valitaan piirteiksi joukko sanoja
- Luokitin soveltaa *Bayesin päätössääntöä* valitessaan luokan, ts. minimoi luokitteluvirheen todennäköisyyttä:

$$P(s_k|c) = \frac{P(c|s_k)P(s_k)}{P(c)} \tag{25}$$

  $P(s_k)$ on merkityksen s_k *prioritodennäköisyys*, eli tn jos emme tiedä kontekstista mitään.

- Jos tehtävänä on vain valita todennäköisin luokka, voidaan jättää kontekstin c todennäköisyys $P(c)$ (joka ei riipu luokasta) laskuissa huo-

miotta: Valitaan merkitys s' jos

$$
\begin{aligned}
s' &= \arg\max_{s_k} P(s_k|c) && (26)\\
&= \arg\max_{s_k} \frac{P(c|s_k)P(s_k)}{P(c)} && (27)\\
&= \arg\max_{s_k} P(c|s_k)P(s_k) && (28)\\
&= \arg\max_{s_k} [\log P(c|s_k) + \log P(s_k)] && (29)
\end{aligned}
$$

- Käytännön hankaluus: kontekstin piirteiden ehdollisen yhteistnjakauman $P(c|s_k)$ luotettava estimointi tietylle merkitykselle edellyttäisi että meillä olisi datajoukko jossa jokainen merkitys esiintyisi kaikissa periaatteessa mahdollisissa konteksteissaan, mieluiten useita kertoja.

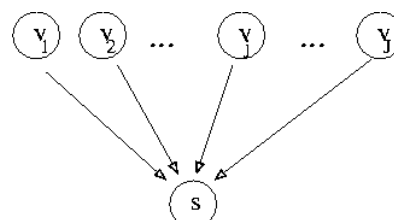
- Ratkaisu: Helpotetaan estimointia tekemällä sopivia yksinkertaistavia oletuksia . . . (Naive Bayes)

Naive Bayes-luokitin

- *Naive Bayes -oletus* tarkoittaa että oletetaan että kukin piirre vaikuttaa luokitukseen toisista piirteistä riippumattomasti:

$$P(c|s_k) = P(v_1, \ldots, v_J|s_k) = \prod_{v_j \text{in} c} P(v_j|s_k) \tag{30}$$

Sama graafisesti:



- Tässä yksinkertaistetussa mallissa (jota kutsutaan myös 'bag of words'-malliksi) sanojen järjestyksellä kontekstissa ei ole merkitystä, ja sama sana voi esiintyä kontekstissa useita kertoja.

- Sovellettaessa edelliseen päätössääntöön (kaava 29) saadaan *Naive Bayes päätössääntö*:

$$\text{Valitaan } s' \text{ jos } s' = \arg\max_{s_k}[\log P(s_k) + \sum_{v_j \text{in} c} P(v_j|s_k)] \qquad (31)$$

- Näistä $P(v_j|s_k)$:lle ja $P(s_k)$:lle lasketaan ML-estimaatit luokitellusta opetusdatajoukosta:

$$\begin{aligned} P(v_j|s_k) &= \frac{C(v_j, s_k)}{C(s_k)} \\ P(s_k) &= \frac{C(s_k)}{C(w)} \end{aligned}$$

jossa C(...) tarkoittaa lukumäärää opetusdatajoukossa

- Kuuden substantiivin *duty, drug, land, language, position, sentence* luokituksessa kun aineistona oli Hansard-korpus saatiin 90% tunnistustulos (Church, Gale & Yarowsky, 1992).
  Esimerkkejä 'drug'-sanan merkityksille sovelletuista piirteistä:

Merkitys	Piirteet ko. merkitykselle
medication	prices,prescription, patent, increase, consumer, pha
illegal substance	abuse, paraphernalia, illicit, alcohol, cocaine, traffi

- Naive Bayes-luokitin on yksinkertainen ja melko robusti; antaa kohtuullisia tuloksia monenlaisissa ongelmissa.

- Naive Bayes-ongelma:epärealistiset riippumattomuusoletukset

- Piste-estimoinnin (ML-estimoinnin) ongelma: käyttää kaikki piirteet, ts. ei kykene tekemään piirrevalintaa

Eräs informaatioteoreettinen lähestymistapa

(Muitakin informaatioteoreettisia lähestymistapoja voi varmasti keksiä)

- Edellisessä mallissa käytettiin kaikki kontekstin sanat estimoinnissa.
- Nyt päinvastainen lähestymistapa: valitaan yksittäinen mahdollisimman hyvä indikaattori jonka arvo voidaan selvittää kustakin kontekstista kullekin merkitykselle.
- **Esimerkki:** Ranskan 'prendre'-sanan merkitykset 'tehdä päätös' (*prendr un decision* ja 'ottaa mitta' (*prendre une mesure*, luotettava indikaattori olisi verbin objektina oleva sana.
- Disambiguoitava sana: $w = prendre$
  Valitaan piirrejoukoksi (indkiaattoriksi) esim. objektipositiossa olevat sanat $V = \{$*mesure, note, exemple, décision, parole*$\}$
  Käännösten joukko: $K = \{take, make, rise, speak\}$

Informaatioteoreettinen lähestymistapa alkaa:

Maksimoidaan yhteisinformaatio piirteen ja merkityksen välillä. Muistellaan yhteisinformaation kaavaa

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

partitiointi = jonkin joukon jako osajoukkoihin.

Flip-Flop-algoritmi

- K= käännösten joukko, V= piirteiden joukko
- Valitse V:lle satunnaisesti jokin partitiointi Y
- Toista kunnes parannus ei enää suuri:
    1. Etsi K:lle partitiointi X siten että $I(X;Y)$ maksimoituu

2. Etsi V:lle partitiointi Y siten että $I(X;Y)$ maksimoituu

Esim. jos partitioidaan vain kahteen ryhmään:
Käännettävä sana: prendre
Mahdolliset käännökset $K = \{take, rise, make, speak\}$
Objektiposition piirteet $V = \{$*mesure, note, exemple, décision, parole*$\}$

Data:
'prendre un mesure ': take measure,
'prendre .. note': take notes,
'prendre ... exemple': take an example
'prendre un decision': 'make...decision'
'prendre la parole': 'rise to speak'

Paras partitiointi:
$X_1 = \{take\}$, $X_2 = \{rise, make, speak\}$
$Y_1 = \{mesure, note, exemple\}, Y_2 = \{decision, parole\}$

- Jos tehdään täyshaku eli kokeillaan kaikki partitioinnit kummallekin joukolle, K ja V, menee exponentiaalinen aika
- Flip-Flop-algoritmi kuitenkin vie lineaarisen ajan
- Toistetaan Flip-Flop-algoritmi eri indikaattorelle (objekti, määre, edellinen sana, jne), valitaan indikaattori joka maksimoi yhteisinformaation
- Brown et al: Parhaat indikaattorit eri sanoille olivat:

Disambiguoitava sana	Paras indikaattori
prendre	akkusatiiviobjekti
vouloir	sanan aikamuoto
cent	edellinen sana

- Brown et.al: 20% parannus konekäännösjärjestelmässä lausetasolla.

Huom. Oikeasti käännöksistä saadut 'labelit' eivät välttämättä sanan eri merkityksiä, vaan osaksi saman merkityksen eri ilmenemismuotoja (ks. sananmuotojen variaatio, esim 'cent' viitatessaan merkitykseen senttimetri voisi

ilmetä suomeksi muodoissa ('cm','sentti'). Partitioidessaan myös eri labelien joukon S algoritmi itse asiassa ryhmittelee nämä eri muodot pienemmäksi joukoksi merkityksiä.

Kääntämisen kannalta katsottuna tehtävä on siis ohjatun oppimisen tehtävä. Merkitysten etsimisen (esim. niiden oikea lukumäärä) ja disambiguoinnin kannalta taas ohjaamatonta oppimista.

7.4 Sanakirjapohjainen disambiguointi

Lesk:in sanakirjamerkitysmäärittelyihin perustuva menetelmä

Notaatio ja piirteet:

sanan kuvaus	symboli	eri merkitykset	niiden määritelmät
tulkittava sana	w	$s_1 \dots s_K$	$D_1 \dots D_K$
w:n kontekstin sana j	v_j	$s_{j_1} \dots s_{j_L}$	$D_{j_1} \dots D_{j_L}$

Piirrejoukko: $E_{v_j} = \bigcup_{j_i} D_{j_i}$ eli ei välitetä kontekstin sanojen monimerkityksisyyksistä (yhdistetään määritelmät sanan w piirteitä laskettaessa).

Valintakriteeri:

$$s' = \arg\max_{s_k} \bigcap (D_k, E_{v_j})$$

Huom: Tämä on matemaattisesti täsmälleen sama kuin vektoriavaruusmenetelmä jossa käytetään binääriarvoisia, normalisoimattomia vektoreita. Jos kerran sovelletaan vektoriavaruusmallia, kannattaisi harkita myös kehitty-

neempiä samankaltaisuusmittoja (piirteiden painotus ja vektorien pituuksien normalisointi).

Sanojen semanttisiin aiheluokkiin perustuva menetelmä

(kirjassa nimellä 'thesaurus-based disambiguation')

- Pohjana yleinen semanttinen luokitus (thesauruksessa, mm. Roget, tai muuten, mm. Longman)
- Luokat t aihealueita (topics) esim. {'urheilu', 'sota', 'musiikki', 'kalastaminen',...}.
- $$\text{score}(s_k) = \sum_{v_j \text{ in } c} \delta(t(s_k), v_j)$$
- $\delta(t(s_k), v_j) = 1$ joss $t(s_k)$ on jokin v_j:n luokista
- Olet. että kukin merkitys s_k kuuluu täsmälleen yhteen luokista
- Sanan luokkien joukko on sen eri merkitysten luokkien unioni ts. ei yritä disambiguoida kontekstisanojen merkityksiä

- score$=$ monellako kontekstin sanalla löytyy yhteinen luokka sananmerkityksen s_k kanssa, ts. tässäkään ei käytetä normalisointeja tai painotuksia tai todennäköisyyksiä

- Jättää hyödyntämättä sanat joita ei etukäteen sem.luokiteltu (esim. uudet sanat, jonain aikakautena kuuluisat henkilönnimet 'Navratilova', 'Jeltsin' jne. jotka voisivat olla oikein hyviäkin piirteitä)

Yarowsky:n adaptiivinen versio edellisestä

- Lähtee liikkeelle annetusta sem. luokittelusta ja parantaa sitä luokittelemalla myös uudet sanat niiden kontekstien luokkatiedon perusteella (bootstrapping)
- Yarowskyn kokeissa konteksti = 100 sanan ikkuna
- Soveltaa Naive Bayes-oletusta, eli olet. piirteiden vaikutus riippumattomiksi:

$$\begin{aligned} P(t_l|c_i) &= \frac{P(c_i|t_l)}{P(c_i)} P(t_l) \\ &= \frac{\prod_{v \text{ in } c_i} P(v|t_l)}{\prod_{v \text{ in } c_i} P(v)} P(t_l) \end{aligned}$$

- Käytetään suurehkoa kynnysarvoa α hyväksymään luokka vain jos se on riittävän todennäköinen tälle kontekstille (eli jos $P(t_l|c_i) > \alpha,$ suuri α).

Yarowsky:n algoritmi:

1. **Luokittele kontekstit** sanojen luokkien perusteella
   päivitä $P(t_l|c_i)$ kaikille c_i
   päivitä $t(c_i) = \{c_i : n \text{ luokat joille } P > \alpha\}$

2. **Luokittele sanat** kontekstien luokkien perusteella
   $V_j = \{\text{kontekstit joissa piirre } v_j\}$
   $T_l = \{\text{kontekstit joilla luokkana } t_l\}$
   päivitä $P(v_j|t_l) = \frac{|V_j \bigcap T_l|}{\sum_j |V_j \bigcap T_l|}$
   päivitä $P(t_l) = \frac{|V_j \bigcap T_l|}{\sum_l \sum_j |V_j \bigcap T_l|}$

3. **Disambiguoi** moniselitteinen sana w

$$\begin{aligned} s' &= \arg\max_{s_k} \prod_{v_j \text{ in } c} P(t(s_k), v_j) \\ &= \arg\max_{s_k} \prod_{v_j \text{ in } c} P(t(s_k)) P(v_j | t(s_k)) \\ &= \arg\max_{s_k} [\log P(t(s_k)) + \sum_{v_j \text{ in } c} \log P(v_j | t(s_k))] \end{aligned}$$

- uusille sanoille: estimoidaan luokat
- vanhoille sanoille: päivitetään luokat
- salliiko algoritmi vanhan sanan luokan poistumisen? (kirjassa ei kerrota...)
- Muuten toimitaan todennäköisyksillä, mutta kontekstien ja piirteiden luokitus on 'kova'.
- Menetelmällä saantu hyviä tuloksia testidatalla

- topic-lähestymistapa toimii hyvin silloin kun merkitys aihealuekohtainen. Kuitenkin kaikilla sanoilla merkitykset eivät riipu aihealueista → huonoja tuloksia.

2-kielisen aineiston käännöksiä hyödyntävä menetelmä

- Tarvitaan: 2-kielinen sanakirja + toisen kielen korpus
- Käännetään sana kaikilla eri tavoilla (kaikilla eri merkityksillä)
- käännetään sanan kontekstipiirre (yksinään, olet. vain yksi käännös)
- Tarkastellaan käännösparien keskinäisiä frekvenssejä toisen kielen korpuksessa. Mikäli jokin vaihtoehto on riittävän todennäköinen, valitaan sen sisältämä tulkinta (testataan merkitsevyys, ja valitaan vain mikäli $p >$ esim. 90%).
- Yleistyy suoraviivaisesti monen piirteen tutkimiselle

Yksi merkitys per aihe, yksi merkitys per kollokaatio

- Sanakirjapohjaiset menetelmät tarkastelivat jokaista sananesiintymää erillisenä
- Kuitenkin esiintymien välillä keskinäisiä riippuvuuksia
- Oletus 1: Yksi merkitys per aihe: Sanalla yleensä yksi merkitys läpi koko dokumentin
- Oletus 2: Yksi merkitys per kollokaatio: sanan merkitys riippuu vahvasti aivan lähikontekstin sanoista, mukaanlukien sanojen järjestys ja sanaluokkatieto (ts. usein toisistaan täysin erilliset merkitykset ovat eri syntaktisessa ja/tai semanttisessa roolissa ympäristön suhteen)

Edellisen mallin ongelma: NB:n tekemä piirteiden riippumattomuusoletus. Vaihdetaan siksi mallia: valitaan 1 'paras' piirre ja käytetään vain sen mukanaantuoma evidenssi. (Kolmas vaihtoehto olisi mallittaa oikeasti piirteiden yhteisvaikutus)

Yarowsky:n (toinen) algoritmi:

1. Sovella oletusta 1: Sovella sanakirjamenetelmää tai bootstrappaystä merkitysten valintaan, mutta nyt vain 'parasta' diskriminoivaa kontekstin piirrettä käyttäen
2. Sovella oletusta 2: 'Äänestetään' sanan merkitys dokumentin sisällä kaikille samaksi

Yarowsky: 2-vaihe parantaa tuloksia 27%. Lopulliset tulokset luokkaa 90%-96%.

7.5 Ohjaamaton merkitysten ryhmittely

- Edelläkuvatut menetelmät tarvitsevat leksikaalisia resursseja: sanakirjoja tai (pieniä) merkityksin tagattuja aineistoja jokaiselle monimerkityksisen sanan eri merkitykselle
- Aina sellaisia ei ole, esim. erikoistermien tai uusien merkitysten ilmaantuessa.
- Jos disambiguoinnilla tarkoitetaan merkitysten taggausta, täysin ohjaamattomasti ei voida disambiguoida.
- Voidaan kuitenkin klusteroida sanan esiintymät, ja toivoa/olettaa että kukin klusteri vastaa sanan yhtä merkitystä.
- Hyvä tai huono puoli: ryhmittely voi olla tarkempaa kuin esim. sanakirjoissa.
- Menetelmiä esim. EM-algoritmi, k-means, SOM, hierarkkiset klusterointimenetelmät, ...

EM-algoritmi lyhyesti

Mallissa kaksi osaa: datan generointitn:t kustakin luokasta ja kunkin luokan jakauman parametrit. Alustetaan parametrit satunnaisesti. Vuorotellaan estimoinnissa kahta askelta:

1. E-askel: Lasketaan datan generointi luokista ja datan log-likelihoodin arvo nyk. parametrien arvoilla
   $\log P(data|\theta, \mathcal{H})$
   (vrt. lasketaan havaintojen luokittelu; pidetään luokkien jakaumien parametrit samoina)

2. M-askel: Lasketaan malliin uudet parametrit siten että ne maksimoivat datan log-likelihoodin (nyk. ennustuksilla)
   (vrt. lasketaan luokkien jakauman parametrit; pidetään havaintojen luokittelu samana)

Sovellus disambiguointiin

Log-likelihoodin laskeminen:
$L(data|\theta, \mathcal{H}) = \log \prod_i \sum_k P(c_i|s_k)P(s_k) = \sum_i \log \sum_k P(c_i|s_k)P(s_k)$

Valitaan malliksi Naive Bayes: $P(c_i|s_k) = \prod P(v_j|s_k)$

Niin kauan kuin $L(\ldots)$ kasvaa (selvästi), toista:

1. E-askel: Lasketaan kontekstien posteriori-generointitn:t kustakin luokasta
   $h_{ik} = \frac{p(c_i|s_k}{\sum_k P(c_i|s_k)}$
   + evaluoidaan L
2. M-askel: Estimoidaan parametrit $P(v_j|s_k)$ ja $P(s_k)$

Huomioita EM-algoritmista

- On herkkä alustukselle.
- L paranee joka kierroksella
- Mitä suurempi määrä luokkia (merkityksiä), sen parempi L. Ongelmana siis ylioppiminen (jos soveltaa sellaisenaan eri luokkamäärien vertailuun).
- Kirjassa esitetään ratkaisuksi "kustannuksen pienenemisen tarkkailu" — käytännön jippo jossa jälleen säädettäviä parametreja.
- Parempi ratkaisu ylioppimisen välttämiseksi: ML-estimoinnin sijaan Bayesiläinen variaatioanalyysi, joka huomioi mallin koon kasvun kustannusfunktiossa. (vrt. MDL)

Paluu lähtökuoppiin

Joissain tilanteissa OK epäonnistua.

Esim. sanaleikit, 'In AI, much of the I is in the beholder'
Vrt. 'Beauty is in the eye of the beholder' (kauneus on katsojan silmässä)

Hypoteesi (Kilgariff): On tavallista että useat merkityksistä yhtäaikaa läsnä:

'For better or for worse, this would bring competition to the licenced trade'
competition - competitors vs. competition - the act of competing

Mahdollinen selitys: ihmiset eivät disambiguoi sanoja, vaan tulkitsevat lauseita ja tekstejä. Jos kaksi sananmerkitystä johtavat samaan lauseen tulkintaan, sananmerkityksiä ei ole tarpeen eritellä.

Disambiguointiongelmia esim.

- systemaattinen polysemia (tekemisen akti vs. osallistujat/yhteisö)
- pisteen merkitys (lauseen loppu vs. muu)
- erisnimi vai yleisnimi 'Brown', 'Bush', 'Army'
- etu- vai sukunimi 'Pentti Jaakko'

Joitain loppuhuomioita menetelmistä

Muita ohjatun oppimisen menetelmiä: kNN

- valitaan k lähintä luokiteltua esimerkkiä, ja luokitellaan tämä esimerkki enemmistöäänestyksellä.
- Sopii hyvin harvalle datalle
- Edellyttää ainoastaan 'samankaltaisuuden' määrittelyn + mittauksen lähimpiin samankaltaisiin (kompleksisuus O(Nd) jossa N datan määrä ja d dimensio)

Muita ohjaamattoman oppimisen menetelmiä

Klusterointimenetelmiä:
K-means (Schutze soveltanut merkitysten ryhmittelyyn), SOM, hierarkkiset klusterointimenetelmät, erilaiset samankaltaisuusmitat

Menetelmien vertailua

- Senseval-projekti: Laaja WSD-menetelmien vertailu yhteisellä datalla
- aineisto + evaluoinnit webissä

8. N-grammi-kielimallit

8.1 Tilastollinen mallinnus

1. Otetaan dataa (generoitu tuntemattomasta tn-jakaumasta)
2. Tehdään estimaatti jakaumasta datan perusteella
3. Tehdään päätelmiä uudesta datasta jakaumaestimaatin perusteella

Mallinnuksen osatehtävät voidaan hahmottaa seuraavasti:

- Datan jakaminen ekvivalenssiluokkiin
- Hyvän tilastollisen estimaattorin löytäminen kullekin luokalle
- Useiden estimaattorien yhdistäminen

Tyypillinen oletus: **stationaarisuus**, eli että datan tn-jakauma ei muutu oleellisesti ajan myötä.

Tilastollisen kielimallin tehtävistä

Klassinen tehtävä: seuraavan sanan (tai kirjaimen) ennustaminen jo nähtyjen sanojen (tai kirjainten) perusteella ('Shannon game'). Esim. seuraavissa sovelluksissa:

- puheentunnistus
- optinen merkkientunnistus, käsinkirjoitettujen merkkien tunnistus
- kirjoitusvirheiden korjaus
- tilastollinen konekääntäminen

Estimointimenetelmät yleisiä, soveltuvat myös muihin tehtäviin (esim. WSD, jäsentäminen)

8.2 N-grammimallit

N-grammimalli: ennustetaan sanaa w_n edellisten $n-1$ sanan perusteella:

$$P(w_n|w_1 w_2 \cdots w_{n-1}) \tag{32}$$

Kaava esiintyy myös muodossa $P(w_t|w_{t-(n-1)} w_{t-(n-2)} \cdots w_{t-1})$ jossa t viittaa sanan järjestysnumeroon (ajanhetkeen) koko aineistossa.

Esimerkki: aineistona tämän luennon kalvot, n=4:

		w_{t-3}	w_{t-2}	w_{t-1}	w_t		
...	sitä	enemmän	dataa	tarvitaan	**mallin**	estimointiin	...

Malleille käytettäviä nimiä

n=1	unigram
n=2	bigram
n=3	trigram
n=4	4-gram, fourgram

Jos nimet tulisivat kreikasta, p.o. mono-, di-, tri-, quadri-, ...

Yhteys ekvivalenssiluokkiin: n-grammimallissa jokainen $n-1$:n sanan pituinen historia saa oman ekvivalenssiluokkansa. Tämä tarkoittaa että historiat joissa viimeiset 3 sanaa samoja käsitellään keskenään identtisinä tilanteina seuraavan sanan ennustamisen kannalta, eli niillä on yhteinen estimaatti.

Sama n-grammien ominaisuus toisesta näkökulmasta: malli olettaa että sana riippuu ainoastaan $(n-1)$ edeltävästä sanasta, mutta ei tätä kauempana olevista sanoista (ns. Markov-oletus).

Markov-malli: k:n asteen Markov-malli on malli joka asettaa kaikki k:n pituiset historiat samaan ekvivalenssiluokkaan. Ts. n-grammimalli on $n-1$:n asteen Markov-malli.

Esimerkkejä:
Sue swallowed the large green ____

Samppa Lajunen voitti kultaa ____

Parametrien määrän kasvu

	Malli	Parametreja jos sanasto 20,000
n=1	unigram	20000
n=2	bigram	$20000^2 = 400$ milj.
n=3	trigram	$20000^3 = 8$ miljardia
n=4	4-gram, fourgram	1.6×10^{17}

8.3 Piirteiden jakaminen ekvivalenssiluokkiin

- Piirteet (sekä jatkuva-arvoiset että diskreetit) voidaan jakaa ekvivalenssiluokkiin 'bins'
- Esim. jatkuva-arvoisen muuttujan 'ikä' jakaminen luokkiin 0-2; 3-5; 7-10; 11-15; 16-25; 26-35 jne
- Mitä useampia ekv.luokkia, sitä enemmän dataa tarvitaan mallin estimointiin, jotta tulokset *luotettavia* kullekin luokalle
- Toisaalta, jos luokkia on kovin vähän, ennustettavan kohdemuuttujan (esim. 'pituus') arvoa ei voida ennustaa kovin *tarkasti*.

Esimerkki: ennustetaan seuraavaa sanaa
1. kolmen edellisen sanan sanaluokan (subst, verbi, adj, num jne) TAI
2. kolmen edellisen sanan perusteella

1. tapauksessa vähemmälläkin datalla jonkinlaiset estimaatit, kun taas
2. tapauksessa tarkempia estimaatteja mutta dataa tarvitaan paljon enemmän.

Joitain tapoja muodostaa ekvivalenssiluokkia

- Isojen ja pienten kirjainten käsittely samalla tavalla (esim. kaiken muuntaminen pieniksi kirjaimiksi)
- Sanojen muuntaminen perusmuotoon (saman sanan eri taivutusmuodot käsitellään ekvivalentteina)
- Ryhmittely sanaluokkatiedon mukaan (syntaktiselta rooliltaan samankaltaiset muodostavat ekv. luokan)
- Sanojen semanttinen ryhmittely (merkitykseltään samankaltaiset muodostavat ekv.luokan)

Kussakin vaihtoehdossa tarvitaan kuitenkin menetelmä jolla sanan ekvivalenssiluokka voidaan luotettavasti päätellä.

Lisäksi ekvivalenssiluokkien olisi hyvä olla sellaisia että niiden sisällä sanat todella käyttäytyvät samankaltaisesti, ts. tarkkuus säilytetään.

Historian huomioimisen eri tapoja

Edellä kuvattiin yksittäisten piirteiden ekvivalenssiluokkien laskemista. Eri tapoja ekvivalenssiluokkien muodostamiseen historian suhteen:

- Poimitaan historiasta tiettyjä piirteitä, mutta niiden sijainnilla ei ole väliä esim. malli: $P(w_t|\text{lauseen predikaatti}, w_{t-1})$
- Käsitellään sanajonon sijaan sanajoukkoa ('sanasäkkiä', bag-of-words), eli ei välitetä sanojen järjestyksestä:
  $P(w_n|w_1, w_2, \ldots, w_{n-1})$

8.4 N-grammimallin tilastollinen estimointi

Annettuna: joukko näytteitä jotka osuvat kuhunkin ekvivalenssiluokkaan (biniin). Bayesin kaavoista:

$$P(w_n|w_1 \cdots w_{n-1}) = \frac{P(w_1 \cdots w_n)}{P(w_1 \cdots w_{n-1})} \tag{33}$$

Mallin optimointi: maksimoidaan datan todennäköisyys (eli sanojen tn:ien tulo).

Notaatio:

N	Opetusnäytteiden lukumäärä
B	Ekv.luokkien (binien) lukumäärä
w_{1n}	n-grammi $w_1 \cdots w_n$
$C(w_1 \cdots w_n)$	ngrammin $w_1 \cdots w_n$ lukumäärä opetusdatassa
r	n-grammin lukumäärä
N_r	Niiden binien lukumäärä joissa on r näytettä
h	historia (edeltävä sanajono)

Maximum likelihood-estimaatti (MLE)

$$P_{\mathsf{MLE}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n)}{N} \tag{34}$$

$$P_{\mathsf{MLE}}(w_n | w_1 \cdots w_{n-1}) = \frac{C(w_1 \cdots w_n)}{C(w_1 \cdots w_{n-1})} \tag{35}$$

- MLE-estimointi johtaa parametrien valintaan siten että opetusdatan todennäköisyys maksimoituu.
  (Huom: tämä pätee vain tietyin oletuksin, kuten että näytteet, esim. tri-grammien sanakolmikot, oletetaan riippumattomiksi toisistaan. Tämä taas ei pidä paikkaansa mm. overlapin takia.)
- Koko tn-massa jaetaan opetusdatassa esiintyneiden tapausten kesken, niiden frekvenssien suhteessa.
- Antaa siis tn=0 tapaukselle jota ei nähty opetusdatassa, eli ei jätä lainkaan tn-massaa aiemmin näkemättömille sanoille.

- Koska yleisesti sanajonon tn lasketaan kertomalla kunkin sanan tn, yksikin nolla saa koko sanajonon tn:n nollaksi.
- Esimerkki datan harvuudesta: ensimmäisten 1.5 miljoonan sanan jälkeen (IBM laser patent text corpus) 23% myöhemmistä trigrammeista oli ennennäkemättömiä.
- MLE ei kovin hyödyllinen estimaatti harvalle datalle, kuten n-grammeille.
- Tarvitaan siis systemaattinen tapa jolla huomioidaan ennennäkemättömi sanojen ja ennennäkemättömien n-grammien tn:t. Tätä kutsutaan mm. nimellä *tasoitus* eli *smoothing*

Taulukko 6.3: MLE-estimaatteja Austenin kirjoista eräälle lauseelle eri n-grammeilla.

In person	*she*		*was*		*inferior*		*to*	
1-gram	$P(\cdot)$		$P(\cdot)$		$P(\cdot)$		$P(\cdot)$	
1	the	0.034	the	0.034	the	0.034	the	0.034
2	to	0.032	to	0.032	to	0.032	**to**	**0.032**
3	and	0.030	and	0.030	and	0.030		
4	of	0.029	of	0.029	of	0.029		
...								
8	was	0.015	**was**	**0.015**	was	0.015		
...								
13	**she**	**0.011**			she	0.011		
...								
254					both	0.0005		
...								
435					sisters	0.0003		
...								
1701					**inferior**	**0.00005**		
2-gram	$P(\cdot \mid person)$		$P(\cdot \mid she)$		$P(\cdot \mid was)$		$P(\cdot \mid inferior)$	
1	and	0.099	had	0.141	not	0.065	**to**	**0.212**
2	who	0.099	**was**	**0.122**	a	0.052		
3	to	0.076			the	0.033		
4	in	0.045			to	0.031		
...								
23	**she**	**0.009**						
...								

Laplacen laki eli 'yhden lisäys'

Annetaan hiukan tn-massaa näkemättömille tapauksille lisäämällä jokaiseen lukuun 1:

$$P_{\mathsf{LAP}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n) + 1}{N + B} \tag{36}$$

- Vastaa Bayesin estimaattia priorilla että kaikki tapahtumat ovat yhtä todennäköisiä, ja tähän prioriin uskotaan aivan kuin olisi nähty yksi näyte joka lajia.
- Esim. 44 milj. sanan AP newswire-korpus, sanaston koko 400,653 sanaa, jolloin bigrammeja 1.6×10^{11}, eli $N =$44 milj., $B = 1.6 \times 10^{11}$
- Jos data on hyvin harvaa, antaa liiaksi tn-massaa ennen näkemättömille tapauksille (tässä 46.5% tn-massasta).
- Ts. uskotaan tasajakauma-prioriin liian vahvasti verrattuna datan määrää
- Kannattaisiko 1:n sijaan uskoa että ollaan nähty esim. 0.0001 jokaista näytettä?

Odotetun frekvenssin estimaatteja seuraavassa taulukossa:

$r = f_{MLE}$	$f_{empirical}$	f_{Lap}	f_{del}	f_{GT}	N_r	T_r
0	0.000027	0.000137	0.000037	0.000027	74 671 100 000	2 019 187
1	0.448	0.000274	0.396	0.446	2 018 046	903 206
2	1.25	0.000411	1.24	1.26	449 721	564 153
3	2.24	0.000548	2.23	2.24	188 933	424 015
4	3.23	0.000685	3.22	3.24	105 668	341 099
5	4.21	0.000822	4.22	4.22	68 379	287 776
6	5.23	0.000959	5.20	5.19	48 190	251 951
7	6.21	0.00109	6.21	6.21	35 709	221 693
8	7.21	0.00123	7.18	7.24	27 710	199 779
9	8.26	0.00137	8.18	8.25	22 280	183 971

Table 6.4 Estimated frequencies for the AP data from Church and Gale (1991a). The first five columns show the estimated frequency calculated for a bigram that actually appeared r times in the training data according to different estimators: r is the maximum likelihood estimate, $f_{empirical}$ uses validation on the test set, f_{Lap} is the 'add one' method, f_{del} is deleted interpolation (two-way cross validation, using the training data), and f_{GT} is the Good-Turing estimate. The last two columns give the frequencies of frequencies and how often bigrams of a certain frequency occurred in further text.

Lidstonen laki, Jeffreys-Perksin laki

$$P_{\text{Lid}}(w_1 \cdots w_n) = \frac{C(w_1 \cdots w_n) + \lambda}{N + B\lambda} \tag{37}$$

Voidaan osoittaa että ylläoleva tarkoittaa lineaarista interpolointia tasajakauma priorin ja MLE-estimaatin välillä. Asetetaan $\mu = N/(N + B\lambda)$:

$$P_{\text{Lid}}(w_1 \cdots w_n) = \mu \frac{C(w_1 \cdots w_n)}{N} + (1 - \mu)\frac{1}{B} \tag{38}$$

- Jeffreysin priori: $\lambda = 1/2$, eli lisätään jokaiseen lukumäärään $1/2$ (vastaa sitä että olisi nähty puolikas näyte jokaista lajia). Käytetään myös nimeä *Expected Likelihood Estimation* (ELE)
- On valittava λ:n arvo tavalla tai toisella
- Alhaisilla frekvensseillä tämäkään ei kovin hyvin vastaa todellista jakaumaa

Good-Turing-estimaattori

Ks. frekvenssien frekvenssi-histogrammeja taulukossa 6.7.

$$P_{GT}(w_1 \cdots w_n) = \frac{r^*}{N}, \text{ jossa } r^* = \frac{(r+1)S(r+1)}{S(r)} \tag{39}$$

ja $S(r)$ on odotusarvo N_r:lle, tai vaihtoehtoisesti, arvo joka on saatu sovittamalla jokin tasainen käyrä frekvenssien frekvensseille: $N_r = S(r)$

Simple Good-Turing-estimaattori: Valitaan käyräksi potenssifunktio: $S(r) = ar^b$ jossa parametrit a ja b sovitetaan frekvenssien frekvenssi-histogrammin mukaan.

Melko hyvä estimaattori, yleisesti käytössä.

	Bigrams				Trigrams		
r	N_r	r	N_r	r	N_r	r	N_r
1	138741	28	90	1	404211	28	35
2	25413	29	120	2	32514	29	32
3	10531	30	86	3	10056	30	25
4	5997	31	98	4	4780	31	18
5	3565	32	99	5	2491	32	19
6	2486		...	6	1571		...
7	1754	1264	1	7	1088	189	1
8	1342	1366	1	8	749	202	1
9	1106	1917	1	9	582	214	1
10	896	2233	1	10	432	366	1
	...	2507	1		...	378	1

Table 6.7 Extracts from the frequencies of frequencies distribution for bigrams and trigrams in the Austen corpus.

Muita tasoitusmenetelmiä

Termi 'discounting' viittaa siihen että nähtyjen n-grammien tn:iä alennetaan ja tätä massaa jaetaan ennen näkemättömille.

- Absoluuttinen alennus (absolute discounting): Kaikkista nähdyistä n-grammeista vähennetään vakio-tnmassa σ joka jaetaan tasan näkemättö n-grammien kesken.
- Lineaarinen alennus (linear discounting): Skaalataan nähtyjen n-grammie tn:ät vakiolla joka on hiukan pienempi kuin 1, ja saatu tnmassa jaetaan tasan ei-nähtyjen kesken. Ei kovin hyvä, koska 'rankaisee' frekventtejä enemmän—kuitenkin niiden estimaatit ovat parempia.
- Witten-Bell-discounting: Arvioidaan yllättävien asioiden näkemisen tn-massa sen perusteella kuinka tavallista yllättävien asioiden näkeminen on ollut tähän mennessä: $\sum_{i:C(i)=0} p_i = \frac{T}{N+T}$ jossa T on tähän mennessä nähtyjen binien määrä.
- Natural Law of Succession-discounting (Ristad, 1995)

Pohjimmiltaan menetelmien eroissa on kyse on siitä minkälaisia oletuksia tehdään tapauksista joita ei ole nähty ja niiden suhteesta tapauksiin joita on nähty.

Huom: Esim. CMU Statistical Language Toolkit toteuttaa useita eri discounting ja smoothausmenetelmiä n-grammeille.

8.5 Estimaattorien yhdistäminen

- Tähän asti tarkasteltu tilannetta jossa pyritään estimoimaan identtinen tn esim. kaikille 3-grammeille joita ei ole nähty.
- Kuitenkin jos 3-grammin osat (esim. 2-grammit) ovat frekventtejä, eikö niistä kerättyä tietoa kannattaisi käyttää 3-grammin tn:n estimoinnissa?
- Motivaationa estimaattien tasoitus (smoothing) tai yleisemmin eri informaationlähteiden yhdistäminen.

Lineaarinen interpolointi

(yleisemmin nimellä äärelliset mikstuurimallit tai sum of experts)

Lasketaan painotettu keskiarvo eri pituisten kontekstien antamista estimaateista:

$$P_{li}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n|w_{n-1}) + \lambda_3 P_3(w_n|w_{n-2}w_{n-1}) \tag{40}$$

($0 \leq \lambda_1 \leq 1$ ja $\sum_i \lambda_1 = 1$)

Parametrit λ voidaan asettaa käsin tai optimoida datan avulla.

Yleinen lineaarinen interpolointi

Edellä parametrit λ eivät riippuneet sanoista joiden kohdalla niitä sovelletaan, eli parametri on vakio vaikkapa kaikille bi-grammeille.

Yleisemmin ne voidaan kuitenkin asettaa riippumaan historiasta:

$$P_{li}(w|h) = \sum_i \lambda_i(h) P_i(w|h) \tag{41}$$

($0 \leq \lambda_1 \leq 1$ ja $\sum_i \lambda_1 = 1$) ja optimoida esim. EM-algoritmilla. Kuitenkin, jos jokaiselle historialle on oma λ ollaan taas datan harvuusongelmassa, ja joudutaan soveltamaan jotain smoothausta, historioiden ekvivalenssiluokkia tms.

Perääntyminen (backing off)

- Periaate: Katsotaan aina spesifeintä mallia joka antaa 'riittävän luotettavaa' informaatiota tämänhetkisestä kontekstista.
- Eli peräännytään pitkien kontekstien käytöstä yhä lyhempiin: Päätetään uskoa estimaattia jos se perustuu vähintään k näytteeseen (k esim. 1 tai 2)
- Kritiikkiä: Uuden opetusdatan lisääminen voi vaikuttaa voimakkaasti tn:iin kun se aiheuttaa muutoksia useiden sanojen kohdalla niille sovellettavissa n-grammipituuksissa
- Kuitenkin mallit yksinkertaisia ja toimivat melko hyvin, joten yleisesti käytössä.
- back-off -malli on erikoistapaus yleisestä lineaarisesta interpoloinnista: $\lambda_i(h) = 1$ kun k:n arvo riittävän suuri, 0 muulloin.

Back-off-mallien käyttöesimerkki:

	$P(she\|h)$	$P(was\|h)$	$P(inferior\|h)$	$P(to\|h)$	$P(both\|h)$	$P(sisters\|h)$
Unigram	0.011	0.015	0.00005	0.032	0.0005	0.0003
Bigram	0.00529	0.1219	0.0000159	0.183	0.000449	0.00372
n used	2	2	1	2	2	2
Trigram	0.00529	0.0741	0.0000162	0.183	0.000384	0.00323
n used	2	3	1	2	2	2

Table 6.11 Probability estimates of the test clause according t
guage models. The unigram estimate is our previous MLE unigram
other two estimates are back-off language models. The last col
overall probability estimate given to the clause by the model.

8.6 Mallien estimoinnista yleisesti

Seuraava koskee mitä tahansa menetelmien vertailua, ei pelkästään n-grammej tai kielimalleja.

Held-out estimation

Tavallisesti data jaetaan ennen menetelmien kehittämistä kolmeen osaan

- **Opetusjoukko:** data jolla malli opetetaan
- **Validointijoukko:** opetusjoukosta riippumaton data, jonka avulla valitaan mallin opetuksessa käytettävät parametrit (esim. edellisen kalvon λ)
- **Testijoukko:** edellisistä riippumaton, satunnaisesti valittu datajoukko (kooltaan esim. 10% opetusdatasta), jolla lopullisen mallin hyvyys mitataan.

Testijoukko on pidettävä kokonaan syrjässä menetelmien kehittämisen ai-

kana! Jos testijoukko pääsee vaikuttamaan menetelmänkehitykseen (vaikka vain alitajuisesti), se ei ole enää soveltuva menetelmän testaamiseen.

Kuitenkin usein menetelmänkehitys on syklinen prosessi jossa välillä muutetaan menetelmää ja sitten taas testataan. Siksi voi olla erikseen:
1. **kehittely-testijoukko**, jolla vertaillaan menetelmän eri variantteja
2. **lopullinen testijoukko** jolla tuotetaan julkaistavat tulokset, ja jota ei ole käytetty mihinkään ennen tätä.

Vaihtoehdot testijoukon (ja validointijoukon) valintaan:
1. täysin satunnainen valinta (satunnaisia lyhyitä tekstinpätkiä)
2. pitkiä yhtenäisiä pätkiä (esim. ajallisesti myöhempiä osia datasta)

2-tapa vastaa paremmin mallin käyttötilannetta: se myös antaa realistisemmat, yleensä hiukan huonommat tulokset johtuen siitä että harvat ilmiöt ovat täysin stationaarisia.

Eri menetelmien vertailusta

Pelkkiä keskiarvo-tuloksia vertaamalla ei voi tietää ovatko havaitut erot menetelmissä merkitseviä.

Eräs ratkaisu: Mitataan lisäksi tulosten varianssi eri datajoukoilla, ja testataan erojen tilastollinen merkitsevyys esim. t-testillä.

	System 1	System 2
scores	71, 61, 55, 60, 68, 49, 42, 72, 76, 55, 64	42, 55, 75, 45, 54, 51 55, 36, 58, 55, 67
total	609	526
n	11	11
mean $\bar{x}_i$	55.4	47.8
$s_i^2 = \sum(x_{ij} - \bar{x}_i)^2$	1,375.4	1,228.8
df	10	10

$$\text{Pooled } s^2 = \frac{1375.4+1228.8}{10+10} \approx 130.2$$

$$t = \frac{\bar{x}_1-\bar{x}_2}{\sqrt{\frac{2s^2}{n}}} = \frac{55.4-47.8}{\sqrt{\frac{2\cdot 130.2}{11}}} \approx 1.56$$

Table 6.6 Using the t test for comparing the performance of two systems. Since we calculate the mean for each data set, the denominator in the calculation of variance and the number of degrees of freedom is $(11 - 1) + (11 - 1) = 20$. The data do not provide clear support for the superiority of system 1. Despite the clear difference in mean scores, the sample variance is too high to draw any definitive conclusions.

Ristiinvalidointi (cross-validation)

- Jaetaan data K:hon osajoukkoon, joista 1 kerrallaan on testidata, muut opetusdataa. Toistetaan siten että kukin osajoukko on vuorollaan testidata. K välillä 2. . . N, jossa N datan määrä.
- Hyöty: Kaikki datat vaikuttavat sekä mallin opetukseen että sen testaukseen, data siis hyödynnetään mahdollisimman tarkasti (tärkeää etenkin kun dataa on vähän).
- useita eri variantteja (deleted estimation, leave-one-out-estimation)

Sekä ristiinvalidoinnin että held-out-estimoinnin avulla voidaan valita mallien parametrejä, ja siis esim. tasoittaa tn-estimaatteja.

8.7 N-grammimallin kritiikkiä

N-grammien ongelmia kielimallina:

- Eivät huomioi pidemmän tähtäimen riippuvuuksia sanojen välillä
- Sanajono yhdessä järjestyksessä ei kontribuoi saman sanajoukon tn:ään jossain toisessa järjestyksessä
- Smoothaus-ongelmat voi myös nähdä mallin rakenteellisena ongelmana
- Riippuvuudet estimoidaan sanojen välillä suoraan. Intuitiivisesti järkevämmältä tuntuisi että olisivat osaksi joidenkin latenttien muuttujien, kuten käsitteiden ja/tai sanaluokkien tms välillä.
- Kuitenkin: n-grammimalli yhdistää syntaktiset ja semanttiset ja kollokationaaliset lyhyen kontekstin riippuvuudet käytännössä yllättävänkin hyvin toimivalla tavalla, etenkin/ainakin englannille.
- Mallin optimointiin ja tasoitusmenetelmien parantamiseen on käytetty hyvin paljon resursseja. On mahdollista että on juututtu lokaaliin minimiin malliperheiden suhteen.

9. Markov-mallit

9.1 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:**

$$P(X_{t+1} = s_k | X_1, \dots, X_t) = P(X_{t+1} = s_k | X_t) \tag{42}$$

2. **Aikariippumattomuus (stationaarisuus):**

$$P(X_2 = s_k | X_1) = P(X_3 = s_k | X_2) = P(X_{t+1} = s_k | X_t) \tag{43}$$

Markov-ketjun esitys taulukkona

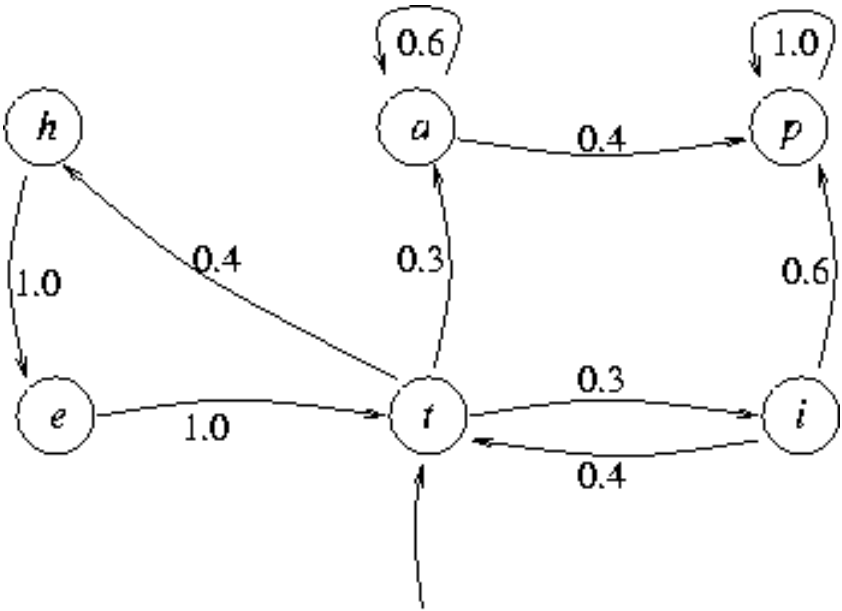
Tilojen joukko $S = \{a, e, i, h, t, p\}$

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:

$$
\begin{aligned}
P(X_1 \dots X_T) &= P(X_1)P(X_2|X_1)P(X_3|X_1X_2)\dots P(X_T|X_1\dots X_{T-1}) && (44)\\
&= P(X_1)P(X_2|X_1)P(X_3|X_2)\dots P(X_T|X_{T-1}) && (45)\\
&= \pi_{X_1}\prod_{t=1}^{T-1} a_{X_tX_{t+1}} && (46)
\end{aligned}
$$

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.

9.2 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}

9.3 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)

$$P(O|X,\mu) = \prod_{t=1}^{T} P(o_t|X-t, X-t+1, \mu) \quad (47)$$

$$P(O|\mu) = \sum_{X} P(O,X|\mu) = \sum_{X} P(O|X,\mu)P(X|\mu) \quad (48)$$

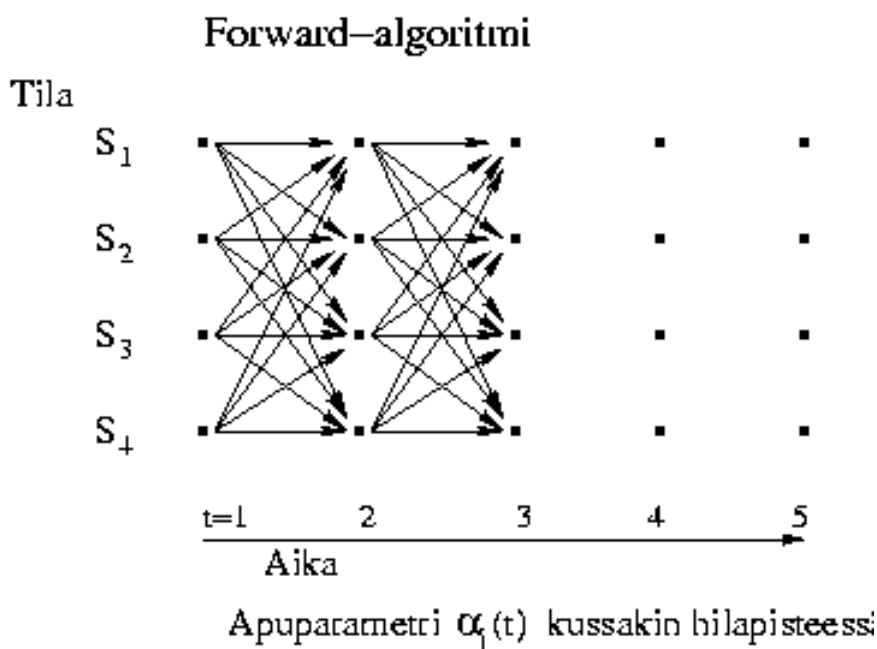
$$= \sum_{X_1 \dots X_{T+1}} \pi_{X_1} \prod_{t=1}^{T} a_{X_t X_t+1} \quad (49)$$

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:

$$P(O|\mu) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t), \text{ jossa } 1 \leq t \leq T+1 \tag{50}$$

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:

$$\delta_j(t) = \max_{X_1 \ldots Xt-1} P(X_i \ldots X_{t-1}, o_1 \ldots o_{t-1}, X_t = j|\mu) \qquad (51)$$

- 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.

9.4 Parametrien estimointi

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

$$\arg\max_{\mu} P(O_{\text{opetusdata}}|\mu) \tag{52}$$

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 $B = b_{ijk}$

9.5 Soveltamistavoista

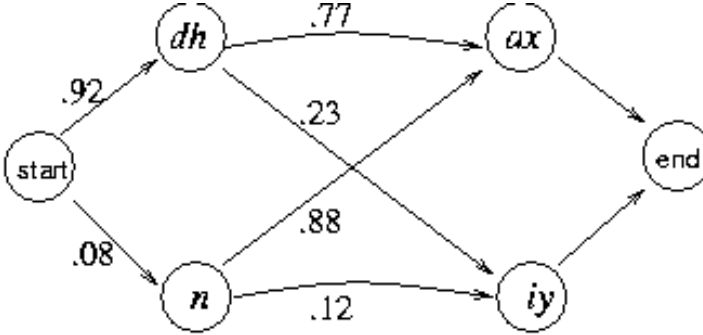
Soveltaminen puheentunnistukseen

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

10. Sanaluokkien taggaus

- Lopullinen tavoite: tekstin ymmärtäminen (ja ymmärretystä ajatuksesta ja kommunikaatiotarpeesta lähtevä tekstin generointi)
- Välitavoitteita mm.: jäsentäminen, pintapuolinen tai osittainen jäsentäm syntaktisten sanaluokkien disambiguointi ja merkitseminen dataan eli *sanaluokkien taggaus* (Part-of-speech tagging, POS tagging).

Esimerkki:

The-AT representative-NN put-VBD chairs-NNS on-IN the-AT table-NN.

Yllä sanoille 'put' ja 'chairs' on olemassa sekä verbi- että substantiivitulkinta.

Ilmiö on yleinen: yleensä substantiivista voi helposti tehdä myös verbin ja useilla pääasiassa verbeillä on myös harvinaisempi substantiivikäyttö.

Next, you **flour** the pan.
I want you to **web** our annual report.

Kyse on siis aidosta disambiguointiongelmasta.

Suomenkielisiä morfo-syntaktisen taggauksen disambiguointiongelmia nähtiin aiemmin mm. laskareissa.

Pohdittavaksi: Missä määrin syntaktinen ja semanttinen disambiguointi ovat päällekkäisiä ongelmia? Milloin kumpaakin tarvitaan?

Syntaktisen taggauksen sovelluskohteita

- Information extraction (jonkin nimenomaisen tyyppisen tiedon esiinkaivaminen), esim. erisnimien tunnistaminen tekstidokumenteissa)
- Kysymyksiin vastaaminen (question answering)
- Osittainen jäsentäminen (shallow/partial parsing)
- Yleisesti: tilanteet joissa ei tarvita täydellistä kielen analyysiä ja ymmärtä

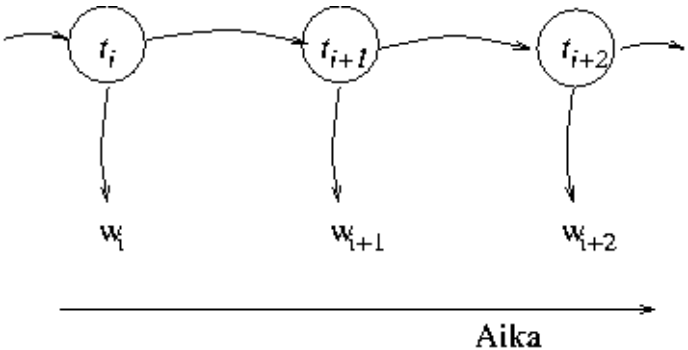
Taggauksessa käytettävä informaatio

- Rakenneinformaatio, eli lähiympäristön syntaktiset tagit: tietyt tagisekvenssit tavallisempia kuin toiset
- Sanakohtainen informaatio: Sanan prioritn kuulua tiettyyn syntaktiseen luokkaan: eri luokkien tn-jakauma yleensä hyvin epätasainen kullekin yksittäiselle sanalle (vastaavasti kuin sem. moniselitteisyydessä, jokin tulkinta on hyvin tyypillinen vaikka useita eri mahdollisia tulkintoja)

Joitain tuloksia englannille:

- Taggaus helpompi ongelma kuin jäsentäminen, tarkkuudet korkeita
- Parhaat taggerit luokkaa 96%-97% (oikein tagattujen sanojen osuus, tarkoittaa että lauseissa keskimäärin 1-2 taggausvirhettä jos lausepituus keskim. 20).
- Pelkkää rakenteista informaatiota käyttävälle sääntöpohjaiselle taggerille raportoitu (vain) 77% tarkkuus.
- Yksinkertainen menetelmä joka ei käytä tietoa rakenteista lainkaan: Luokitellaan sana aina yleisimpään POS-luokkaansa. Englannille raportoitu 90% tarkkuus $\rightarrow$ käytetään usein baseline-menetelmänä.

10.1 Markov-malli-taggerit



- Tagijono mallinnetaan Markov-ketjuna, eli $P(X_{i+1} = t^j | X_1, \ldots, X_i) = P(X_{i+1} = t^j | X_i)$ jossa i on ajanhetki ja t^j on tila jonka indeksi j. X:n arvojoukko on tilojen joukko, $\{t^1 \ldots t^n\}$.
- Sanat ovat havaintoja (observations), todennäköisyysmalli $P(w_i | X_i)$ eli sanan generointitn riippuu vain kulloisestakin tilasta.
- Bigrammitaggerissa jokainen tagi vastaa yhtä tilaa. Tällöin tämänhetkise

sanan tagi riippuu ainoastaan edeltävän sanan tagista.

- Malli opetetaan tagatulla datalla, näkyvänä Markov-mallina, eli tilamuuttujan arvo tunnetaan kullakin hetkellä (kullakin sanalla).

- Tagattaessa uutta dataa, mallia käytetään HMM-mallina: tiloja ei tunneta, vaan todennäköisin tilajono annetulla sanajonolla lasketaan mallista Viterbi-algoritmilla. Nyt Viterbin käyttö ei ongelmallista, koska ollaan kiinnostuttu nimenomaan todennäköisimmästä tilajonosta, ei sanajonosta.

- Markov-mallin rajallinen horisontti-riippumattomuusoletus ei aivan päde, edes englannille. Vielä vähemmän kielille joissa sanajärjestyksen määräyty syntaktiset tekijät eivät ole yhtä keskeisiä.

- Sanojen generointitodennäköisyydet tageista:

$$P(w_{1\ldots n}|t_{1\ldots n}) = \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1}) \tag{53}$$

Estimoinnissa tehtäviä muita riippumattomuusoletuksia siis: Sanat riippumattomia toisistaan. Sanan tn. riippuu ainoastaan sen tagista (eli tagi generoi sanan).

- Optimaalisen tagijonon estimointi lauseelle, sovelletaan Bayesin sääntöä:

$$\hat{t}_{1\ldots n} = \arg\max_{t_{1\ldots n}} P(t_{1\ldots n}|w_{1\ldots n}) \quad (54)$$

$$= \arg\max_{t_{1\ldots n}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1}) \quad (55)$$

Intuitiivisesti: Valitaan maksimaalisen todennäköinen reitti jolla päästään markov-ketjua pitkin tagista t_{i-1} sanaan w_i, eli maksimaalisen todennäköinen tapa generoida havaittu sana w_i kun on kiinnitetty tagi t_{i-1}.

- Huom: Kannattaa käyttää tasoitusta laskettaessa $P(t^k|t^j)$ ja $P(w^l|t^j)$ (ei siis pelkkiä ML-estimaatteja).
- Todennäköisimmän tagijonon haku tehokkaasti Viterbi-algoritmilla.

Tuntemattomien sanojen käsittely

Tunnetuille sanoille estimointi helppoa. Kuitenkin tuntemattomat sanat aiheuttavat usein suurimmat erot taggerien välillä.

Strategioita

- Yksinkertaisin lähestymistapa: Tuntemattoman sanan tagitn:t seuraavat tagijakaumaa koko datan yli (ts. tuntemattoman sanan malli on painotettu keskiarvo kaikkien sanojen malleista).
  Ongelma: ei kovin hyvä estimaatti.

- Muiden piirteiden, esim. morfologian hyväsikäyttö: valitaan morfologialtaan samankaltaisten sanojen osajoukko ja lasketaan tn:t siitä. Esimerkki: jos sana loppuu '-ed', keskiarvoistetaan tagijakauma -ed -loppuisten sanojen yli.

- Eräs malli (Weichschedel jne): katsotaan todennäköisyyttä jolla tagi generoi tuntemattomia sanoja, sekä sanan eri piirteiden generointitn:iä

tästä tagista:

$$P(w^l|t^j) = \frac{1}{Z}P(tuntematon|t^j)P(isoalkuk|t^j)P(lopuke_i|t^j)$$

jossa Z on normalisointitekijä ja $lopuke_i$ sanan w^l lopuke (esim. '-ed'). Joissain kokeissa mallin todettiin pudottavan tuntemattomien sanojen virhetn:iä 40% 20%:een (tosin ei kerrota mikä oli vertailumenetelmä, tai datan koko).

- Useimmat mallit olettavat piirteet riippumattomiksi (ns. Naive Bayes-malli, kuten yllämainittu), mikä yleensä ei pidä paikkaansa. Esim. yllä isoalkukirjaimiset sanat ovat melko todennäköisesti myös tuntemattomia, koska ovat todennäköisesti erisnimiä, joten ko. piirteet eivät toisistaan riippumattomia.

Ks. kirjan taulukko 10.5.

Variantteja

- Voisi olettaa että oikeasti tagin tn. voi riippua pidemmästä historiasta, esim. 2 edeltävästä tagista.
- Esim. '...was clearly marked...' ja '...he clearly marked...' tagattaisiin 'BEZ RB VBN' ja 'PN RB VBD'.
- Kahden edellisen tagin (ja sanan itsensä) perusteella ennustamista voidaan kutsua trigrammitaggaukseksi.
- Historian pidentämisestä ei aina ole hyötyä: esim. syntaktiset riipuvuudet harvoin kulkevat pilkkujen yli. Harvan datan ja puutteellisen tasoituksen takia pitkästä historiasta voi olla haittaakin, ja trigrammitaggeri voi pärjätä bigrammitaggeria huonommin.
- Kuten puheentunnistuksessa, voidaan myös käyttää lineaarista interpolointia eri n-grammitaggerien yli, tai muita tasoitusmenetelmiä.
- Variable-Memory Markov Model (VMMM): eri tiloissa voi olla tiedossa eri pituinen historia. Opetusvaiheessa tilan historian pituus valitaan in-

formaatioteoreettisella kriteerillä. Voidaan rakentaa joko topdown (tiloja halkomalla) tai bottom-up (yhdistelemällä).

10.2 HMM-taggerit

- Piilo-Markov-mallia voidaan soveltaa myös opetusvaiheessa, jos ei ole tagattua esimerkkidataa. Esim. kieli jolle ei olemassa tagattua dataa, tai tunnetun kielen osa-alue jossa sanojen generointitn:t ja/tai kielen tyypilliset rakenteet erilaisia kuin mitä opetusdatassa.
- Mallin rakennusosat: Tilat S, havainnot O, tilojen lähtötodennäköisyydet π_i, tilasiirtymätn:t a_ij, havaintojen generointitn:t b_{ijk}
- Kuten näkyvillä Markov-malleilla, tilat vastaavat jälleen tageja ja havainnot ovat sanoja.
- Periaatteessa voidaan initialisoida mallin parametrit eli todennäköisyydet π_i,a_ij, ja b_{ijk} satunnaisesti ja estimoida niitä iteratiivisesti yhä paremmiksi (esim. forward-backward-algoritmilla).
- Kuitenkaan tällä tavalla ei välttämättä päädytä taggaukseen joka vastaisi jotain olemassaolevaa lingvististä tagijoukkoa ja tagien syntaktisia rooleja. Pikemminkin tällä tavalla voidaan 'keksiä' taggaus joka toteuttaa mallin riippumattomuusoletukset.

- Tavallisemmin käytettävissä on tunnettu tagijoukko, sekä sanakirja jossa kerrotaan mitkä tagit mahdollisia tai mahdottomia millekin sanalle (esim. JJ ei mahdollinen sanaluokka sanalle 'book'). Eri tyyppisen sanakirjatiedon vaihtoehdot on kuvattu taulukossa alempana.

- Vaihtoehtoisesti ryhmitellään sanat joille sallittu samat tagit, ekvivalenssiluokiksi, joille yhteiset parametrit (ainakin mallin initialisointivaiheessa). Voidaan soveltaa myös pelkästään harvinaisille sanoille, koska yleisten osalta dataa on riittävästi.

	Leksikaalinen resurssi	Strategia
L_0	Jokaiselle sanalle tunnetaan sallitut tagit	Sallituille tageille T^s $p(w\|t^j) = 1/(\#T^s)$, muille p=0.
L_1	Sallitut tagit tn-järjestyksessä	Annetaan satunnaiset tn-arvot, mutta järjestyksessä.
L_2	Tagien tn:t annettu kullekin sanalle	Käytetään näitä.

Koska tn-malli ei täysin vastaa todellisuutta, jos on käytettävissä riittävästi tagattua dataa, kannattaa opettaa pääasiassa sillä.

Täysin ohjaamattomasta oppimisesta vaikuttaisi olevan hyötyä lähinnä mikäli tagattua dataa on kovin vähän tai ei ollenkaan, tai jos tagattu testiaineisto (tai sovelluskäyttö) on melko erilaista kuin tagattu opetusaineisto.

Eräs tapa hahmottaa syy tähän: mallin rakenne sinällään ei vastaa kovin hyvin tarkoitusta, ja ilman tagattua opetusaineistoa mallin rakenteen merkitys dominoi.

Parametrien optimoinnissa voidaan käyttää erillistä (tagattua) validointijoukkoa jolla varmistetaan että opetusta jatketaan vain niin kauan kuin tarkkuus validointijoukolla paranee.

10.3 Muunnoksiin perustuva taggaus

- Edellä kuvatut mallirakenteet, eli mallien tekemät riippumattomuusoletukset, eivät erityisen hyvin soveltuneet luonnollisen kielen kuvaamiseen. Tarvitaan siis parempia malleja.
- Kontekstia (n-grammin n) voitaisiin pidentää. Tai tagin tn voisi riippua myös edeltävistä sanoista (ei pelkästään tageista). Ongelma: parametrien määrä moninkertaistuu, estimointiongelmia.

Toisenlainen lähestymistapa: **Muunnoksiin perustuva taggaus (transformation-based tagging)**

Soveltaminen ohjatun oppimisen ongelmaan:

- Tagattua dataa
- Sanakirja, jossa kerrotaan sanalle sallitut tagit ja näiden tn:t.
- joukko kontekstiriippuvia muunnossääntöjä ('virheenkorjauksia') joita taggaukselle voidaan tehdä.

- Algoritmi jolla valitaan mitä muunnoksia milloinkin kannattaa soveltaa (ohjattu oppiminen)

Perusalgoritmi:

1. Tagataan aluksi jokainen sana todennäköisimmän taginsa mukaan.
2. Valitaan muunnoksia jotka vievät taggausta vähitellen lähemmäksi oikeaa (opetusaineistossa olevaa) taggausta.

Muunnokset

- Muunnossääntö joka kertoo mikä tagi korvataan millä. Esim. 'korvaa tagi VBD tagilla NN'
- Heräteympäristö (triggering environment): olosuhteet joissa muunnos aktivoituu. Esim. 'Tagi t^j esiintyy 2-3 sanaa ennen korvattavaa tagia ja t^k korvattavaa tagia seuraavassa sanassa'. ks. kirjan taulukko 10.7.

- Heräteympäristöön voidaan ottaa myös sanoja tai näiden ominaisuuksia, ei pelkästään tageja:
  Tag-triggered: heräteympäristössä voi olla tageja
  Word-triggered: heräteympäristössä voi olla sanoja
  Morphology-triggered: heräteympäristössä voi olla morfologisia piirteitä

Esimerkkejä muunnossäännöistä ja herätteistä englannille:

Muunnos	Heräte
NN $\Rightarrow$ VB	edellinen tagi oli TO
VBP $\Rightarrow$ VB	jokin kolmesta edellisestä tagista oli MD
JJR $\Rightarrow$ RBR	seuraava tagi on JJ
VBP $\Rightarrow$ VB	toinen kahdesta edellisestä sanasta ei ole *n't*

Oppimisalgoritmi

Sovelletaan ahnetta optimointia, valitaan joka kierroksella se transformaatio joka eniten vähentää virheellisten taggausten lukumäärää.

Notaatio: Transformaatiot $u_i(\cdot)$, $v(\cdot)$, C_k on korpus k:n transformaation soveltamisen jälkeen. $E(\cdot)$ on virhemäärä ja ϵ virheen pieni kynnysarvo.

Algoritmi:

- Alustus: C_0, korpus jossa jokainen sana tagattuna yleisimmällä tagillaan.
- for (k=1; ; k++)
  - v:= valitse transformaatio u joka minimoi virheen $E(u_i(C_k))$
  - Jos v ei pienennä virhettä tämänhetkiseen verrattuna enempää kuin ϵ, lopeta.
  - Sovella transformaatiota korpukseen: $C_{k+1} = v(C_k)$

- Tulosta tagijono.

Päätettävä: muunnosten soveltamisjärjestys datassa (esim. vasemmalta oikealle) ja käytetäänkö välitöntä muuntamista vai viivästettyä. Jos välitöntä, muunnosten soveltamisjärjestyksellä on väliä.

Soveltaminen ohjaamattomassa oppimistilanteessa

Tilanne: ei tagattua dataa, mutta tunnetaan joka sanalle sallitut tagit (sanakirjasta).

Huom: Useimmilla sanoilla vain yksi sallittu tagi, eli useimpien sanojen tagi tiedetään ennalta. Vain osa tageista epäselviä.

Periaate: Käytetään samassa kontekstissa esiintyvien, tagiltaan yksiselitteisten sanojen tagijakaumaa epäselvän tagin ennustamiseen.

Transformaation hyvyys lasketaan siten, että kuvitellaan tunnetut, yksikäsitteis sanojen luokat tuntemattomiksi ja sovelletaan transformaatiota niiden luokitteluun. Pienimmän osuuden virheluokituksia aiheuttava transformaatio on

paras.

Esimerkki. 'The **can** is open' AT _ IS
Kontekstissa 'AT _ IS' olevat sanaluokaltaan yksikäsitteiset sanat ovat (lähes) aina substantiiveja, eivät verbejä. → 'can' tagataan verbiksi.

Tämänkaltainen 'ohjaamaton' soveltaminen: 95,6% (Brill, 1995)

Hyvä puoli: ei juuri ylioppimista, toisin kuin HMM:llä.

[Huomautus: HMM:illäkin voidaan periaatteessa välttää ylioppimista soveltamalla mallin rakenteen optimoimista (parametrien karsimista), jos käytetään täyttä Bayeslaista estimointia, esim. enseble-oppimista (variaatioanalyysiä).
]

Haaste: potentiaalisia transformaatioita (erityisesti herätekonteksteja) hyvin suuri joukko.

10.4 Yhteys muihin menetelmiin

Päätöspuut (Decision Trees)

Labeloidaan kaikki puun haaraan kytkeytyvä data ko. haaran majority-luokalla.

Puuta haaroitetaan sen mukaan että alemman tason datan luokittelussa tehtäisiin mahdollisimman pieni osuus virheitä.

Huono puoli: potentiaalisia sääntöjä hyvin suuri joukko. Hakua sääntöjoukossa voidaan kuitenkin nopeuttaa.

Pääasiallinen ero muunnostaggaukseen: Päätöspuu jakaa datajoukon osiin, ja myöhemmät transformaatiot operoivat ainoastaan ko. haaran osadatalla.

Muunnostaggauksessa datan osajoukko johon muunnosta sovelletaan valitaan heräteympäristön perusteella koko datasta.

Eroja aitoon probabilistiseen mallinnukseen verrattuna

Ei käytettävissä prob.mallinnuksen kaikkea välineistöä.

Esim. laajentaminen tilanteeseen jossa tuotetaan yhden parhaan tagin sijaan k parasta ei yhtä suoraviivaista

Prioritiedon huomioiminen:

Muunnostaggaus pystyy helposti huomioimaan heräteympäristojen muodossa annetun prioritiedon.

Ei kykene hyödyntämään eri luokkien prioritn:iä, ainoastaan tiedon sanan todennäköisimmästä luokasta (initialisoinnissa).

Muita eroja:

Joustavuus: Muunnostaggauksessa hyödynnetään hyvin joustavaa kokoelmaa potentiaalisia vaikuttavia tekijöitä (piirteitä) kussakin vaiheessa ja kullekin transformaatiolle.

Ymmärrettävyys: Binääriset säännöt ovat yksinkertaisempia ihmiselle ymmärtä
Kuitenkin sekvenssistä jonkin yksittäisen säännön muuttaisen vaikutusta on
vaikea ennustaa, johtuen sääntöjen välisestä interaktiosta.

Yhteys automaatteihin

Taggerin sääntöjen oppiminen tapahtuu kvantitatiivisesti.

Valmis munnostaggeri voidaan kuitenkin muuttaa deterministiseksi FST:ksi
ja saada sille näin tehokas toteutus.

10.5 Taggauksen evaluoinnista

Taggausprosentit englannille tyypillisesti luokkaa 95-97%, kun raportoidaan kaikkien sanojen yli (ei vain moniselitteisten).

Tulokseen vaikuttavat mm. seuraavat tekijät:

- Datan määrä (isommalla datalla tulee parempia tuloksia)
- Tagijoukko: yleensä, mitä enemmän tageja, sen vaikeampi ongelma. Toisaalta, jos käytetän joillekin sanoille ihka omia tageja (esim. 'to'= TO) ei näitä voi tagata väärin.
- Erot opetusdatan, sanakirjan ja sovelluksen (testidatan) välillä
- Tuntemattomien sanojen tn: vaikuttaa suuresti onnistumisprosenttiin.

Eri (hyvien) menetelmien väliset erot ovat puolen prosentin luokkaa. Yllämainit mm. aineistokohtaisten ominaisuuksien aiheuttamat erot ovat usein paljon suurempia.

Kuitenkin pienikin sanakohtainen parannus menetelmässä aiheuttaa merkittävän lausekohtaisen parannuksen, mikä on taggausta hyödyntävien sovellusten kannalta relevanttia.

Eräs parhaista tunnetuista taggereista englannille: Helsingin Yliopistossa kehitetty EngCG (Voutilainen, Tapanainen et.al): ihmisen kehittämä sääntöpohja taggeri (asiantuntijajärjestelmä taggaukseen). 99% tarkkuus. Muistuttaa transformaatitaggausta paitsi että sääntöjen valinnan tekee ihmisasiantuntija.

Taggausten sovelluksista

- Yllättävän vähän julkaisuja taggauksen sovelluksista.
- Useissa sovelluksissa tarvitaan lisäksi *osittaisjäsennys*
- Information extraction: taggausta jossa syntaktisten kategorioiden sijaan pyritään tunnistamaan (tiettyjä) semanttisia kategorioita (esim. erisnimet). Syntaktisesta kategoriasta voi olla apua.

- Tiedonhaun indeksointitermien valinta tai painotus (sekä taggaus että osittaisjäsennys)

- Kysymyksiin vastaus: 'Who killed president Kennedy' , vastaus 'Oswald' edellyttää että ymmärretään mikä asiaan liittyvä tieto (esim. aika, paikka, vai henkilö) kysyjälle pitäisi kertoa. Lisäksi on pystyttävä eristämään oikea tieto dokumenteista jotka aiheeseen liittyvät. Molemmissa voi olla hyötyä taggauksesta.

- Negatiivinen tulos taggauksen hyödyllisyyden kannalta: Parhaat sanatietoa hyödyntävät prob.jäsentimet toimivat paremmin lähtemällä taggaamattomasta tekstistä ja taggaamalla sitä itse kuin hyödyntämällä valmista taggausta.

11. Probabilistinen jäsentäminen ja PCFG:t

PCFG=Probabilistic Context Free Grammar
Todennäköisyyksiin perustuva kontekstiton kielioppi

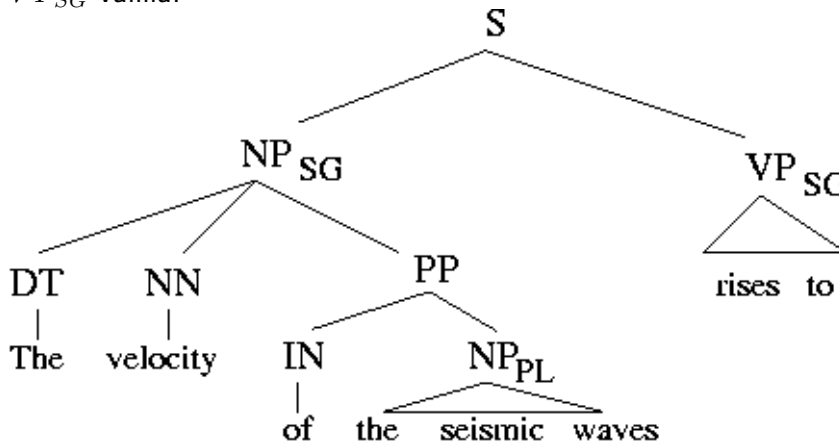
Motivaatio

- Tähän asti kielen säännönmukaisuuksia kuvattu vain sanatasolla (n-grammitmallit) tai sanakategoriatasolla (sanaluokat ja HMM)
- Sekventiaalisten riippuvuuksien lisäksi muunkinlaisia ilmiöitä, esim. rekursiivisia.
- Esim. 'The velocity of the seismic waves rises to...'
  P('waves rises') on hyvin pieni esim. sanaluokkiin perustuvalla HMM-taggerilla, koska yksiköllistä verbimuotoa harvoin edeltää monikollinen substantiivi (yleensä sitä edeltää yksiköllinen substantiivi, verbin subjekti. Ilmiötä kutsutaan nimellä 'verb agreement').
- Keskeinen havainto: yksittäisten sanojen sijaan riipuvuudet voidaan

ehkä paremmin kuvata suurempien kokonaisuuksien välillä → oletetaan hierarkkinen rakenne myös lauseiden tai virkkeiden sisällä. (Vastaava hierarkkisuus yleisemmällä tasolla itsestäänselvä: dokumentti, kappale, lause, sana, kirjain, veto).

- Vastaavaa hierarkkisuutta myös muualla kuin kielessä, esim. konenäön hahmontunnistusongelmat, joihin myös voidaan soveltaa syntaktisia hahmontunnistusmenetelmiä.

Esimerkki lauseen sisäisestä hierarkkisesta rakenteesta (kirjan kuva 11.2). Voidaan ajatella että riippuvuus on tässä peräkkäiden rakenneosien NP_{SG} ja VP_{SG} välillä.



11.1 Mikä on PCFG

CFG:n luonteva laajennus: CFG jonka sääntöihin on liitetty laukeamistodennäköisyydet. [CFG = kontekstivapaa kielioppi]

PCFG on probabilististen kielioppien eräs alalaji, ts. vain eräs tapa. mallintaa hierarkkista rakennetta probabilistisesti. Tekemällä erilaisia riippumattomuusoletuksia päädytään erilaiseen malliperheeseen.

PCFG:n osaset:

- terminaalisymbolien joukko
- ei-terminaalisymbolien joukko,
- joukko korvaussääntöjä $N^i \rightarrow \gamma^j$ jossa γ^j on terminaali- ja ei-terminaalisymboleista koostuva jono
- Kullekin säännölle ehdollinen laukeamistodennäköisyys:

$$\sum_j P(N^i \rightarrow \gamma^j | N^i) = 1 \tag{56}$$

Notaatio:

G	kielioppi
$\mathcal{L}_G$	Kieli jonka kielioppi G tuottaa
t	jäsennyspuu
tulostus(t)	jäsennyspuun t jäsentämä sanajono
$\{N^1, \ldots, N^n\}$	Ei-terminaalisymbolien aakkosto (N^1 on alkusymboli)
$\{w^1, \ldots, w^V\}$	Terminaalisymbolien aakkosto
$w_1 \ldots w_m$	Jäsennettävä lause (sanasekvenssi)
N^j_{pq}	Ei-terminaalisymboli N^j kattaa sekvenssissä positiot p:stä

Riippumattomuusoletukset

PCFG:ssä korvaussäännön tn on ehdollinen ainoastaan sille missä ei-terminaalis
nyt ollaan, eli N^i.

Tarkastellaan tarkemmin riippumattomuusoletuksia joita tässä tehdään. (Esi-
merkki: ks. kielioppi kirjan taulukossa 11.2 ja sitä soveltaen generoidut 2
jäsennystä kuvassa 1.11.)

- Rakenteellinen paikkariippumattomuus: alipuun tn ei riipu siitä missä
  osassa isompaa puuta alipuu sijaitsee (vrt. HMM:n ajallinen stationaa-
  risuus).

- Leksikaalinen kontekstiriippumattomuus: alipuun tn ei riipu alipuuhun
  kuulumattomista kontekstin sanoista.

- Riippumattomuus esivanhemmista: Alipuun tn ei riipu siitä mitä sääntöjä
  käyttäen alipuu generoitiin.

Huomioita probabilistisista kieliopeista

- Antaa probabilistisen mallin kielen generoinnille.
- Arvioi kunkin jäsennyksen todennäköisyyden. [Kattavissa kieliopeissa yhdelle lauseelle yleensä hyvin monta mahdollista jäsennystä, jopa tuhansia per lause].
- Prob. kielioppi (esim. PCFG-malliperhettä käyttäen opittu) ei yritäkään jakaa lauseita ei-kieliopillisiin ja kieliopillisiin. 'Virheelliset' lauseet vain ovat paljon epätodennäköisempiä.
- Jos dataa riittävästi, PCFG:llä kieliopin oppiminen mahdollista ilman negatiivista evidenssiä (ei-kieliopillisia lauseita).

Peruskysymykset PCFG:lle

- Lauseen todennäköisyyden laskeminen
- Todennäköisimmän jäsennyksen valinta lauseelle
- PCFG:n parametrien estimointi joko jäsennetystä (ohjattu oppiminen) tai jäsentämättömästä datasta (ohjaamaton oppiminen)

Keskitytään tästedes kontekstivapaisiin kielioppeihin jotka ovat *Chomskyn normaalimuodossa* eli säännöt ovat binäärisiä tai unaarisia:

$N^i \rightarrow N^j N^k$
$N^i \rightarrow w^j$

PCFG:n parametrien määrät ovat tällöin

$P(N^j \rightarrow N^r N^s \mid G)$	Jos n ei-terminaalia, n^3 parametria
$P(N^j \rightarrow w^k \mid G)$	Jos V terminaalia, nV parametria

Kaikille j pätee lisäksi:

$$\sum_{r,s} P(N^j \to N^r N^s) + \sum_k P(N^j \to w^j) = 1 \tag{57}$$

(Tämä johtuu siitä että yo. parametrit ovat ehdollisia todennäköisyyksiä ehdolla N^j)

Chomskyn hierarkia kielille

Konteksti-vapaus on eräs kielen kompleksisuutta kuvaava taso. Chomsky jakaa kielet seuraavanlaiseen kompleksisuushierarkiaa: (Taulukko on kirjasta Jurafsky & Martin, s. 479)

Tässä A on yksittäinen ei-terminaalisymboli ja α, β ja γ ovat mitä tahansa terminaalien ja ei-terminaalien jonoja.

Tyyppi	**Nimi**	**Sääntörunko**
0	Turing-ekvivalentti	$\alpha \rightarrow \beta$ s.e. $\alpha \neq \epsilon$
1	Kontekstiherkkä	$\alpha A \beta \rightarrow \alpha \gamma \beta$ s.e. $\gamma \neq \epsilon$
2	Kontekstivapaa	$A \rightarrow \gamma$
3	Säännöllinen	$A \rightarrow xB$ tai $A \rightarrow x$

Tyypin 0 kielet ovat rekursiivisesti numeroituvia, eli kuvaavat ne kielet joiden stringit voidaan tuottaa (listata) Turing-koneella. Ainoa rajoitus säännöille on että tyhjästä ei voi nyhjästä.

Kontekstiherkät kielet muuntavat symbolin toiseksi riipuen sen oikean- ja vasemmanpuoleisesta kontekstista. Lisäksi nämä symbolit eivät voi pelkästään kadota, vaan sääntöjen on myös tuotettava jotain.

Kontekstivapaissa kielissä ei-terminaalisymboli voidaan korvata millä tahansa ei-terminaalien ja terminaalien jonolla (mukaanlukien tyhjä jono). Näitä toteuttavat esim. lauserakennekieliopit.

Säännölliset kielet ovat ekvivalentteja säännöllisten lausekkeiden (regular expressions) kanssa. Ne voivat olla oikea- tai vasenkätisesti lineaarisia. Niitä toteuttavat esim. äärelliset tilakoneet.

Yhtäläisyys HMM:n ja PRG:n välillä

PRG = probabilistic regular grammar, eli todennäköisyyksiä hyödyntävä säännö kieli.

Esimerkkilause *P(John decided to bake a)* saisi luultavasti korkean todennäköis HMM:ssä (koska se on järkevä sanajono) mutta matalan PRG:ssä (koska ei

ole kokonainen lause).

PRG voidaan kuitenkin implementoida HMM:n avulla lisäämällä HMM:ään alkutilan lisäksi lopputila, joka vastaa lauseen päättymistä.

11.2 Lauseen todennäköisyyden laskeminen:

$$P(w_{1m}|G) = \sum_t P(w_{1m}, t|G) \quad (58)$$

$$= \sum_{t, tulostus(t)=w_{1m}} P(t|G) \quad (59)$$

$$(60)$$

jossa t on kieliopin G säännöillä tuotettu jäsennyspuu lauseelle w_{1m}.

Oletetaan että t:n aikaansaamiseksi sovellettiin G:n sääntöjä s^i,s^k, ja s^a ja tulostuksena oli sanajono w_{pq}. Tällöin

$$\begin{aligned} P(t|G) &= P(s^i s^k s^a) = P(s^i, s^k, s^a) \\ &= P(s^a|s^i, s^k) P(s^k|s^i) P(s^i) \\ &= P(s^a) P(s^k) P(s^i) \end{aligned}$$

(ylläolevassa kaikki tn:t ovat ehdollisia kieliopille G joten tämä jätettiin merkitsemättä).

Eli $P(w_{pq}|G)$ on niiden G:n sääntöjen todennäköisyyksien tulo joita t:n muodostamiseksi sovellettiin.

Yleisessä tapauksessa ei kannata (laskennallisista syistä) summata yli kaikkien mahdollisten jäsennysten. Syy on sama kuin HMM:llä, mahdollisia 'polkuja' on liikaa. Samoin kuin HMM:llä, voidaan ongelma kuitenkin paloitella osaongelmiksi ja ratkaista palat kerrallaan, sekä vierittää palojen tuloksia eteenpäin.

Inside-algoritmi
(vastaa HMM:ien forward-algoritmia)

$b_j(p,q)$ tarkoitaa N_j:n alla olevan alipuun, joka kattaa sanajonon w_{pq}, tn:n laskemista

Perustapaus: lasketaan tn kun muunnetaan ei-terminaali terminaaliksi:
$b_j(k,k) = (w_k|N_k^j k, G)$

Induktioaskel: tn kun muunnetaan ei-terminaali kahdeksi ei-terminaaliksi:

Olet. sääntö $N^j \rightarrow N^r N^s$ ja stringin jako $w_{pq} = w_{pd} w_{dq}$

Tällöin $b_j(p,q) = \sum_{r,s} \sum_{d=p}^{q-1} b_r(p,d) \times b_s(d+1,q)$

Eli samalla kun puu jakautuu kahdeksi osapuuksi, stringi w_{pq} jaetaan kahtia, ja rekursiivisesti sovelletaan induktioaskelta kumpaankin osapuuhun.

Lisäksi summataan yli kaikkien mahdollisten stringin jakojen ja kaikkien ei-terminaalisymbolien nimien jotka voidaan muuntaa N^j:stä kieliopissa G.

Inside-todennäköisyydet voidaan laskea tehokkaasti bottom-up.

Todennäköisimmän jäsennyksen etsiminen

Kuten HMM:llä, voidaan soveltaa Viterbi-tyypistä algoritmia maksimi'polun' (tässä siis jäsennyspuun) etsimiseen.

Tämä on kompleksisuudeltaan $\mathcal{O}(n^3m^3)$ jossa m on stringin pituus ja n eiterminaalien lukumäärä kieliopissa.

11.3 PCFG:n estimointi

Annettuna sääntöjoukko:

Tapaus 1: Jokin olemassaoleva CFG joka hyväksyy koko aineiston.

Tapaus 2: Kaikkien mahdollisten sääntöjen joukko.

Tapaus 3: Kaikkien mahdollisten sääntöjen joukko, paitsi että oletetaan valmiiksi jotain rakennetta, esim. jokin ei-terminaalisymbolien osajoukko varattu generoimaan pelkästään terminaalisymboleja.

Estimointi, kun käytettävissä jäsennettyä aineistoa

Annettuna sääntöjoukon Tapaus 1 sekä 'puupankki' (**treebank**) eli suuri määrä jäsennettyä aineistoa.

Lasketaan suoraan sääntöjen soveltamiskertoja ja normalisoidaan.

ML-estimaatti:

$$P(\alpha \rightarrow \beta) = \frac{Count(\alpha \rightarrow \beta)}{Count(\alpha)}$$

Estimointi, kun käytettävissä vain jäsentämätöntä aineistoa

Jos sääntöjoukon tapaus 1: Jäsennetään aineisto annetulla CFG:llä ja tämän jälkeen kuten edellä.

Käytännössä jäsennykset moniselitteisiä: yhden jäsennyksen sijaan painotetaan eri jäsennyksiä niiden tn:llä.

Sääntöjoukon tapaukset 2 ja 3:

Kuten HMM:llä, sovelletaan erästä EM-algoritmia, Inside-Outside-algoritmia, joka muistuttaa forward-backward-algoritmia.

Mallin ML-estimointi: Maksimoidaan datan todennäköisyys.

Parametreja ovat sääntöjen tn:t.

Perusperiaate:

- Initialisointi esim. satunnaisilla parametreilla.
- Vaihe 1: Sovelletaan tämänhetkistä mallia (PCFG:tä) datan tn:n laskemiseen, ja samalla pidetään kirjaa siitä kuinka paljon mitäkin sääntöä käytettiin.
- Vaihe 2: Päivitetään sääntöjen tn:t vaiheen 1 kirjanpidon perusteella.
- Toistetaan vaiheita 1 ja 2.

Ongelmia oppimisessa

- Ongelman kompleksisuus, ja tästä johtuva estimoinnin hitaus. Jokaiselle lauseelle jokainen iteraatio on kompleksisuudeltaan $\mathcal{O}(m^3n^3)$ jossa m on lauseen pituus ja n ei-terminaalien määrä lauseessa.

- Kustannusfunktion lokaalit maksimit hyvin todennäköisiä. Esim. yksinkertaisella keinotekoisella PCFG:stä generoidulla aineistolla estimoitaessa 300 eri ajolla päädyttiin jokaisella eri lokaaliin maksimiin. Eli initialisoinnilla hyvin suuri vaikutus.

- Mallien koko helposti kasvaa suureksi, koska mallin kokoa ei kustannusfunktiossa mitenkään huomioida. Ratkaisu: Kustannusfunktion vaihto sellaiseksi joka huomioi myös mallin koon. Perusteltuja lähestymistapoja tähän ovat MDL ja Bayesilainen estimointi.

- Ei-terminaaleilla joita malli oppii ei välttämättä mitään tekemistä lingvistisen teorian ei-terminaalien kanssa.

Eli, vaikka PCFG:n estimointi jäsentämättömästä datasta on periaatteessa mahdollista, on se käytännössä toistaiseksi hyvin hankala ongelma.

11.4 PCFG:n ongelmia

Säännön soveltamisen todennäköisyys ei riipu siitä missä kohtaa koko puuta alipuu sijaitsee.

Tämä on perua CFG:n riippumattomuusoletuksista, eli oikeus soveltaa jotain sääntöä riippuu ainoastaan senhetkisestä ei-terminaalisymbolista.

Tämä oletus ei kuitenkaan usein päde. Esim. Englannissa suurista aineistoista analysoituna pronominilausekkeen esiintymistn riippuu voimakkaasti sijainnista koko lauseessa:

Väitelauseiden subjektina pronominin P=91% (loput 9% leksikaalisia) kun taas objektina P=34% (loput 66% leksikaalisia).

Mistä tämä riippuvuus johtuu? Eräs hahmotus: Usein lauseen alkupäässä viitataan siihen mitä jo tiedetään aiemman keskustelun pohjalta (ts. voi hyvinkin olla pronomini) ja loppupäässä tuodaan ilmi ko. asiaa koskevaa uutta tietoa. Ilmiö on yleinen monissa muissakin kielissä kuin Englannissa.

Esim:
Presidentti Tarja Halonen aloitti kautensa maakuntakierroksella.
Hän tapasi ensin Joensuun kaupungin johtavia virkamiehiä, ja jatkoi ...

Ongelma ratkeaisi jos P(NP $\rightarrow$ Pronomini) ja P(NP $\rightarrow$ Nomini) ehdollistettaisiin sillä onko NP subjekti- vai objektipositiossa. Mutta juuri tätä eivät PCFG:n riippumattomuusoletukset salli.

PCFG ei huomioi leksikaalista kontekstia.

Englannissa prepositiolausekkeen kiinnittymisongelma:
'I shot an elephant in my pyjamas'
'USA sent more than 10,000 soldiers into Afghanistan'

Sotilaiden lähettämisen tapauksessa leksikaalinen tieto, eli että on kyseessä send-verbi ja into-prepositio auttaa disambiguoimaan esimerkin. Elefantti-esimerkkiin saatetaan tarvita maailmantietoa (että elefantit eivät yleensä käytä yöpukuja, paitsi saduissa).

Suomessa vastaavanlainen monitulkintaisuus:

'Ammuin elefantin yöpuvussa'
→ Ammuin elefantin jolla oli päällään yöpuku.
→ Ammuin yöpukusillani elefantin.

Vastaava ilmiö: Koordinaatiomonitulkintaisuus:

'dogs in houses and cats' $\rightarrow$
1. 'dogs in [NP houses and cats]'
2. '[NP dogs in houses] and cats'

Semanttinen preferointi: koirat eivät mahdu kissoihin, joten 2. vaihtoehto olisi oikea.

PCFG:ssä kuitenkin ylläoleviin lukutapoihin sovellettavat säännöt ovat rakenteisesti identtisiä, eli ei kykene preferoimaan toista tai toista.

Joitain havaintoja toimivuudesta englannille:

- PCFG:n ennustuskyky tavallisesti parempi kuin HMM:n kun sama määrä parametrejä.
- Englannille PCFG yleensä huonompi kielimalli kuin n-grammimalli. Johtuu siitä ettei kontekstin leksikaalista tietoa huomioida (riippumattomuusoletukset).
- Puhdas PCFG ei hyvä, mutta ehdollistaminen leksikaalisella tiedolla tai puun rakennekontekstilla voi parantaa tulosta.

Onko PCFG aito kielimalli?

Jotta PCFG olisi aito kielimalli niin täytyisi päteä:
$\sum_{w \in \mathcal{L}} = \sum_t P(t) = 1$
Eli kieliopin tuottamien lauseiden saaman todennäköisyysmassan pitäisi olla 1.

Kuitenkin tämän toteutuminen edellyttää että mallissa ei ole mahdollisuutta äärettömään rekursioon. Jos ääretön rekursio on mahdollista, osa tn-massasta katoaa sinne, eikä päädy koskaan kielen lauseiksi.

Esimerkiksi kielioppi:
S $\rightarrow$ raparperi, P=1/3
S $\rightarrow$ S S, P = 2/3

$$P(\text{raparperi}^n) = \sum_n (2/3)^{n-1}(1/3)^n = 1/2 \text{ kun } n \rightarrow \infty$$

Eli puolet jäsennysten tn-massasta katoaa äärettömään rekursioon. Kyseessä

on *epäkonsistentti* tn-jakauma.

Ei ole ongelma kun:

- Etsitään todennäköisin jäsennys tälle lauseelle
- Vertaillaan kahden lauseen tn:ää tässä kieliopissa
- Estimoidaan PCFG:n parametrit jäsennetystä korpuksesta (Chi & Geman, 1998)

On ongelma kun tehdään vertailuja eri kielioppien/kielimallien välillä, siis kun:

- Lasketaan jonkin lauseen tn tässä kieliopissa ja verrataan tn:ään jossain toisessa kieliopissa (jossa tn-massa kokonaan tai sen erisuuri osuus kieliopin lauseilla)
- Estimoidaan PCFG:n parametrit ohjaamattomasti, ts. jäsentämättömästä korpuksesta
- Tehdään mallin valintaa, eli vertaillaan eri CFG:n sääntöjoukkoja

11.5 Probabilistinen leksikalisoitu CFG

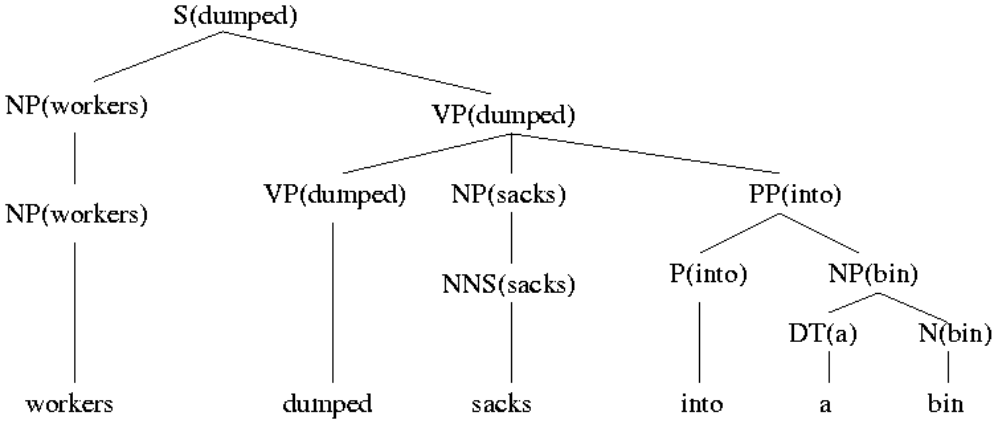
Leksikalisointi tarkoittaa sana-informaation ottamista huomioon

Menetelmä:

- Jokaiseen ei-terminaalisymboli N^i jäsennyspuussa t liitetään ko. alipuun 'pääsana' (head) (Charniak 1997, Collins 1999)

- CFG:ssä jokaisen säännön kohdalle merkitään mikä säännön oikealla puolella olevista symboleista on 'päätytär' ts. sisältää pääsymbolin (esim. NN on NP:n päätytär).

- Päätyttären valinta suoraviivaista oppikirjalauseille, mutta yleisesti ottaen hankalaa (lingvistiikan kirjat saattavat spesifioida säännöt tämän tekemiseen kyseiselle kielelle).



- Kielioppia jossa jokaisella ei-terminaalinoodilla myös jokin terminaali-symboliarvo kutsutaan myös *attribuuttikieliopiksi*.
- PCFG:ssä jokaista päätyttären mahdollista arvoa varten oma todennäköi (ts. oma sääntö). Periaatteessa siis esim:

  $VP(dumped) \rightarrow VBD(dumped)NP(sacks)PP(into)[3 \times 10^{-10}]$
  $VP(dumped) \rightarrow VBD(dumped)NP(cats)PP(into)[8 \times 10^{-11}]$

$VP(dumped) \rightarrow VBD(dumped)NP(hats)PP(into)[4 \times 10^{-10}]$
$VP(dumped) \rightarrow VBD(dumped)NP(sacks)PP(above)[1 \times 10^{-12}]$

- Käytännössä mikään aineisto ei riitä näin valtavan mallin estimointiin $\rightarrow$ tehdään taas tiettyjä riippumattomuusoletuksia, joiden avulla ryhmitellään (sidotaan) osa todennäköisyyksistä samoiksi.

- Esim. voidaan käyttää leksikalisointia vain säännön vasemmalla puolella:
$VP(dumped) \rightarrow VBD\ NP\ PP$

- Eri tilastolliset parserit eroavat siinä minkälaisia riippumattomuusoletuksia tekevät.

- Suuremmasta sääntömäärästä johtuen leksikalisoidun PCFG:n estimointi ohjaamattomalla (ilman jäsennettyä dataa) oppimisella käytännössä mahdotonta, ainakin toistaiseksi.

12. Leksikaalinen semanttinen tieto ja semanttinen samankaltaisuus

Krjan luvut 8.4 ja 8.5 (luennon pohjana käytetty lisäksi Jurafsky & Martin -kirjan lukua 16)

Termejä

Lekseemi: yksittäinen sana-artikkeli sanakirjassa (lexicon).

Leksikaalinen semantiikka: Lingvistinen tutkimusala joka keskittyy lekseemien (sanojen) merkitysten ja niiden sisäisen merkitysrakenteen systemaattiseen tutkimiseen.

Perustavoitteet sanojen merkitysten kuvaamisessa:

- Ensisijainen tavoite: Löytää merkityksen 'perusyksiköt' joiden avulla kaikkien sanojen merkitykset voitaisiin kuvata
- Vaihtoehtoisesti: Kuvata sanojen väliset merkityssuhteet, esim. mer-

kitysten samankaltaisuus. Jos 1. tunnetaan, 2. seuraa siitä suoraan. Kuitenkin 2. on mahdollista kuvata vaikka 1 ei tunnettaisikaan (vrt. etäisyystaulukko).

Millä eri tavoilla kaksi sanaa voivat olla merkitykseltään samankaltaisia?

Lingvistiikan puolella on lukuisia semanttisia teorioita, joita ei kuitenkaan käydä läpi tällä kurssilla. (Lisätietoja, ks. esim. Helsingin yliopiston kurssit).

Wordnet-tietokanta

- Tietokanta joka kuvaa englannin sanojen välisiä semanttisia suhteita.
- Pyrkii soveltamaan konsistentisti tiettyä teoriaa sanojen merkitysten representoinnista ja merkityssuhteista.
- Sisältää n. 130,000 sanaa joilla yli 170,000 merkitystä (Wordnet 1.6)
- Tehty käsityönä ihmisvoimin
- Relevanssi tilastollisille menetelmille: tarjoaa tavoitetason merkitysrepresentaatioiden evaluointiin englannille.

Wordnet kuvaa mm. seuraavanlaisia suhteita lekseemien välillä:

Substantiiveille:

Relaatio	Esimerkki
Osana oleminen, osista koostuminen	pöydänjalka → pöytä
Ryhmän suhde jäseneen	tiedekunta → professori
Yläluokat ja aliluokat	ateria → aamiainen
Vastakohdat	vetäjä → seuraaja

Verbeille:

Relaatio	Esimerkki
Tekemisen kokonaisuus ja osat	nukkua → kuorsata
Ylä- ja aliluokat	matkustaa → lentää
Vastakohdat	suurentua → pienentyä

Adjektiiveille ja adverbeille:

Relaatio	Esimerkki
Vastakohdat	kevyt ↔ painava
	hitaasti ↔ nopeasti

Esimerkki yläluokkarelaatiosta:

bass, merkitys 3 → laulaja → muusikko → esittävä taiteilija → viihdyttäjä (entertainer) → henkilö, ihminen, joku → elämänmuoto, elollinen organismi, olento → olio, jokin

bass, merkitys 7 → soitin → instrumentti → laite → väline → keinotekoinen esine → fyysinen olio → olio, jokin.

Synonymiteetti:

Wordnetissa synonyymeina pidetään sanoja jotka ovat vaihdettavissa toisikseen tietyssä kontekstissa ilman että merkitys oleellisesti muuttuu.

Synset = keskenään tietyssä merkityksessä synonyymisten lekseemien joukko (+ ko. merkityksen kuvaus).

Esim. synset: {*chump, fish, fool, gull, mark, patsy, fall guy, sucker, schlemiel, shlemiel, soft touch, mug*}

Yo. synsetin 'sanakirjamääritelmä': *A person who is gullible and easy to take advantage of*

Polyseemisen (monimerkityksisen) lekseemin merkitykset esitetään kertomalla mihin synsetteihin sana kuuluu, ts. synsetit ovat merkityksen perusyksikköjä. Voidaan ajatella että kukin niistä vastaa jotain *käsitettä* (ajattelun yksikkö, concept).

12.1 Sanojen sisäinen rakenne

Temaattiset roolit

Lähestymistapa verbien argumenttien semanttiseen kuvaamiseen

Esim. mitä yhteistä on verbeillä 'rikkoa' ja 'avata'?

- molempien toiminnan *kohteena* on tyypillisesti jokin **eloton olio**
- molempien toiminnan *tekijänä* on tyypillisesti jokin **elollinen olento** joka **tahtoo aiheuttaa** tietyn tapahtuman (rikkoutuminen, avautuminen), eli kummankin verbin subjektit ovat *agentteja*.

Joitain temaattisia rooleja ja esimerkkejä niistä (Kuva 16.9. Jurafsky & Martin-kirjasta):

Temaattinen rooli	Esimerkki
AGENT	*Tarjoilija* läikytti keittoa.
EXPERIENCER	*Jukalla* on päänsärky.
FORCE	*Hirmumyrsky* repi talojen kattoja irti.
THEME	Vasta Benjamin Franklinin rikottua *jään* ...
RESULT	Ranskan hallits on rakentanut kokomääräysten mukaisen *timanttisen pesäpallon*
CONTENT	Mona kysyi *"Tapaat ilmeisesti Niinan keskustassa?"*
INSTRUMENT	Hän ryhtyi virvelöimään ja nosti saaliit veneeseen *haavi*
BENEFICIARY	Aina kun sihteeri teki pöytävarauksen *pomolleen*, ...
SOURCE	Lensin tänne eilen *Joensuusta*.
GOAL	Ajoin *Jyväskylään*.

Huom. lista ei tietyn teorian mukainen, eikä ainoa mahdollinen.

Temaattisia rooleja voidaan käyttää matalan tason semanttisessa tulkinnassa, esim. taggaamaan niillä sanoja. Esim. jotkut jäsentimet tuottavat myös

tämänkaltaisia semanttisia analyyseja.

Verbien argumentinvalintapreferenssit (selectional preferences)

Useimmat verbit suosivat tiettyyn semanttiseen kategoriaan kuuluvia argumentteja. Tätä ilmiötä kutsutaan nimellä 'selectional preferences' tai 'seletional restrictions' (jos tarkoitetaan kovia sääntöjä).

Esim. *ajatella*-verbin tekijä on tavallisesti ihminen (toisinaan myös eläin tai jokin muu elollinen olento), ei kuitenkaan eloton olio (esim. kirja).

Kuitenkin esim. metaforisessa käytössä voidaan käyttää sanoja toisin kuin yleensä.

Esim. 'syödä'-verbi preferoi tekemisen kohteeksi ruoka-argumentteja. Silti 'hän söi sanansa', 'koira söi kotitehtäväni', 'tuo valitettava tapahtuma söi pääministerin valtaa hallituksessa'.

Voidaan myös ajatella että syödä-verbillä on kaksi erillistä käyttötilannetta,

konkreettinen ruoan tai objeltin syöminen ja abstrakti. Abstrakti merkitys lainaa konkreettiselta joitain prosessin ominaisuuksia, kuten syötävän asian väheneminen ja/tai radikaali muodonmuutos syömisen seurauksena.

Preferenssit antavat tietoa tuntemattomista sanoista

'Susan ei ollut koskaan aiemmin syönyt tuoretta duriania'.

syödä-sanan konkreettisen merkityksen ja siihen liittyvien preferenssien perusteella voisi siis pitää todennäköisenä että 'durian' on ruoka-aine. Vastaavasti 'tuore'-sanan preferenssit viittaavat samaan suuntaan, ja lisäksi siihen että kyse on potentiaalisesti pilaantuvasta ruoka-aineesta.

Eräs menetelmä argumentinvalintapreferenssien estimointiin (Resnik, 1993, 1996)

Tehtävä: selvittää kuinka vahvasti verbi preferoi tiettyjä sanoja (tai sanaluokkia) tietyssä argumentissaan.

Yksinkertaistus 1: Keskitytään tarkastelemaan pelkästään tekemisen kohdeargumentin pääsanaa. Esim. 'Susan söi vihreän **omenan**.' (Jätetään siis huomiotta objektia luonnehtivat tai tarkentavat sanat, tässä 'vihreän').

Yksinkertaistus 2: Yksittäisten substantiivien sijaan tarkastellaan substantiivikategorioita.

Määritellään **valintapreferenssin voimakkuus** $S(v)$ verbille v.

$$S(v) = D(P(C|v)||P(C)) = \sum_C P(c|v) \log \frac{P(c|v)}{P(c)} \qquad (61)$$

$S(v)$ on etäisyys luokkien C priori-ja posteriorijakaumien välillä, laskettuna Kullback-Leibler-divergenssiä käyttäen.

Substantiivikategoriat voidaan tuottaa klusteroimalla tai ottaa jostain valmiista leksikaalisesta resurssista (esim. Wordnet).

$S(v)$ kuvaa sitä kuinka valikoiva verbi v on argumenttiensa substantiivikategorioiden osalta.

Määritellään seuraavaksi **valinta-assosiaation voimakkuus** $A(v, c)$ verbin v ja kategorian c välillä:

$$A(v, c) = \frac{P(c|v) \log \frac{P(c|v)}{P(c)}}{S(v)}$$

Lopuksi, assosiaation voimakkuus verbin v ja substantiivin n välillä on

$$A(v, n) = \max_{c \in classes(n)} A(v, c)$$

(n voi kuulua useaan substantiivikategoriaan mikäli on polyseeminen).

Positiiviset $A : n$ arvot kuvaavat positiivisiä preferenssejä, negatiiviset negatiivisia ja 0 on neutraali. Esim.

$A(eat, \text{food}) = 1.08$
$A(find, \text{action}) = -0.13$

Eli syödä-sana preferoi ruokaobjekteja kun taas löytää-sana karkottaa toimintatyyppisiä objekteja. Ks. myös kirjan taulukko 8.6. Tätä tietoa voidaan käyttää lauseen semanttisen rakenteen hahmottamisessa, tai esim. jäsennysten disambiguoinnissa.

Eräs $S(v)$:n sovellustapa:

$S(v)$ ennustaa englannilla melko hyvin sitä salliiko kyseinen verbi suoran objektin poisjättämisen. Esim. 'Tiina söi omenaa.' ja 'Tiina söi.' mutta ei niin luontevasti 'Tiina etsi.'

12.2 Semanttinen samankaltaisuus

- Yleispätevän semanttisen representaation löytäminen on kieliteknologian tärkeä ja hankala ongelma.
- Riittävää representaatiota joka olisi automaattisesti opittavissa ja hyödyn ei vielä ole kehitetty.
- Huomattavasti helpompaa on tutkia sanojen semanttista samankaltaisuutta.
- Samankaltaisuutta voidaan hyödyntää esim. ennennäkemättömiin tapauksiin yleistämisessä.

Samankaltaisuuksiin perustuva yleistäminen

Esimerkki: Voiko 'durian' olla syödä-verbin kohde?

Oletetaan että ei tiedetä mitä 'durian' on, mutta tiedetään että sen kanssa samankaltaisimmat sanat ovat 'omena', 'banaani' ja 'mango'. Koska kaikki ovat syödä-verbin kohteeksi kelpaavia, voidaan olettaa että myös 'durian' on.

Samankaltaisuuteen perustuva yleistäminen on sukua luokkatietoon perustuvalle yleistämiselle.

Erona on se, että ei ole mitään tiettyä joukkoa viiteryhmiä joiden sisällä yleistäminen tapahtuu, vaan kulloinenkin viiteryhmä ovat kyseenäolevan yksilön kanssa samankaltaisimmat yksilöt.

K:n lähimmän naapurin luokitusmenetelmä (KNN):

Luokitellaan tuntematon näyte sen perusteella mitkä ovat k:n sitä lähimpänä olevan luokitellun näytteen luokat.

Kontekstuaalinen vaihdettavuus semanttisen samankaltaisuuden kriteerinä

Miller & Charles (1991): sanojen semanttisen samankaltaisuuden arviot korreloivat vahvasti sanojen vaihdettavuuden kanssa tietyssä kontekstissa.

Monitulkintaisuuden tuoma ongelma: Semanttinen samankaltaisuus on tavallisesti *tiettyjen sananmerkitysten* välinen relaatio, ei polyseemisten sanojen kaikkien merkitysten välinen.

12.3 Vektorietäisyyksiin perustuvat mitat

Dokumentti-sana-matriisissa alkion i, j arvo on 1 jos sana j esiintyy dokumentissa i. 1:n sijaan voidaan käyttää myös sanan lukumäärää.

Matriisin rivit ovat tällöin *dokumenttivektoreita sana-avaruudessa* ja sarakkeet *sanavektoreita dokumenttiavaruudessa*.

Sana-sana-matriisissa raportoidaan kuinka monta kertaa sana j esiintyi sanan i kontekstissa.

Konteksi voidaan määritellä eri tavoin, esim. koko dokumentti, kappale, lause, tai jonkin levyinen sanaikkuna.

Tai voidaan kerätä esim. substantiivien osalta tietoja siitä mitkä sanat niitä määrittävät (modifier-head-relaatio, määrittäviä sanoja esim. pieni, juokseva,...). Kaksi substantiivia ovat tällöin samankaltaisia siinä määrin kuin niitä määrittävät samat sanat. Kaksi määrittävää sanaa taas ovat samankaltaisia siinä määrin kuin ne määrittävät samoja substantiiveja.

[Esimerkejä, ks. kirjan kuvat 8.3-8.5.]

Dokumentti-sana-matriisin pohjalta laskettaessa esim. sanat 'astronautti', 'kosmonautti' ja 'kuu' voivat olla samankaltaisia, koska ne kaikki liittyvät avaruusmatkoihin, eli ovat *aihepiiriltään* samankaltaisia.

Sen sijaan katsottaessa lyhyttä kontekstia, tai vielä tarkemmin, tiettyä relaatiota, saadaan tulokseksi toisenlaisia samankaltaisuuksia.

Tällöin 'astronautti' ja 'kosmonautti' olisivat edelleen keskenään samankaltaisia, mutta eivät 'kuun' kanssa—ilmiselvästi koska kuu on taivaankappale ja edellämainitut ihmisiä, joten niitä eivät tyypillisesti määritä samat sanat.

Reaalilukuvektorien väliset samankaltaisuusmitat

n-ulotteiset reaalilukuvektorit $\mathbf{x}, \mathbf{y} \in R^n$.

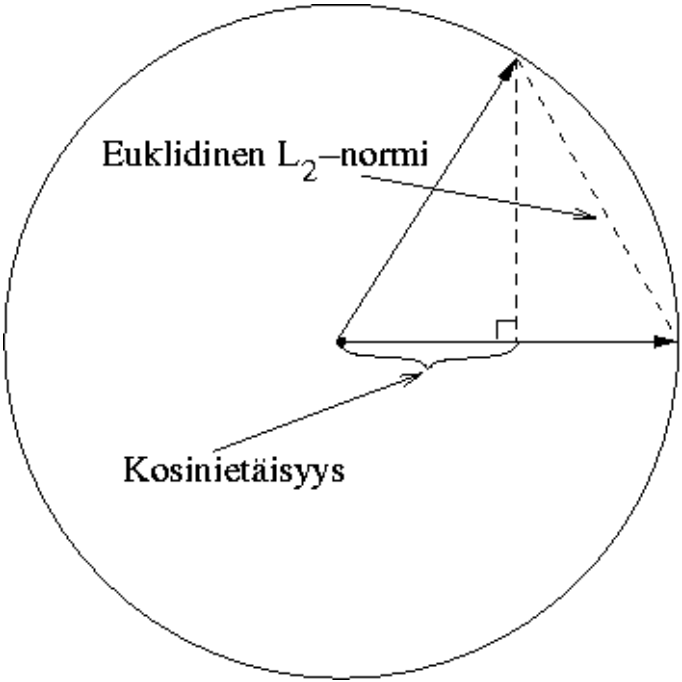
Vektorin pituus on $|\mathbf{x}| = \sqrt{\sum_{i=1}^{n} x_i^2}$

Kosinietäisyys: $cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}||\mathbf{y}|}$, jossa $\mathbf{x} \cdot \mathbf{y}$ tarkoittaa pistetuloa, eli sisätuloa, eli elementeittäin kertomista.

Huom: normalisoiduille (yksikkövektorin pituisille) vektoreille kosinietäisyys on pelkästään vektorien pistetulo. Normalisointi tarkoittaa siis vektorin itsensä pituudella jakamista

Euklidinen L_2-etäisyys: $|\mathbf{x} - \mathbf{y}| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$

Huom: Jos vektorit ovat normalisoituja, ja jos samankaltaisuusmittaa käytetään pelkästään sanojen (tai dokumenttien) järjestämiseen samankaltaisuusjärjestyksi saadaan Euklidisella ja kosinietäisyydellä aikaan sama järjestys.


Euklidinen L_2–normi
Kosinietäisyys

Joitain tuloksia

Ks. kirjan taulukko 8.8. Jos konteksti on suhteellisen pitkä (esim. 25 sanaa kumpaankin suuntaan), tyypillisiä tuloksia:

- joillekin, etenkin keskiharvinaisille sanoille hyviä tuloksia
- aihepiirisamankaltaisuudet keskeisessä roolissa (koska konteksti on melko pitkä)
- sanan eri sananmuodot (esim. verbin preesens ja imperfekti) keskenään samankaltaisia
- ei tunnisteta sanan syntaktista tai semanttista roolia (koska konteksti niin pitkä, ja koska sanojen järjestyksellä ei merkitystä)
- yleisille sanoille, esim. 'simple', saadaan hyvin sekalaisia tuloksia (koska voivat esiintyä hyvin monenlaisissa pidemmissä ts. aihepiirikonteksteissa)

Hyvin lyhyttä kontekstia käytettäessä, esim. vain edeltävä ja seuraava sana, korostuvat puolestaan erityyppiset samankaltaisuudet, esim. sanan syntaktinen ja/tai semanttinen rooli.

Lopuksi: Saadut samankaltaisuudet ovat aina *erittäin* voimakkaasti riippuvaisia valitusta korpuksesta. Ei siis saada yleispäteviä tuloksia.

Vektorietäisyysmittojen hyviä puolia

- Yksinkertaisia ja intuitiivisia, saatu lukuarvo on helposti samaistettavissa samankaltaisuuden asteeseen.
- Mitoilla laskeminen on hyvin nopeaa (etenkin pistetulo, eli kosini normalisoiduilla vektoreilla)
- Mittoja on sovellettu menestyksekkäästi ja pitkään mm. tiedonhaussa.
- Havaittu samankaltaisuutta psykologisen priming-efektin kanssa.

Priming-efekti

Psykologiassa tutkittava ilmiö: sanan havaitsemisen nopeuteen vaikuttaa se, onko lähihistoriassa havaittu ko. sanan kanssa semanttisesti samankaltaista sanaa: jos lähihistoriassa havaittiin esim. sana 'hoitaja' myöhemmin sanan 'lääkäri' havaitseminen tapahtuu nopeammin.

Tutkimalla priming-efektin vahvuutta sanaparien välillä saadaan niille psykologinen samankaltaisuusmitta, jota voidaan periaatteessa käyttää verrokkina yritettäessä kehittää sanojen samankaltaisuus tekstiaineistojen perusteella.

Tällä tavoin saatava tieto on kuitenkin yleensä hyvin työlästä ja hidasta tuottaa, joten tuloksena olevat samankaltaisuusmatriisit ovat hyvin harvoja.

Vektorimittojen huonoja puolia

- Euklidisen metriikan olettaminen ei ole matemaattisesti optimaalista kun kyse on pohjimmiltaan *lukumäärätiedosta*.
- Esim. etäisyys 0 ja 10 esiintymän välillä on yhtä suuri kuin 990 ja 1000 esiintymän välillä. Jälkimmäinen ero on kuitenkin huomattavasti vähämerkityksisempi.
- Esim. kosinietäisyys sopii hyvin normaalijakautuneille suureille, mutta ei niin hyvin vinoille jakaumille, kuten lukumäärien jakaumille.
- Mitoilla ei selkeää todennäköisyystulkintaa.

12.4 Todennäköisyyksiin perustuvat mitat

Yhteisesiintymämatriisista saadaan ehdollisten todennäköisyyksien matriisi (tai siis sen ML-estimaatti) jakamalla kunkin alkion lukumäärä ko. rivin alkioiden summalla.

Semanttista samankaltaisuutta (tai pikemminkin erikaltaisuutta) mitataan tämän jälkeen tutkimalla kahden tn-jakauman välistä eroa:

Mitan nimi	Määritelmä
KL divergenssi	$D(p\|\|q) = \sum_i p_i \log \frac{p_i}{q_i}$
Informaatioradius	$D(p\|\|\frac{p+q}{2}) + D(q\|\|\frac{p+q}{2})$
L_1 normi	$\sum_i \|p_i - q_i\|$

(Olet. $0 \log 0 = 0$)

Kullback-Leibler-divergenssi, KL-divergenssi

- Tulkinta: Mittaa paljonko informaatiota kadotetaan jos oletetaan jakauma q kun todellinen jakauma on p
- Ongelma 1: vastauksena ∞ jos on yksikin dimensio jossa $q_i = 0$ ja $p_i \neq$

0 (näin käy usein jos käytetään ML-estimaatteja todennäköisyyksille)

- Ongelma 2: epäsymmetrinen, siis ei ole *metriikka*

Informaatioradius (IRad)

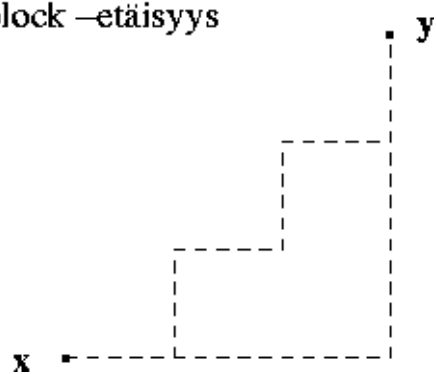
- Tulkinta: Paljonko informaatiota kadotetaan jos kuvataan sekä p että q niiden keskiarvojakaumalla?
- Ei edellisen mitan ongelmia (on symmetrinen, ei ääretöntä arvoa jos vain jompikumpi tn on nolla)

(Kuitenkin: jos oikeasti haluttaisiin ilmaista että on joitain mahdottomia asioita, niin niitä vastaisi etäisyyksissä ääretön).

L_1 normi eli Manhattan-etäisyys eli city-block-etäisyys

- Tulkinta: Toisistaan poikkeavien tapahtumien suhteellisen frekvenssin odotusarvo.

L1–normi eli Manhattan eli
city–block –etäisyys



Loppuhuomioita

- Tn-pohjaisilla mitoilla on selkeä jakaumatulkinta
- Saadut erikaltaisuusmitat täytyy kuitenkin erikseen muuntaa samankaltaisuusmitoiksi (ei aivan suoraviivaista, sisältää parametrin optimoinnin)
- Sekä tn-pohjaiset että vektoripohjaiset mitat hyödyllisiä leksikaalisen tiedon keräämisessä.

13. Tiedonhaku

Information retrieval, text retrieval

Tiedonhaussa tehtävänä on hakea käyttäjän tiedontarvetta vastaavaa tietoa suurista dokumenttikokoelmista.

Ongelmaa tutkittu vuosikymmenet erillään NLP-tutkimuksesta, johtuen erilaisista käytetyist menetelmistä. Nykyisin lähentymistä, koska myös NLP:ssä tilastolliset menetelmät valtaavat alaa.

Ad hoc retrieval - käyttäjä kirjoittaa hakulausekkeen ja systeemi vastaa palauttamalla joukon dokumentteja, joiden on tarkoitus vastata tiedontarpeeseen.

Kaksi pääsuuntaa: *exact match* ja *ranking*.

Exact match- -retrieval –täsmälliset osumat

Hakukriteerit määrittelevät täsmällisiä haettavia ominaisuuksia, ja vastauksena annetaan dokumentit jotka täyttävät nämä kriteerit täsmälleen.

Tämä hakutyyppi on käytössä monissa vanhemmissa tietokannoissa [esim. kirjastojen cdrom-tietokannat]

Tunnetuin alalaji: Boolen haut, joissa haettavan dokumentin kriteerit yhdistetään Boolen logiikan avulla.

Lähestymistapa toimii kohtuullisesti pienillä ja homogeenisilla kokoelmilla, kokeneen hakijan käytössä.

Ongelmia etenkin suurilla ja heterogeenisilla kokoelmilla:

- Tuloksena voi olla tyhjä joukko tai valtava määrä osumia – ei voi tietää ennalta.
- Käyttäjän on hyvin vaikea rajata haku siten että saisi juuri haluamansa dokumentit, mutta mahdollisimman vähän roskaa. Tiedontarve kun

harvoin ilmenee täsmällisinä luonnollisen kielen lausekkeina.

- Saman sisällön voi ilmaista monella eri tavalla — täsmällisen haun systeemeissä pitäisi nämä kaikki tavat ilmaista täsmällisesti, ja toisilleen vaihtoehtoina.
- Haun tulokset eivät ilmesty paremmuusjärjestyksessä (koska kaikki ovat yhtä hyviä).
- Ei tiedetä paljonko ja minkälaisia 'lähes yhtä hyviä' dokumentteja oli.

Ranking – Järjestetyt osumat

Täsmällisen osumajoukon palauttamisen sijaan järjestetään kaikki dokumentit paremmuusjärjestykseen sen mukaan miten hyvin ne vastaavat hakulausekketta. [Esim. Altavista.]

Lähestymistapoja esim. probabilistinen haku, ja johonkin samankaltaisuusmittaan perustuva haku.

Nykyään täsmähakua yleisempi hakujärjestelmätyyppi.

Sanojen selityksiä

query: hakusana, hakulause, hakulauseke. Se millä haetaan.

termi, indeksointitermi: Sanastoon kuuluva sana, siis osa dokumenttien representaatiota. Kaikki sanat eivät ole termejä. Termien ei edes tarvitse välttämättä olla sanoja: ne olla myös sanojen alkuosia (esim. 5 ensimmäistä kirjainta) tai sanojen perusmuotoistettuja muotoja. Termeihin voi kuulua myös geneerisiä koodeja, esim. URL tai PUHNO jotka voivat olla esikäsittelyohjelmilla alkuperäisistä dokumenteista prosessoituja.

Relevanssi, relevance: vastaavuus hakulauseen (tai sen tarkoituksen) kanssa.

Relevance feedback: tapa jolla käyttäjä voi interaktiivisesti tarkentaa ja uudelleenkohdentaa hakuaan, antamalla palautetta siitä kuinka hyviä systeemin antamat dokumentit olivat. Ad-hoc-hauissa eräs tutkimuskohde.

sudoatus (filtering), reititys (routing): tekstinkategorisoinnin erikoistapaus;

kategorisoidaan dokumentit relevantteihin ja ei-relevantteihin.

Lisätietoa tiedonhausta: Modern Information retrieval (Baeza-Yates & Ribeiro-Neto, 1999)

13.1 Tiedonhakujärjestelmien perusosia

Käänteisindeksi, inverted index

Osoittimet sanoista dokumentteihin, sekä frekvenssit dokumenteissa. Joskus myös osoittimet tekstipositioihin dokumentissa.

Estolista (stop list)

Lista sanoista joiden indeksointi estetään, yleensä aineistoriippumaton.

Listaan valitaan sanoja joita pidetään hakujen kannalta hyödyttöminä tai häiritsevinä. Esim. kieliopilliset tai funktiosanat, mm. suljettujen sanaluokkien sanat kuten pronominit.

Voi sisältää myös muita yleisiä, indeksoinnin kannalta melko tyhjiä sanoja (esim. apuverbit ja muut yleisimmät verbit kuten 'mennä', 'tulla')

Osuu jossain määrin päällekkäin yleisimpien sanojen listan kanssa.

Estolista vähentää merkittävästi indeksin kokoa, koska monet estettävistä sanoista yleisiä.

Huono puoli on että estolistalla olevat sanoilla ei voi hakea, esim. 'milloin ja missä' sisältää pelkästään estolista-sanoja.

Stemming (juureksi palautus) tai perusmuotoistaminen

Stemming on approksimaatio morfologisele analyysille. Siinä poistetaan sanoista päätteiksi katsotut pätkät, tarkoituksena saada pelkkä sananvartalo. Vartaloita käytetään indeksointitermeinä.

Esimerkkejä mahdollisista vartaloista ja sananmuodoista:

Vartalo	Vartalon sananmuotoja
laugh-	laughing, laugh, laughs, laughed
gall-	gallery, galleries (ongelma: gall)
etsi-	etsiskellä, etsittiin, etsin
yö-	yöllinen, yötön, yöllä
öi-	öisin, öinen
aika-	aikana, aikaan, aikaa
aj-	ajallaan, ajaton, ajat, ajoissa
ajat-	ajatella, ajatus (ongelma: vrt. ed.)

Kuten esimerkeistä näkyy, stemming on rankasti yksinkertaistava ratkaisu, ja sopii huonosti esim. suomelle.

Yhdellä perusmuodolla voi olla useita eri hakuvartaloita.

Vartalon katkaisukohdan valinta on jossain määrin mielivaltainen kompromissi spesifiyden ja kattavuuden välillä.

Tavallisia stemmereitä englannille ovat Porter ja Lovins. Suomen perusmuotoistus mm. TWOLilla (Koskenniemen 2-tasomalli morfologialle).

13.2 Hakumenetelmien evaluointimittoja

$N =$ dokumenttimäärä, joka hakujärjestelmää pyydettiin palauttamaan
$REL =$ tälle haulle relevanttien kokonaismäärä dokumenttikokoelmassa
$rel =$ tälle haulle relevanttien lukumäärä palautetussa dokumenttijoukossa

Tarkkuus ja saanti

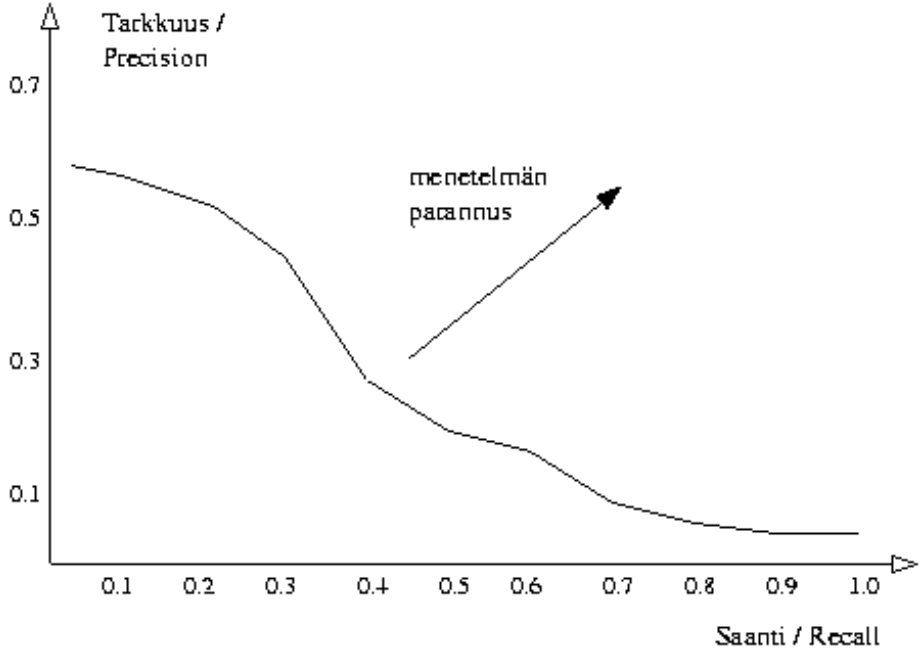
Perusmitat hakujärjestelmien evaluoinnissa.

Tarkkuus l. *precision* P: Relevanttien osuus vastaukseksi saaduista dokumenteista, $P = rel/N$

Saanti l. *recall* R: Vastaukseksi saatujen relevanttien osuus kaikista relevanteista, $R = rel/REL$.

Kun palautettavien lukumäärä nousee, yleensä tarkkuus laskee ja saanti kasvaa.

Tarkkuus-saanti-käyrä:



Esimerkki soveltamisesta menetelmien vertailuun
(%= relevantti, x=epärelevantti dokumentti):

Mitta	Menetelmä 1	Menetelmä 2	Menetelmä 3
	d1: %	d10: x	d6: x
	d2: %	d9: x	d1: %
	d3: %	d8: x	d2: %
	d4: %	d7: x	d10: x
	d5: %	d6: x	d9: x
	d6: x	d5: %	d3: %
	d7: x	d4: %	d5: %
	d8: x	d3: %	d4: %
	d9: x	d2: %	d7: x
	d10: x	d1: %	d8: x
Tarkkuus kun n=5	1.0	0.0	0.4
Tarkkuus kun n=10	0.5	0.5	0.5
Interpoloimaton tarkk.	1.0	0.3544	0.5726
Interpoloitu (11-pist.)	1.0	0.5	0.6440

Jos ajattelee lukevansa hakukoneen palauttamaa listaa ylhäältä alkaen, menetelmä 1 on näistä selvästi paras. Kuitenkin tarkkuus 10 dokumentin kohdalla on niille sama.

Huonoa: tarkkuus ja saanti eivät huomioi tulevatko oikeat osumat alku- vai loppupäässä. Siksi myös muita mittoja:

Un-interpolated average precision (interpoloimaton keskimääräinen tarkkuus)

Kerää useita tarkkuuslukuja yhteen mittaan. Tarkkuus mitataan *aina kun systeemi palauttaa relevantin dokumentin*. Näin saadut luvut keskiarvoistetaan. Relevantit dokumentit, joita ei palautettu, lasketaan mukaan tarkkuudella 0.

Esim. Menetelmälle 3: $1/2 + 2/3 + 3/6 + 4/7 + 5/8 = 0.5726$

(mikäli 10 ekan palautetun joukossa olivat kaikki relevantit dokumentit).

Interpolated average precision (interpoloitu keskimääräinen tarkkuus)

Eroina edelliseen:

1. Tarkkuudet lasketaan tietyillä saantitasoilla (tavallisesti 10% välein 0%:sta alkaen).

2. Mikäli tarkkuus jossain vaiheessa kohoaa, kaikkien aiempien lukujen tarkkuuksiksi otetaan tämä uusin, korkeampi luku.

F-mitta

Toinen tapa mitata tarkkuutta ja saantia yhtäaikaa, yhdellä mitalla:

$$F = \frac{1}{\alpha \frac{1}{P} + (1-\alpha)\frac{1}{R}} \tag{62}$$

jossa R on recall (saanti) ja P precision(tarkkuus).

Voidaan käyttää evaluoimaan menetelmiä kun palautettavien dokumenttien lukumäärä on kiinnitetty ja halutaan huomioida sekä tarkkuus että saanti.

Menetelmien vertailu

Yleensä luvut keskiarvoistetaan useiden hakujen (esim. 50) yli, ja verrataan menetelmien saamia keskiarvoja.

Lisäksi pitäisi tehdä tilastollinen testi (esim. t-testi) jolla varmistetaan havaittujen erojen tilastollinen merkitsevyys.

TREC: Text retrieval competition: Kansainvälinen vuosittainen tiedonhaun kilpailu jossa eri sarjoja (esim. monikielinen tiedonhaku).

Aluksi jaetaan aineisto jolla menetelmänsä hyvyyttä voi tutkia ja optimoida menetelmää

Testiaineisto annetaan sokkona, eli kilpailijat eivät saa tietää oikeita vastauksia (ts. mitkä dokumentit ovat relevantteja millekin haulle).

Lopuksi julkaistaan relevanssitiedot kullekin dokumentille, sekä lasketaan kunkin menetelmän hyvyydet keskitetysti samoilla mittareilla.

Ongelmanasettelun ja evaluoinnin ongelmallisuudesta

Edellä esitetyn evaluoinnin taustalla on periaate:

Probability ranking principle (PRP): On optimaalista järjestää dokumentit niiden relevanssin todennäköisyyden mukaan, ts. relevanteimmiksi estimoidut ensin.

Poikkeuksia/ongelmia:

PRP olettaa että dokumentit ovat riippumattomia, mutta todellisuudessa näin ei ole.

Esim. duplikaatit, tai dokumentit jotka muuten toistavat päällekkäistä informaatiota jonkin edellisen kanssa: Käyttäjä ei ehkä halua lukea samaa asiaa monesta lähteestä, vaan pikemminkin saada kattavan kuvan hyvistä hakua vastaavista dokumenteista.

PRP olettaa että peräkkäisten hakujen sarjassa haut ovat toisistaan riippumattomia, ts. että kyse on yksittäisistä toisiinsa liittymättömistä kysymys-

vastaus-pareista.

Kuitenkin parhaimmillaan kyse on pikemminkin dialogista, jonka aikana kyselijän tiedon tarve tarkentuu, laajentuu tai uudelleenkohdistuu ymmärryksen kasvaessa. Peräkkäiset haut ovat siis toisistaan riippuvia.

13.3 Vektoriavaruusmalli

Vector space model, VSM

- Yleisesti käytetty, ad-hoc-retrievalin standardimenetelmä.
- Dokumentti esitetään vektorina, jonka dimensioita ovat sanaston sanat (indeksointitermit), ja dimension arvona jokin funktio termin frekvenssistä dokumentissa ja sen painosta, joka ei riipu tästä dokumentista
- Dokumentit ja hakulause esitetään samassa vektoriavaruudessa. Hakua lähimpänä olevat dokumentit palautetaan.
- Etäisyydet lasketaan tyypillisesti kosinietäisyyksinä (ks. edellisen luennon kohta 'Vektorietäisyyksiin perustuvat mitat'), joskus myös Euklidisena etäisyytenä.

Termien painotusmenetelmä: tf.idf

tf: term frequency
idf: inverse document frequency

IDF yhdistää termin lokaalin merkittävyyden, eli esiintymistiheyden tässä dokumentissa sekä termin globaalin merkittävyyden, eli esiintymistiheyden koko aineistossa tai dokumenteissa.

Notaatio:
$tf_{t,d}$ = termin w_t lukumäärä dokumentissa d
df_t = niiden dokumenttien lukumäärä joissa termi w_t esiintyy
cf_t = termin frekvenssi koko kokoelmassa

Näiden huomioiminen voidaan tehdä monella eri tavalla. Eräs vaihtoehto:

$$w(i,j) = (1 + \log(tf_{t,d})) \log \frac{N}{df_t} \tag{63}$$

jossa N on dokumenttien lukumäärä kokoelmassa.

Eri tf.idf-komponentteja taulukossa:

termifrekvenssi	dokumenttifrekvenssi	normalisointi
n (natural) $tf_{t,d}$	n (natural) df_t	n (none)
l (logarithm) $1 + \log tf_{t,d}$	t $\log \frac{N}{df_t}$	c (cosine)
a (augmented) $0.5 + \frac{0.5tft,d}{max_t(tf_{t,d})}$		

Muita harvinaisempia painotusmenetelmiä esitellään kirjan luvussa 15.3 (ei käsitellä tarkemmin kurssilla):

RIDF, K-mikstuurit, kahden Poisson-jakauman menetelmä.

13.4 Latenttien muuttujien menetelmät

- Aiemmin hyödynnetty dokumentin representoinnissa vain tietoja yksittäisten sanojen esiintymistä
- Ongelma: Ei käytä minkäänlaista tietoa sanojen semanttisesta samankaltaisuudesta (kahden sanan keskinäinen etäisyys oletetaan samaksi, sanoista riippumatta)
- Ratkaisu: Jos voidaan projisoida sanat ja dokumentit jonkinlaiseen *latenttien semanttisten piirteiden avaruuteen* ja suorittaa etäisyyslaskenta siellä.
- Hyödynnetään semanttisen avaruuden muodostamisessa sanojen yhteisesiintymätietoja. Esim. jos sanat 'HCI', 'vuorovaikutus', 'käyttäjä' ja 'käyttöliittymä' esiintyvät poikkeuksellisen usein yhdessä (tässä: samoissa dokumenteissa), voidaan olettaa että ne liittyvät semanttisesti toisiinsa.

Latent Semantic Indexing-menetelmä (LSI)

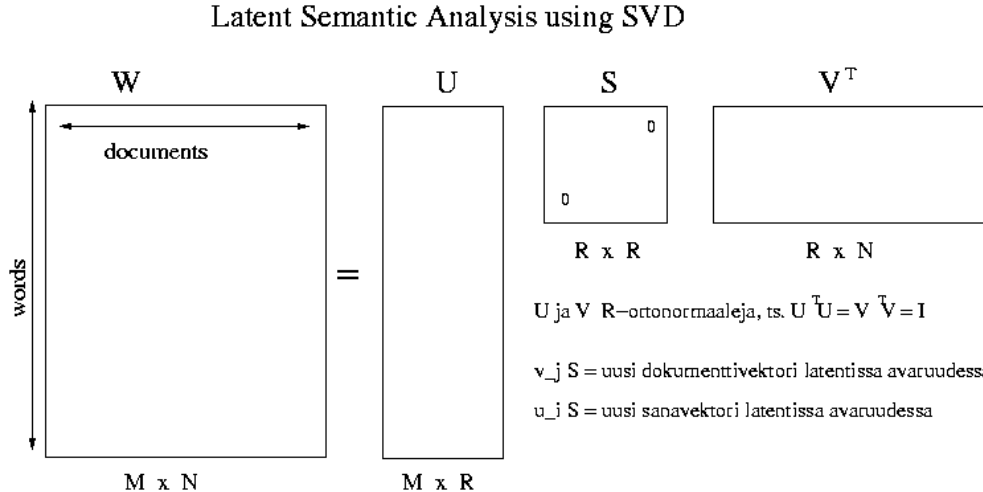
Perusajatus: tehdään sana-dokumenttimatriisille singulaariarvohajotelma eli SVD (Singular Value Decomposition), ja otetaan lopputulokseen mukaan vain avaruuden R merkitsevintä dimensiota.

Lähtökohta: W eli dokumentti-sana-yhteisesiintymämatriisi, jonka alkiot ovat jokin funktio sanan lukumäärästä dokumentissa. Esim.

$w_{i,j} = (1 - \epsilon_i \frac{c_{i,j}}{n_j})$, jossa $c_{i,j}$ on sanan i määrä dokumentissa j, n_j on dokumentin j sanojen kokonaismäärä, ja $epsilon_i$ on sanan i normalisoitu entropia koko korpuksessa. Myös tf.idf-painotuksia voidaan käyttää.

Lasketaan R:n asteen SVD(W): $(\hat{W}) = USV^T$, jossa
S on diagonaalimatriisi jonka diagonaalissa singulaariarvot ja U ja V tarvitaan sanojen ja dokumenttien projisointiin latenttiin avaruuteen. (T tarkoittaa matriisin transpoosia).

SVD laskee optimaalisen R-ulotteisen approksimaation W:lle.



R:n arvoksi suositellaan 100-200.

Tulkinta

LSI esittää dokumentin sisällön semanttisten piilomuuttujien ('abstraktien käsitteiden') lineaarikombinaationa (summana). Piilomuuttujia on R kappaletta.

Samankaltaisissa dokumenttiympäristöissä esiintyneet sanat saavat samankaltaisen latentin representaation.

Projektioita latenttiin, semanttiseen avaruuteen voidaan soveltaa mm. tiedonhakuun, dokumenttien klusterointiin, ja sanojen klusterointiin.

Kritiikkiä: SVD optimoi representaation neliöllisen virheen mielessä (least-squares, L_2-normi). Tämä implikoi oletuksen että yhteisesiintymät ovat normaalijakautuneita latenteilla dimensioilla, mikä ei välttämättä pidä paikkaansa kielidatalla.

On olemassa variantti jossa L_2-normin sijaan käytetään etäisyysmittana KL-divergenssiä.

Riippumattomien komponenttien analyysi

Vastaavalla tavalla kuin LSA voidaan sana-dokumenttimatriisille laskea toinen muunnos, nimittäin ICA, Independent component analysis eli riippumattomien komponenttien analyysi.

Erona edelliseen on, että nyt etsitään latentit muuttujat (projektiosuunnat) jotka ovat mahdollisimman *riippumattomia* toisistaan jonkin tietyn jakaumien riippumattomuutta mittaavan mitan mielessä (esim. kurtoosi).

ICA:aa, mukaanlukien sen soveltaminen keskustelujen analyysiin, tutkitaan mm. TKK:lla Neuroverkkojen tutkimusyksikössä.

13.5 Dimension pienennys

Vektoriavaruusmallissa vektorien dimensio on sanaston koko, eli valtava.

Vielä satojen tuhansien dokumenttien aineistossa sanasto voi olla yhtä suuri kuin dokumenttien määräkin—uudet dokumentit tuovat yhä uutta sanastoa.

Sanaston määrää kuitenkin pienentävät seuraavat esikäsittelyn toimet:

- stoplistan käyttö (pieni vaikutus)
- harvinaisten sanojen karsiminen (huomattava vaikutus, koska suuri osa sanaston sanoista on harvinaisia, ks. Zipf'in laki)
- sanojen perusmuotoistaminen (mutta suuri osa kohdatuista sanoista on lingvististä tietoa käyttävälle perusmuotoistavalle mallille aina tuntemattomia, oten auttaa vain osaksi) TAI
- sanojen katkaisu 'juurimuotoihin'

Edellämainittujen jälkeenkin sanasto voi kuitenkin helposti sisältää kymmeniä tuhansia sanoja (indeksointitermejä).

Tämänkokoisten representaatioiden käsittely ja tallennus on ongelmallista sekä tilankäytön että prosessointiajan kannalta.

Mikäli vektoreita halutaan lisäksi ryhmitellä tai luokitella, monet oppivat menetelmät joiden kompleksisuus on vahvasti sidoksissa datan dimensioon, ovat vaikeuksissa.

Seuraavilla menetelmillä voi dimensiota pienentää edelleen:

- LSI (on käytetty dokumenttien dimension pienennykseen)
- ICA (dimension pienennys on sivuvaikutus, mutta siis ainakin periaatteessa sovellettavissa)
- Satunnaisprojektio (on käytetty dokumenttien dimension pienennykseen)

Näinollen esim. LSI:n käyttöä voi perustella pelkästään dimension pienennysnäkökulmasta, välittämättä siitä parantuuko dokumenttien semanttinen kuvaus vai ei.

Satunnaisprojektio

Satunnaisprojektiossa otetaan tietyllä tavalla muodostettu satunnaismatriisi jota käytetään datavektorien projisointiin pienempiulotteiseen avaruuteen.

$\mathbf{n}_i$ - alkuperäinen dokumenttivektori dokumentille i
$\mathbf{R}$ - satunnaismatriisi jonka kolumnit ovat normaalijakautuneita yksikkövektore
Dimensionaalisuus on $rdim \times ddim$, $ddim$ on alkuperäinen dimensio ja $rdim$ uusi, $rdim << ddim$
$\mathbf{x}_i$ - uusi, satunnaisprojisoitu dokumenttivektori dokumentille i, vektorin dimensio $rdim$.

Tällöin projisoidut dokumenttivektorit saadaan seuraavasti:

$$\mathbf{x}_i = \mathbf{R}\mathbf{n}_i \; . \tag{64}$$

Dimension pienennyksessä on oleellista että projektion yksikkövektorit ovat mahdollisimman ortogonaalisia (ts. korrelaatiot vektorien välillä ovat mahdollisimman pieniä). $\mathbf{R}$:n kohdalla vektorit eivät ole täysin ortogonaalisia, mutta mikäli $rdim$ on riittävän suuri, ja vektorit on poimittu satunnaisesti

hyper-yksikköympyrän tasajakaumasta, keskimääräiset korrelaatiot ovat hyvin pieniä.

$rdim$:lle tyypillisesti käytetyt arvot ovat luokkaa 100-1000.

[Lisätietoja: Ks. esim. Kohonen et al, in Transactions on Neural Networks, 2000,]

14. Klusterointi

(Lähteet: kirjan luku 14 ja kurssin T-61.231 'Hahmontunnistuksen perusteet' kalvot)

Ohjaamattomassa oppimisessa tyypillinen tehtävä on klusterointi, jossa pyritään muodostamaan havainnoista rykelmiä (klustereita), joiden sisällä havainnot ovat keskenään jollain lailla samankaltaisia ja klustereiden välillä erikaltaisia.

Havaintoihin ei liity mitään luokkatietoa. Myöskään klustereiden lukumäärää ei välttämättä tiedetä ennalta.

Klusterointia voidaan käyttää myös silloin kun jotain luokkatietoa on annettu, mutta halutaan selvittää tarkemmin luokkien sisäinen rakenne.

Klusterointialgoritmeja voidaan jakaa ryhmiin eri tavoin, mm.

- parametriset (EM, mikstuurimallit) vs. epäparametriset
- hierarkkiset (esim. single-link clustering) vs. litteät (esim. K-means)
- kovan vs. pehmeän klusteroinnin tekevät menetelmät

Parametriset algoritmit yrittävät samanaikaisesti mallittaa näytteiden kuuluminen klustereihin sekä klusterien todennäköisyysjakauman parametrit piirreavaruudessa.

Epäparametriset algoritmit jakavat havainnot eri luokkia vastaaviksi osajoukoiksi, luokkien tn-jakaumia ei estimoida.

Ensimmäiset ovat tilastollisesti perustellumpia, kun taas jälkimmäiset voidaan usein toteuttaa yksinkertaisesti ja tehokkaasti.

Klusterointiongelman ratkaiseminen voidaan jakaa seuraaviin vaiheisiin:

- Piirteiden valinta (esikäsittely & normalisointi!)
- Samankaltaisuus/erilaisuusmitan valinta havaintoparien, havaintojen ja klustereiden, sekä klusteriparien välille (piirteiden samanarvoinen kohtelu!)
- Klusterikriteerin valinta
- Klusterointialgoritmin valinta
- Tulosten validointi erilaisten testien avulla
- Tulosten tulkinta

Huom! Saatu ratkaisu on erittäin subjektiivinen, koska sama havaintojoukko voidaan jakaa hyvin erilaisiin klustereihin riippuen em. valinnoista. Yleensä ratkaisun tulkinnan ja sen hyvyyden arvioinnin suorittaa sovellusalan asiantuntija

14.1 Erilaisia klusterin määritelmiä

- *Luonnolliset* klusterit ('natural clusters'): piirreavaruuden alueita, joissa havainnot ovat suhteellisen tiheässä ja joiden välissä havainnot ovat suhteellisen harvassa.
- Yksikäsitteiset eli *kovat* klusterit ('hard','crisp'): näyte kuuluu kokonaan yhteen luokkaan.
  NLP:n kannalta kova klusterointi voi olla epätoivottava, johtuen esim. monitulkintaisuus-ilmiön yleisyydestä.
- Pehmeät eli *sumeat* ('fuzzy') klusterit: havainnon kuulumisasteet eri klustereihin ilmaistaan jäsenyysfunktioiden ('membership functions') avulla.
- *Todennäköisyyksiin* perustuvat klusterit: havainto $\mathbf{x}$ kuuluu siihen klusteriin C_k, jonka *a posteriori* tn on korkein eli $P(C_k|\mathbf{x}) \geq P(C_i|\mathbf{x})\ \forall i = 1, \ldots, m$

Lisäksi *disjunktiivisessa* klusterimallissa näyte voi samanaikaisesti kuulua ai-

dosti moneen luokkaan. Kuitenkin tällöin joudutaan määrittelemään lisäksi kriteerit joista voidaan päätellä milloin joku näyte kuuluu moneen eri klusteriin, milloin vain yhteen.

14.2 Hierarkkinen klusterointi

Hierarkkiset klusterointialgoritmit muodostavat havainnoille klusterointiratkaisujen sarjan, joilla on selkeä sisäkkäinen ('nested') rakenne.

Algoritmit voidaan jakaa kahteen alaluokkaan: kasaaviin ('agglomerative') ja pilkkoviin ('divisive') algoritmeihin.

Kasaavat algoritmit muodostavat klusterointiratkaisujen sarjan yhdistelemällä klustereita. Algoritmi lähtee liikkeelle tilanteesta, jossa jokainen havainto vastaa yhtä klusteria ($m = N$). Algoritmin edetessä klustereiden lkm pienenee, kunnes kaikki havainnot kuuluvat yhteen klusteriin ($m = 1$).

Pilkkovat algoritmit toimivat päinvastoin. Aluksi kaikki havainnot kuuluvat yhteen klusteriin ($m = 1$) ja algoritmin edetessä klustereita jaetaan kahtia, kunnes jokaisessa klusterissa on tasan yksi havainto ($m = N$).

Hierarkkisen klusteroinnin tulokset voidaan esittää *dendrogrammina* jossa samaan klusteriin kuuluvat havainnot on kytketty viivalla tietyllä korkeudella.

Jokainen kytkentätaso vastaa tiettyä klusterointiratkaisua (esimerkki: ks. kirjan kuva 14.1).

Esimerkkejä eräiden hierarkkisten kasaavien klusterointialgoritmien kanssa sovellettavista samankaltaisuusmitoista:

- *Single link* klusterointi: Samankaltaisuus lasketaan *kahden samankaltaisimman* yksilön välillä.
  Seuraus: klustereilla hyvä paikallinen koherenssi, mutta klusterit voivat olla repaleisia ja monimuotoisia. Kompleksisuus $\mathcal{O}(n^2)$.
- *Complete link* klusterointi: Samankaltaisuus lasketaan klusterin *kahden erikaltaisimman* yksilön välillä.
  Seuraus: klusterit ovat globaalisti kiinteämpiä. Kuitenkin laskennallisesti raskaampi $\mathcal{O}(n^3)$
- *Group average* klusterointi: Samankaltaisuus lasketaan *keskimääräisenä samankaltaisuutena* ryhmän jäsenten välillä. Kompromissi edellisten väliltä, ja laskennallisesti kohtuullisen tehokas $\mathcal{O}(n^2)$ jos havainnot representoidaan normalisoituina vektoreina ja käytetään pistetuloetäisyyttä.

14.3 Ei-hierarkkinen klusterointi

Useat ei-hierarkkisista algoritmeista iteroivat klusterointia siten että se vähitelle
paranee jonkin optimoitavan funktion tai mitan mielessä.

Joissakin algoritmeissa asetetaan klusterien määrään vaikuttava parametri,
esim. havainnon suurin sallittu etäisyys klusterin painopisteestä.

Algoritmien perusperiaate:

1. Alustetaan klusterointi jakamalla havainnot ryhmiin jollain tavalla, esim.
   satunnaisesti.

2. Iteroidaan: Uudelleenallokoidaan datoja siten että kustannusfunktion
   arvo pienenee.

3. Lopetetaan, esim. kun kustannusfunktion arvo alkaa kasvaa, tai kun
   parantuminen saavuttaa melko tasaisen alueen.

K-means-algoritmi

K-means tekee kovan klusteroinnin.

Klusterit esitetään prototyyppivektorien avulla, siten että prototyyppivektori on klusteriin kuuluvien havaintojen massakeskipiste (centroid).

Alustus voidaan tehdä satunnaisesti, esim. valitsemalla prototyyppien arvoiksi satunnaisesti valittu osajoukko havainnoista.

Havainto $\mathbf{x}_i$ sijoitetaan siihen klusteriin j jonka prototyypin $\mathbf{c}_j$ etäisyys havainnosta on pienin ($\arg\min_j d(\mathbf{x}_i, \mathbf{c}_j)$).

Algoritmi: Vuorottele vaiheita 1 ja 2:
Vaihe 1: Uudelleenallokoi kaikki havainnot klustereihin.
Vaihe 2: Laske klusterien keskipisteet
Lopetusehto: lopeta kun klusterointi ei enää muutu, tai kun on ajettu pyydetty määrä iteraatioita

Etäisyysfunktio d on tavallisesti Euklidinen etäisyys (L_2-normi) mutta myös muita metriikoita voidaan käyttää.

Jos yhtäsuuria minimietäisyyksiä esiintyy, nämä voidaan ratkaista satunnaisesti (mistä voi kuitenkin seurata ettei algoritmi konvergoidu, jos lopetusehtona on klusteroinnin muuttumattomuus).

Jos iteraatioita käydään läpi vakiomäärä, algoritmin kompleksisuus on $\mathcal{O}(n)$.

Algoritmi on herkkä alustukselle, joten on suotavaa ajaa useilla eri satunnaisalustuksilla, tai alustaa muuten 'fiksusti'.

Mikäli halutaan pehmeä klusterointi, se voidaan toteuttaa esim. Gaussin mikstuurimallia käyttäen ja optimoida esim. EM-algoritmilla. Mikäli Gausseilla käytetään diagonaalisia, kiinnitettyjä kovarianssimatriiseja joissa on pienet varianssit, saadaan käytännössä lähes sama malli kuin K-meansilla.

Keskeisenä erona on että K-means valitsee 'kiistatapauksissa' jommankumman klusterin, kun taas pehmeä klusterointi laittaa datan kuulumaan molempiin jollain tn:llä.

Oikean klusterien määrän määrittäminen

Useiden klusteroinnissa käytettyjen optimointikriteerien arvolla on taipumus parantua (opetusdatalla mitattuna) kun klusterien määrä nousee. Mitat ovat siis herkkiä ylioppimiselle, eivätkä varsinaisesti sovellu mallin rakenteen optimointiin.

Useat algoritmit eivät siksi suoraan kontrolloi klusterien määrää.

Klusterien optimaalisesta määrästä voi olla sovelluskohtaista prioritietoa, jolloin määrä voidaan valita etukäteen. Optimaalinen määrä voi löytyä myös klusterien sovelluksen tarjoaman evaluointitiedon kautta, kokeilemalla eri klusterimääriä.

Matemaattisesti hyvin perusteltuja tapoja klusterien lukumäärän optimointiin ilman aineiston ulkopuolista hyvyyskriteeriä tarjoavat MDL-periaate ja Bayeslainen mallinnus jossa sovelletaan variaatioanalyysiä. Molemmat näistä mittaavat yhtäaikaa sekä datan todennäköisyttä että mallin kompleksisuutta, eli ne voidaan laskea myös opetusdatalla.

Klusterien sopiva lukumäärä voidaan myös estimoida mittaamalla hyvyyttä erillisellä validointidatajoukolla: Valitaan klusterien määrä jolla validointijoukon hyvyysarvo on maksimi. (ks. kirjan kuva 16.4).

14.4 Klusteroinnin sovelluksia

Sovellustyyppejä yleensä:

- Exploratory data analysis: suuren ja kompleksisen aineiston havainnollistaminen ja tuki hypoteesien muodostamiselle
- kandidaattiluokkittelun muodostaminen jos luokkatietoa ei ole
- yleistäminen kun dataa on liian vähän per yksittäinen näytetyyppi (esim. kielimalleissa).
- Datan tehokas koodaus ja tiedonsiirto: välitetään kunkin havainnon osalta vain tieto klusterista johon se kuuluu ja etäisyys klusterista TAI etäisyydet useista/kaikista klustereista. (Tätä kutsutaan myös nimellä *vektorikvantisaatio*= kvantisoidaan datavektorien arvot johonkin pienempään joukkoon sovittuja 'perusarvoja')

Joitain klusteroinnin sovelluksia kieliteknologiassa:

- Syntaktisten tai semanttisten sanaluokitusten automaattiseksi löytämise tai hypoteesien tarjoamiseksi
- Sanojen ryhmittely kielimallinnusta varten (yleistäminen)
- Ohjaamaton sananmerkitysten disambiguointi (kontekstien klusteroinnti)
- Puheentunnistuksen foneemimallien (tai ali-sellaisten) ryhmittely (yleistä
- Dokumenttien ryhmittely tiedonhakua ja eksplorointia varten

HUOM: Tässä käytiin läpi vain muutamia esimerkkejä tavallisimmista klusterointialgoritmeista. Klusterointi käsitellään kattavammin ja yksityiskohtaisemmin kurssilla **T-61.231 Hahmontunnistuksen perusteet**.

14.5 Visualisointi ja datan eksplorointi

Ohjaamattoman oppimisen ja klusteroinnin eräs tärkeä sovellusalue on datan tutkiskelu, eli *exploratory data analysis (EDA)*.

Klusterointimenetelmien lisäksi tällä alueella sovelletaan ja kehitetään datan *visualisointimenetelmiä*, kuten:

- Pääkomponenttianalyysi (principal component analysis, PCA)
- Moniulotteinen skaalaus (multidimensional scaling, MDS)
- Itseorganisoiva kartta (self-organizing map, SOM)

Itseorganisoiva kartta on itse asiassa samanaikaisesti klusterointi- ja visualisointimenetelmä. Klusterointimenetelmänä se muistuttaa K-meansia.

Keskeinen ero on että SOM:in prototyypit muodostavat aineistosta epälineaaris järjestyneen kuvauksen (tavallisesti) 2-ulotteiselle hilalle. Kuvaus järjestyy itsestään mallin opetuksen aikana.

SOMin muodostama kuvaus eroaa MDS:n muodostamasta siten että MDS (eli 'Sammonin kuvaus') painottaa enemmän pitkien (globaalien) etäisyyksien esittämistä oikein.

SOM taas painottaa enemmän lokaalien (lyhyiden) etäisyyksien esittämistä, ja käyttää samalla visualisointipinta-alan paremmin hyväkseen. Kummallakin kuvauksella on meriittinsä.

Luennolla esitetään esimerkkejä itseorganisoivan kartan sovelluksista sanojen englannin ja suomen sanojen klusterointiin sekä WEBSOM-demo kartan sovelluksesta dokumenttien klusterointiin ja tiedonhakuun.

15. Tekstin luokittelu

(Text categorization)

Ohjatussa oppimisessa tyypillinen tehtävä on *luokittelu*.

Luokittelussa on annettuna joukko näytteitä joille kullekin tunnetaan luokkamuuttujan arvo. Tehtävänä on valita tai päätellä luokkamuuttujan optimaalinen arvo uudelle näytteelle, jonka luokkaa ei tunneta.

Päättely tehdään vertailemalla näytteen ominaisuuksia sekä luokiteltujen näytt ominaisuuksia, sekä hyödyntämällä tietoa tunnettujen näytteiden luokkamuuttujien arvoista.

Potentiaalisia sovelluksia SNLP:n piiristä on tuotu esiin aiemmin kurssin aikana, esim. sananmerkitysten yksikäsitteistäminen, ja tiedonhaun piirissä dokumenttien kategorisointi joihinkin ennalta annettuihin aiheluokkiin.

Erilaisia luokittelumenetelmiä käsitellään tarkemmin muilla informaatiotekniikan laboratorion kursseilla.

15.1 Evaluointi

Tavallisin mitta: Classification accuracy = oikein luokiteltujen osuus kaikista näytteistä.

Lisäksi tiedonhausta tutut mitat, kuten precision, recall, fallout

Julkisesti saatavia datasettejä tekstidokumenttien luokitteluun mm. Reuters collection.

Evaluointiin sovelletaan aiemmin kuvattuja periaatteita aineiston jaosta osiin (ks. luku 8.6 . Myös ristiinvalidointia voidaan soveltaa.

15.2 Esimerkkejä luokittelumenetelmistä

Seuraavaksi joitain esimerkkejä hyvin erityyppisistä, yleisesti käytetyistä luokitusmenetelmistä.

Naive Bayes-luokitin

NB (esiteltiin kurssin alkupuolella) on esimerkki tilastollisista luokittimista, jotka soveltavat Bayesin pätösteoriaa.

Erotuksena monimutkaisemmista Bayes-luokittimista, Naive Bayes olettaa piirteiden vaikuttavan luokitukseen toisistaan riippumattomasti.

Menetelmä toimii yllättävän hyvin huolimatta siitä että oletus ei useinkaan pidä paikkaansa.

Käytetään usein baseline-luokitusmenetelmänä.

Päätöspuu (decision tree)

Osittaa dataa aina jonkin piirteen mielessä kerrallaan kunnes jokaisessa osassa on pelkästään yhden luokan alkioita.

Osiin jako tapahtuu ns. maksimaalisen informaationlisäyksen kriteerillä (maximum information gain): Valitaan sellainen piirre-arvo-pari jolla jakaminen maksimoi äiti-noodin ja sen lapsi-noodien välisen luokkaentropian erotuksen.

Puita rakennettaessa yleensä ensin jaetaan dataa osiin yhä uudelleen, ja lopuksi 'karsitaan' puuta ylioppimisen välttämiseksi.

Menetelmän paras puoli on se että se on ihmiselle helposti ymmärrettävä ja helposti tulkittavissa.

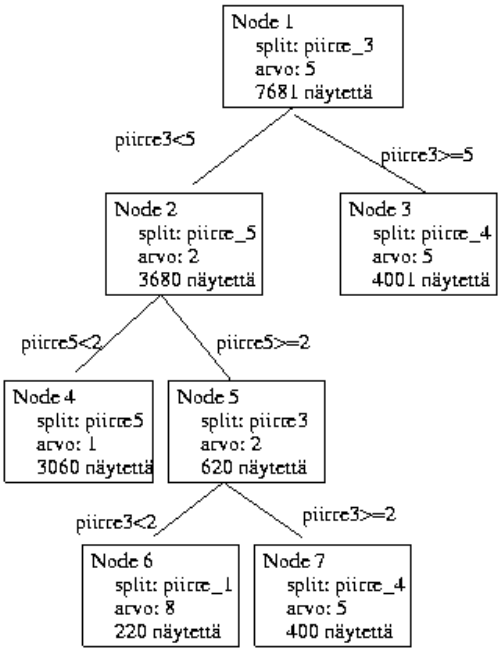
Menetelmän eräs huono puoli on että se tarkastelee vain yhtä muuttujaa kerrallaan, ts. jakaa data-avaruutta vain alkuperäisten muuttujien suunnissa (ks. kuva). Useat ongelmat ovat kuitenkin aidosti monimuuttujaisia.

Menetelmä myös johtaa helposti epäoptimaalisiin partitiointijärjestyksiin, ja

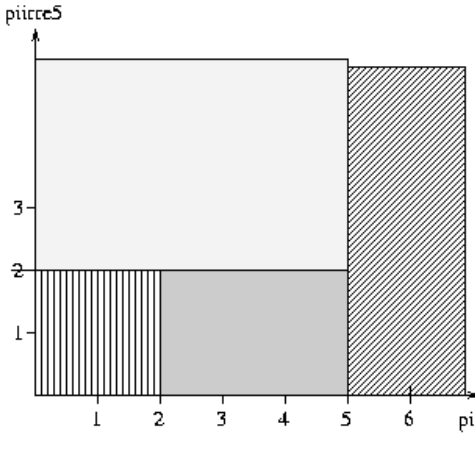
epätasapainoisiin puihin, jonka seurauksena päätöksenteko hidastuu radikaalisti ($\mathcal{O}(\log(n)) \rightarrow \mathcal{O}(n)$).

Puiden karsimisesta huolimatta ylioppiminen on menetelmälle tyypillinen ongelma.

Päätöspuu



Luokittelupäätösten geometrinen tulk



Monikerrosperseptroni (multi-layer perceptron, MLP)

Ehkä tunnetuin neuroverkkoluokitin. Käydään läpi mm. neuraalilaskennan peruskurssilla.

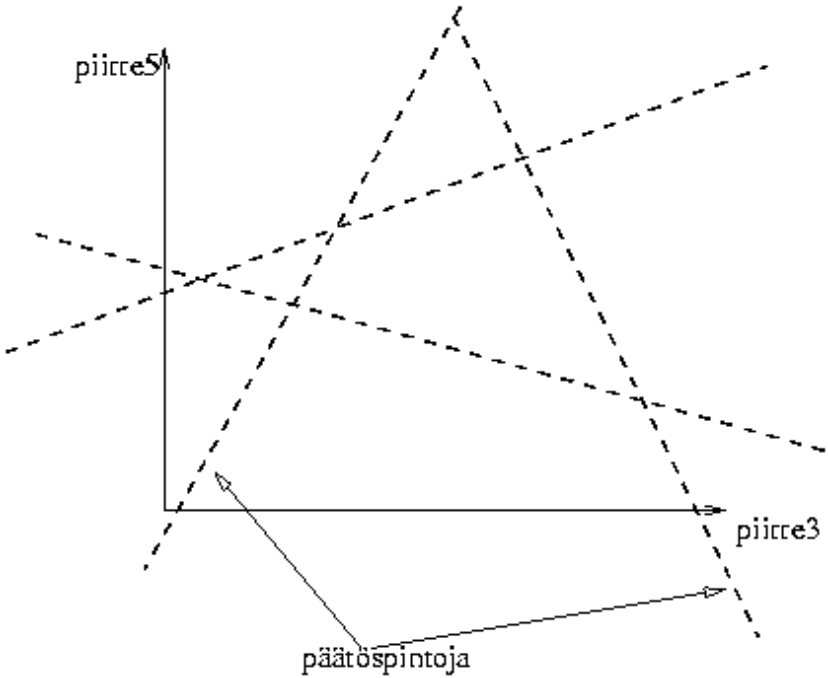
MLP jakaa data-avaruutta osiin hypertasoilla jotka eivät rajoitu alkuperäisten muuttujien määräämiin suuntiin (ks. kuva, päätöspinnat).

MLP asettaa päätöspinnat Bayesin päätösteorian mielessä optimaalisiin sijainteihin.

Hankaluutena on mm. verkon välikerrosten neuronien määrän valinta, sekä epälineaarisen menetelmän mahdolliset lokaalit optimit.

Menetelmästä on kehitetty suuri joukko variantteja.

MLP:n geometrinen tulkinta



K:n lähimmän naapurin luokitin (k nearest neighbor, KNN)

Ei-parametrinen menetelmä joka tekee luokituspäätöksen äänestyksenä näytteen kanssa lähimpien k:n luokitellun näytteen kesken.

Tasatilanteet ratkaistaan arvalla.

Kutsutaan joskus myös nimellä Memory-based learning (MBL) koska pienen joukon malleja sijasta menetelmässä tallennetaan koko opetusdata.

Etäisyysmitta voidaan valita vapaasti, tai jopa vaihdella tilanteen mukaan.

Toimii usein yllättävän hyvin yksinkertaisuudestaan huolimatta.

Mikäli dataa on käytettävissä kovin paljon, menetelmä on luokittelun aikana hidas ja vie paljon muistia: jokaisen luokiteltavan näytteen kohdalla on laskettava sen etäisyys kaikkiin opetusjoukon näytteisiin.

Usein käytetty baseline-menetelmä.

16. Tilastollinen konekääntäminen

- Automaattinen kielenkääntäminen on eräs pitkäaikaisimmista kielioteknologian tavoitteista.
- Konekääntäminen (machine translation) on kuitenkin hyvin vaikea ongelma.
- Nykyisten konekäännösohjelmien tulos toimii lähinnä raakakäännöksenä joka voi nopeuttaa aidon kielenkääntäjän työtä, muta ei sellaisenaan kelpaa ihmislukijalle.
- Hyvin rajallisissa sovellusalueissa (kuten säätiedotukset) voidaan päästä kohtuulliseen lopputulokseen täysin automaattisesti.

Kääntämisen eri tasoja

- Yksinkertaisin lähestymistapa on *sanasta-sanaan käännös* korvaa lähtökielen sanoja kohdekielen sanoilla. Lopputuloksen sanajärjestys on usein väärä.

- *Muunnosmenetelmät* (syntaktinen ja semanttinen) rakentavat rakenteisen välirepresentaation lähtökielen sanajonosta muuntavat sen kohdekielen välirepresentaatioksi (jonkinlaisia sääntöjä käyttäen) ja generoivat tästä kohdekielen sanajonon.

- *Syntaktinen muunnosmenetelmä* rakentaa lähtökielen sanajonosta syntaktisen rakennekuvauksen. Lähestymistapa edellyttää toimivaa syntaktista disambiguointia.

  Tällä tavoin voidaan ratkaista sanajärjestysongelmat, mutta usein lopputulos ei ole semanttisesti oikein. Esim. saksan 'Ich esse gern' (Syön mielelläni) kääntyisi syntaktisella menetelmällä 'I eat readily' (tai 'willingly', 'gladly'). Englannissa saksan ilmausta vastaavaa verbi-adverbiparia ei kuitenkaan ole, vaan oikea käännös olisi 'I like to eat'.

- *Semanttisissa muunnosmenetelmissä* tehdään syntaktista jäsennystä täydellisempi kuvaus, semanttinen jäsennys, jonka tarkoituksena on saada aikaan käännös joka on myös semanttisesti oikein.

  Kuitenkin semanttisesti 'sanatarkka' käännös voi olla kohdekielessä kömpelö, vaikka onkin periaatteessa ymmärrettävissä. Esim. espanjan lauseen 'La botella entró la cueva flotando.' tarkka käännös olisi 'the bottle entered the cave floating' (pullo tuli luolaan kelluen) mutta luontevampaa olisi sanoa 'the bottle floated into the cave' (pullo kellui luolaan).

  Useiden kömpelöiden ja epäluontevien käännösten käyttö hidastaa ymmärtämistä, vaikka ymmärtäminen olisikin periaatteessa mahdollista. Monitulkintaisuuden mahdollisuudesta johtuen epäluonteva käännös voidaan myös helpommin tulkita väärin.

- *Interlingua* – keinotekoinen yleinen (kieliriipumaton) välikieli tai tietämysrepresentaatio. Käännetään lähtökielestä interlingualle ja interlinguasta mille tahansa kohdekielelle.

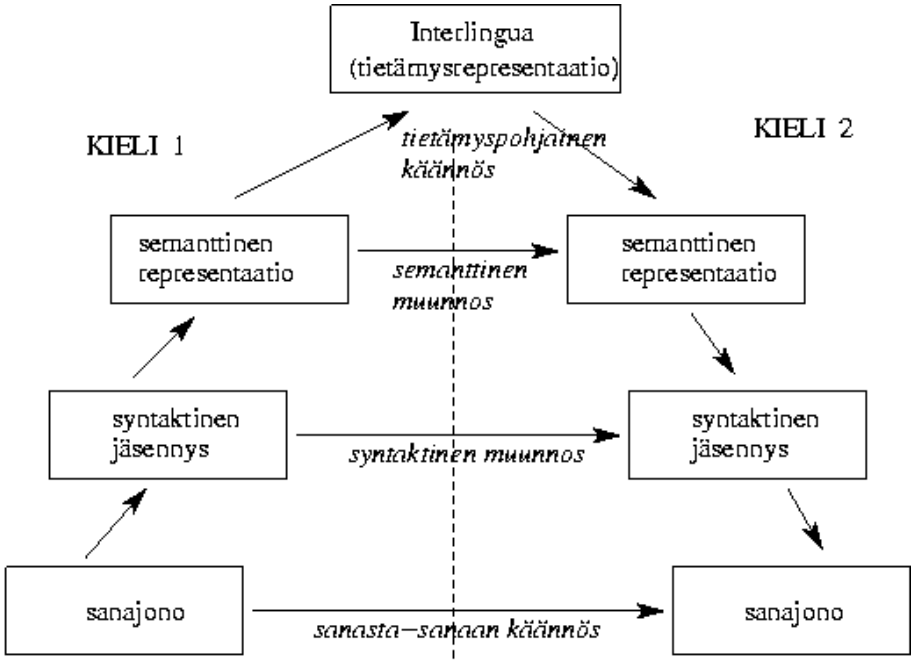
Kääntimiä n kielen välille tarvitaan tällöin n^2 kpl sijaan vain $2n$ kpl. Lisäksi ne voidaan toteuttaa mahdollisimman suurelta osin yleiskäyttöisil kielenkäsittelymenetelmillä. Kuitenkin riittävän välikielen määrittely on itsessään hankala ongelma, jota ei ainakaan toistaiseksi ole ratkaistu riittävässä laajuudessa.

Seuraavan sivun kuvassa on näytetty konekäännösjärjestelmän vaihtoehtoiset toteutustavat.

Tilastollisen kielenkäsittelyn menetelmiä voidaan käyttää järjestelmän komponentteina minkä tahansa nuolen kohdalla (esim. jäsentäminen, disambiguointi jne).

Konekääntimet voivat myös olla kombinaatioita symbolisista ja tilastollisista komponenteista.

Pelkästään kielenkääntämiselle erityinen ongelma on *tekstinlinjaus*.


Interlingua
(tietämysrepresentaatio)
KIELI 1
KIELI 2
tietämyspohjainen
käännös
semanttinen
representaatio
semanttinen
muunnos
semanttinen
representaatio
syntaktinen
jäsennys
syntaktinen muunnos
syntaktinen
jäsennys
sanajono
sanasta–sanaan käännös
sanajono

16.1 Tekstinlinjaus

(text alignment)

- Tekstinlinjauksella tarkoitetaan kahden erikielisen *rinnakkaistekstin* asettamista kohdakkain siten että osoitetaan toisiaan vastaavat tekstijonot.
- Rinnakkaisteksteillä tarkoitetaan saman dokumentin erikielisiä käännöksiä.
- Useimmin käytetyt rinnakkaistekstit ovat hallinnollisia tekstejä peräisin maista tai valtioliitoista joissa on useita virallisia kieliä (esim. EU, Kanada, Sveitsi, Hong Kong).
- Helpon saatavuuden lisäksi hallinnolliset rinnakkaistekstit ovat yleensä konsistentisti ja mahdollisimman tarkasti käännettyjä. Tällainen aineiston korkea laatu on tärkeää sekä tilastollisten menetelmien kehittämiselle että menetelmien evaluoinnille.
- Myös sanoma- ja aikakauslehtiä joskus käytetään, ja myös uskonnollisia tekstejä olisi helposti saatavilla. Kuitenkin tulokset ovat yleensä

selvästi heikompia, oletettavasti johtuen vähemmän sanatarkoista ja konsistenteista käännöksistä, ja vähemmän stationaarisesta tekstilajista (esim. ajankohtaiset uutisaiheet muuttuvat nopeasti).

- Tekstinlinjauksessa on yleensä kaksi vaihetta:

    1. Lauseiden ja kappaleiden linjaus: tekstin raakalinjaus, jossa toisiaan vastaavat kappaleet, lauseet ja lauseparit asetetaan suunnilleen kohdakkain.
    2. Sanojen linjaus ja kaksikielisen sanakirjan indusointi, jossa raakalinjatun aineiston perusteella etsitään mitkä sanat yleensä käännetään miksi toisiksi sanoiksi.

16.2 Lauseiden ja kappaleiden linjaus

Yleensä lauseiden linjaus on välttämätön ensimmäinen askel monikielisen korpuksen tuottamisessa.

Konekäännöksen ja kaksikielisten sanakirjojen tuottamisen lisäksi linjaus voi hyödyttää myös muita sovelluksia, kuten

- Sananmerkitysten disambiguointi: sanan eri merkityksiä voidaan ryhmitellä sen saamien eri käännösvastineiden perusteella.
- Monikielinen tiedonhaku: Tiedonlähde voi olla eri kielellä kuin millä kysymys esitetään.
- Kääntäjän apuväline: Kun dokumenttien tiedot muuttuvat, voidaan automaattisesti osoittaa toisenkielisen dokumentin kohta jota täytyy myös päivittää, ja ehkä ehdottaa päivitystä.

bead, jyvä: lause tai muutaman lauseen jono ja sitä vastaavat (linjatun tekstin) toisenkielinen lausejono. Kumpi tahansa jono voi olla myös tyhjä. Jokainen lause kuuluu täsmälleen yhteen jyvään.

Tehtävänä on *muodostaa koko tekstin osalta jyvitys*, eli kuvaus jossa on jaettu tekstit paloihin ja kerrottu mitä kielen 1 palaa mikin kielen 2 pala vastaa.

Lauseiden linjaus ei ole triviaali ongelma, koska yhtä lähtökielen lausetta ei läheskään aina vastaa yksi kohdekielen lause (1:1-jyvä):

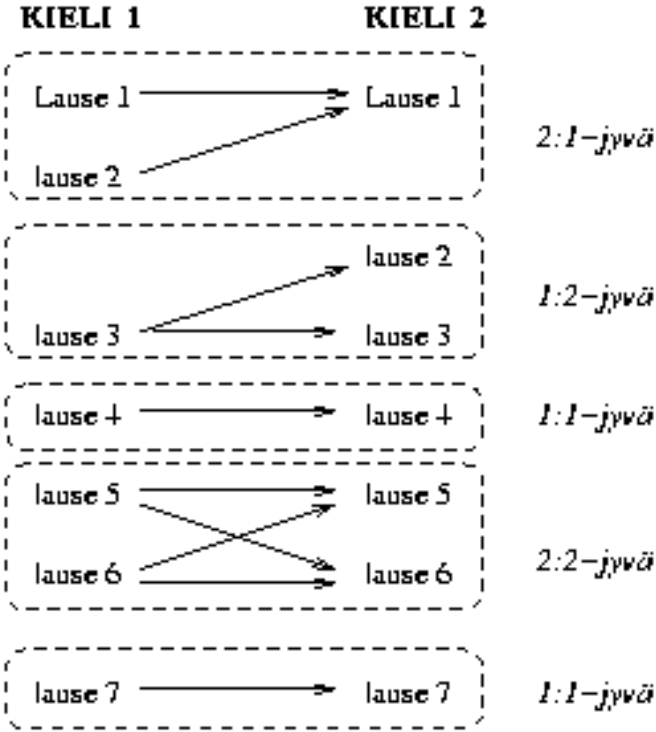
1:2 ja 2:2-jyvät: Lauseita voidaan pilkkoa eri tavoilla, ja ihmiskääntäjä voi järjestää asioita eri järjestyksiin tehdäkseen lopputuloksesta luontevan.

2:2-vastaavuudessa lähtökielen kahden peräkkäisen lauseen osia esitetään kohdekielen kahdessa peräkkäisessä lauseessa (riittävä päällekkäisyys).

Milloin päällekäisyys on riittävä? Yleensä muutaman sanan siirtyminen ei riitä, vaan edellytetään kokonaisen lausekkeen päällekkäisyyttä.

Ks. esimerkki 2:2-vastaavuudesta kirjan kuvassa 13.2.

Joskus esiintyy myös 1:3 ja 3:1-vastaavuuksia.


KIELI 1
KIELI 2
Lause 1
Lause 1
lause 2
2:1–jyvä
lause 2
lause 3
lause 3
1:2–jyvä
lause 4
lause 4
1:1–jyvä
lause 5
lause 5
lause 6
lause 6
2:2–jyvä
lause 7
lause 7
1:1–jyvä

Poistot ja lisäykset eli 1:0 ja 0:1-jyvät: Joitain asioita voidaan sanoa eksplisiittisesti toisella kielellä, mutta jättää pois toisella kielellä, koska ne oletetaan implisiittisesti tulkittaviksi (ehkä asioiden erilaisen järjestyksen ansiosta, ehkä sanojen erilaisten sivumerkitysten takia, ehkä kulttuurisista syistä).

Eri tutkimusten perusteella voidaan arvioida että n. 90% vastaavuuksista on tyyppiä 1:1 (tosin prosentti on luultavasti kielipari- ja tekstilajiriippuva).

On myös melko tavallista että kääntäjät järjestävät lauseita eri järjestyksiin. Tässä esitetyt mallit eivät kuitenkaan kykene representoimaan tätä mahdollisuutta, vaan tulkitsevat tapaukset mm. poistoiksi ja lisäyksiksi.

Menetelmiä

Osa tilastollisista menetelmistä perustuu ainoastaan tekstinpätkien pituuksien tarkasteluun, osa taas huomioi lauseissa käytetyn sanaston (merkkijonot).

Olkoon kielen 1 teksti S jono lauseita $S = (s_1, \ldots, s_I)$ ja kielen 2 tekti T samoin $T = (t_1, \ldots, t_J)$

Tekstinpätkien pituuksiin perustuvat menetelmät

Useat varhaiset tekstinlinjausmenetelmät ovat tätä tyyppiä.

Etsitään linjaus A jolla on suurin tn:

$$\arg\max_A P(A|S,T) = \arg\max_A P(A,S,T) \tag{65}$$

(todennäköisin linjaus voidaan etsiä mm. dynaamisella ohjelmoinnilla).

Useat menetelmät jakavat linjatun tekstin jonoksi jyviä $(B_1, \ldots, B_K)$ ja approksimoivat koko linjatun tekstin tn:ää olettamalla että jyvän tn ei riipu sen ympäristöstä, vaan ainoastaan jyvän sisältämistä lauseista:

$$P(A,S,T) = \prod_{k=1}^{K} P(B_k) \tag{66}$$

Miten lasketaan jyvän todennäköisyys?

Gale & Church, 1991, 1993:

Jyvän tn. riippuu jyvässä olevien lauseiden pituuksista merkkeinä mitattuna. Perustuu oletukseen että yhden kielen pitkiä pätkiä todennäköisesti vastaavat pitkät pätkät myös toisessa kielessä.

Oletetaan että aineistot on jo linjattu kappaletasolla (laskennallisen tehokkuuden vuoksi).

Sallitaan vain linjaustyypit $\{1:1, 1:0, 0:1, 2:1, 1:2, 2:2\}$

Olkoon $D(i,j)$ etsitty pienimmän kustannuksen linjaus lauseiden $s_1, \ldots, s_i$ ja $t_1, \ldots, t_j$ välillä.

Lasketaan $D(i,j)$ rekursiivisesti. Perustapaus, määritellään: $D(0,0) = 0$.

Rekursio:

$$
\begin{aligned}
D(i,j) = \min \quad & D(i,j-1) && + cost(0:1 \text{ linjaus } 0, t_j) \\
& D(i-1,j) && + cost(1:0 \text{ linjaus } s_i, 0) \\
& D(i-1,j-1) && + cost(1:1 \text{ linjaus } s_i, t_j) \\
& D(i-1,j-2) && + cost(1:2 \text{ linjaus } s_i, t_{j-1}, t_j) \\
& D(i-2,j-1) && + cost(2:1 \text{ linjaus } s_{i-1}, s_i, t_j) \\
& D(i-2,j-2) && + cost(2:2 \text{ linjaus } s_{i-1}, s_i, t_{j-1}, t_j)
\end{aligned}
$$

Kunkin tyyppisen linjauksen (jyvän) kustannus lasketaan seuraavasti:

Oletetaan malli: yksi kielen L_1 merkki generoi satunnaisen määrän merkkejä kieleen L_2. Oletetaan generoinneille gaussinen tn-jakauma, jonka keskiarvo μ ja varianssi σ^2 estimoidaan suurista rinnakkaiskorpuksista (saksa/englanti-parille estimoitiin $\mu = 1.1$ koko korpuksesta, ranska/englanti-parille 1.06. Varianssi voidaan estimoida kappaletason linjausta hyväksikäyttäen.)

Kustannuksena voidaan käyttää tekstinpätkien etäisyyden negatiivista log-

likelihoodia mallissa:

$$cost(l_1, l_2) = -\log P(\alpha \text{ linjaus } |\delta(l_1, l_2, \mu, \sigma^2) \quad (67)$$

jossa α on jokin sallituista linjaustyypeistä ja $\delta(l_1, l_2, \mu, \sigma^2) = (l_2 - l_1\mu)/\sqrt{l_1\sigma^2}$

Tarvittavat todennäköisyydet estimoidaan soveltamalla Bayesin kaavaa $P(\alpha|\delta)$ $P(\alpha)P(\delta|\alpha)$. Tällöin siis 1:1-linjauksen suuri prioritodennäköisyys (90 %) aiheuttaa preferenssiä sen valintaan.

Rekursiivinen kustannusten laskenta-algoritmi on hidas jos tekstinpätkät ovat pitkiä. Yksittäisillä kappaleilla kuitenkin suhteellisen nopea.

Menetelmä toimii melko hyvin sukukielillä: raportoitu 4% virhemäärä. Kun lisäksi pyrittiin erikseen tunnistamaan epäilyttävät linjaukset, ja linjaamaan vain parhaat 80% päästiin virhetasoon 0.7

Menetelmä toimii parhaiten 1:1-linjauksilla (2%), mutta hankalammille linjauksille virheprosentit ovat suuria.

Menetelmän variantissa (Brown et al, 1991) verrataan lauseiden pituuksia

sanoina, ei merkkeinä.

Church, 1993: Identtisiin merkkijonoihin perustuva menetelmä

Edelliset menetelmät eivät sovellu kohinaiseen tekstiin, esim. optisen tekstintunnistuksen tuottamaan, jossa saattaa olla roskaa välissä, tai kokonaan kadonneita kappaleita, tai kappale- ja lauserajat vaikeita havaita mm. kadonneiden välimerkkien tai roskan takia.

Tämän menetelmän perustana oleva huomio:

Teksteissä jotka on kirjoitettu jokseenkin samalla aakkostolla (esim. roomalaiset aakkoset) esiintyy samaatarkoittavia, identtisiä kirjainsekvenssejä, kuten erisnimiä tai numeroita.

Sukulaiskielillä, tai läheisessä vuorovaikutuksessa olevilla kielillä voi lisäksi esiintyä muitakin yhteisiä sekvenssejä, johtuen yhteisestä kantamuodosta (esim. englannin 'superior' ja ranskan 'supérieur') tai lainasanoista.

Lasketaan identtisiä merkki-n-grammeja (n esim. 4). Etsitään n-grammien

linjaus joka sisältää mahdollisimman paljon identtisiä n-grammipareja. Lisäksi n-grammeja voidaan painottaa frekvenssin mukaan.

Menetelmä ei tuota varsinaista lauseiden jyvitystä.

Voi epäonnistua täydellisesti mikäli kielissä ei ole riittävästi yhteisiä merkkijonoja.

Fung ja McKeown, 1994

Estimoidaan pieni kaksikielinen sanakirja joka antaa todennäköisesti vastaavia sana-käännös-pareja. Käytetään näitä 'ankkureina' linjauksessa, kuten aiemmin käytettiin yhteisiä n-grammeja.

Edelliseen verrattuna parannusta on, ettei tarvita yhteisiä sanoja tai n-grammej

Parannus pituuspohjaisiin menetelmiin on että tämä hyödyntää leksikaalista tietoa.

Leksikaaliset menetelmät

Tavoitteena kuitenkin tuottaa aito lausetason 'jyvitys'.

Vaikutaa selvältä että tieto sanojen todennäköisistä käännöspareista auttaisi linjausta huomattavasti.

Puhtaasti tilastollisten menetelmien keskeinen ajatus: Vuorotellaan todennäköi osittaislinjauksen tekemistä sanatasolla ja todennäköisimmän lausetason linjauksen tekemistä.

Apuna käytetään lisäksi oletusta että toisiaan vastaavat lausejonot eivät luultavasti ole kovin kaukana toisistaan (esim. ristiinmenoja ei ole tai ne eivät ole pitkiä).

Iteraatioita ei yleensä tarvita kovin montaa (johtuen yo. rajoituksesta).

Variantteja

Chen, 1993: Sovelletaan yksinkertaista sana-sana-käännösmallia estimoimaan sanaparien käännöstn:t. Lasketaan tällä mallilla maksimaalisen todennäköinen

linjaus.

Hyviä puolia: aidon käännösmallin käyttö oletettavasti parantaa tarkkuutta puoliheuristisiin menetelmiin verrattuna. Yksinkertaisen käännösmallin käyttö tekee menetelmästä laskennallisesti tehokkaan.

Huono puoli: mallin yksinkertaisuuden aiheuttamat approksimaatiot voivat aiheuttaa ongelmia kun pitävät huonosti paikkaansa jollekin kieliparille tai tekstiparille.

Menetelmää on sovellettu suurten korpusten (useita miljoonia lauseita) linjaamiseen. Sen virheprosentiksi on estimoitu 0.4% mikä on yhtä hyvä tai parempi kuin muilla menetelmillä saman korpuksen osajoukolla.

Sanojen linjaus ja kaksikielisten sanakirjojen estimointi

Sanatason linjauksen peruslähestymistapa: vuorotellaan seuraavia askeleita:

1. muodostetaan jokin sanatason linjaus 2. estimoidaan sen perusteella sanaparien käännöstodennäköisyydet

Sovelletaan siis EM-tyyppistä algoritmia.

Kaksikieliseen sanakirjaan hyväksytään (lopulta) vain sanaparit joista on saatu riittävästi evidenssiä, eli esim. riittävän monta näytettä kyseisten sanojen vastaavuudesta.

Voidaan olettaa että jatkossa sanojen (ja lauseiden) linjauksessa käytetään myös kaksikielisten sanakirjojen sisältämää tietoa.

Suoraviivainen sovellustapa olisi käyttää tällaista sanakirjaa initialisoimaan edellä esitetyn algoritmin sanaparien käännöstodennäköisyydet.

16.3 Konekääntäminen

Kohinaisen kanava-mallin eräs sovellus on konekääntäminen (malli esiteltiin puhuttaessa Shannonin informaatioteoriasta kurssin alussa).

Kun halutaan rakentaa malli jolla käännetään tekstiä englannista e suomeksi s, ajatellaan että on olemassa kohinainen kanava johon syötetään tekstiä suomeksi ja se tulee ulos englanniksi. Meidän on siis ainoastaan dekoodattava kohinainen signaali takaisin suomeksi.

Tarvittavaan menetelmään kuuluu kolme osaa:

1. Kielimalli $P(s)$ kertoo suomen lauseiden tn:t,
2. Käännösmalli $P(e|s)$ kertoo lauseiden kääntymistn:t englanniksi,
3. Dekooderi $\hat{s} = \arg\max_s P(s|e)$ laskee todennäköisimmän suomenkielisen lauseen kun tunnetaan englanninkielinen.

Kielimallien estimointia on käsitelty laajasti kurssin muissa osissa.

Käännösmalli

Valitaan tässä käännösmalliksi hyvin yksinkertainen sana-käännösmalli, joka olettaa että jokainen englannin sana generoituu (kääntyy) 0 tai 1:stä suomen sanasta, kun taas suomen sana voi vastata useaa englannin sanaa. Oletetaan lisäksi että peräkkäisten sanojen generoituminen (kääntyminen) on toisistaan riippumatonta.

Olkoon s suomenkielinen lause ja e englanninkielinen. Tällöin käännöstn on

$$P(e|s) = \frac{1}{Z} \sum_{a_1=0}^{l} \cdots \sum_{a_m=0}^{l} \prod_{j=1}^{m} P(e_j|s_{a_j}), \tag{68}$$

jossa l ja m ovat sanojen lukumäärät lauseissa s ja e, ja $P(e_j|s_{a_j})$ on tn jolla sana englannin lauseessa positiossa j generoituu suomen sanasta joka on positiossa a_j (0 tarkoittaa tyhjää joukkoa). Z on normalisointitekijä.

Sisäkkäiset summaukset summaavat siis yli kaikkien mahdollisten vaihtoehtoisten linjausten, ja kertolasku kertoo yli sanajonon.

Dekooderi

Dekooderin estimointi tapahtuu samoilla periaatteilla ja menetelmillä kuin aiemmin kurssilla on esitetty (ks. luku Markov-malleista).

$$\hat{s} = \arg\max_s P(s|e) = \arg\max_s \frac{P(e|s)P(s)}{P(e)} = \arg\max_s P(e|s)P(s) \quad (69)$$

Lisäksi voidaan huomioida

sanojen sijainti lauseessa: esim. että linjaukset joissa suomen lauseen alussa oleva sana linjataan englannin lauseen lopussa olevaan sanaa ovat epätodennäk kuin lähempänä toisiaan olevat vastaavuudet.

sanan hedelmällisyys: sanan tn. tuottaa useita sanoja kohdekieleen (jotkut sanat ovat toisia hedelmällisempiä)

Tulokset

Mallin soveltaminen englannista ranskaksi kääntämiseen Hansard-korpuksella antoi tulokseksi vain 48% oikein käännettyjä lauseita.

Virheissä oli sekä kieliopillisia että semanttisia virheitä.

Ongelmia mm.

Näin yksinkertaisella lähestymistavalla on monia ilmiselviä ongelmia, jotka johtuvat mm. siitä että käännetään suoraan sanatasolla.

Joitain menetelmälle ominaisia ongelmia

- herkkyys opetusdatalle: pienet muutokset opetusdatan (tai testidatan) valinnassa aiheuttavat suuria muutoksia tulosprosentteihin. Vastaavuuden testi- ja opetusdatan välillä on oltava hyvin suuri jotta tällainen sanatason käännösmalli toimisi hyvin.
- Tehokkuus: raskas pitkille lauseille
- Datan harvuus (riittämättömyys)
- Jos kielimalli on lokaali (esim. n-grammimalli), ei auta vaikka käännösmalli osaisi tuottaa käännöksiä hyödyntäen pitkän matkan riippuvuuksia. Eri mallien tekemien oletusten pitäisi olla konsistentteja.