Relevanssi, Information Bottleneck ja sarjallinen ryhmittely

Ville H. Tuulos
tuulos@cs.helsinki.fi

Klusterointimenetelmät-seminaari
Helsingin yliopisto, tietojenkäsittelytieteen laitos
Raportti C–2002–54, s. 52–63, marraskuu 2002

Tiivistelmä

Ryhmittelytraditiossa usein turvaudutaan menetelmiin, joiden käyttö on teoreettisesti arveluttavaa. Menetelmiin sisältyy oletuksia, jotka oleellisesti vaikuttavat ryhmittelyn lopputulokseen, mutta eivät välttämättä laisinkaan käyttäjän odottamalla tavalla. Vieläpä kun ryhmittely on luonteeltaan eksploratiivista, on tulosten validointi vaikeaa. Pahimillaan käyttäjä joutuu tekemään tulkintoja perustuen mielivaltaiseen lopputulokseen.

Työssä esitellään Information Bottleneck -kustannusfunktio, joka sallii käyttäjän määritellä ryhmittelyn määräävän relevanssin eksplisiittisesti. Menetelmän formalisointi nojautuu informaatioteorian tuloksiin ja käsitteistöön. Lisäksi esitellään käytännöllinen ryhmittelyalgoritmi, joka voi hyödyntää Information Bottleneckia kustannusfunktionaan.

1 Taustaa

Teoreetikon näkökulmasta klusterointi on vaikea tehtävä. Vaikka klusterointia tehdään täysin eksaktein ja matemaattisesti motivoiduin menetelmin, ei itse ongelmalle ole muodostunut yleisesti hyväksyttyä määritelmää. Osittain syy on alan termistön löysässä käytössä. Osittain vika on tutkijoissa, jotka tukevan teoriapohjan luomisen sijasta usein keskittyvät olemassaolevien menetelmien adhoc-parannuksiin suhtautuen välinpitämättömästi tunnettuihin teoreettisiin sudenkuoppiin. Pohjimmiltaan syy lienee kuitenkin itse klusteroinnissa, joka luonteeltaan on laskennallisesti vaikeasti lähestyttävä.

Tässä työssä klusterointia pidetään hahmojen etsintäongelmana (*pattern discovery*)[1], minkä ei käytännössä pitäisi rajoittaa tarkastelun yleisyyttä. Tällä valinnalla pyritään kiertämään ryhmä-käsitteen sisällöttömyys. Hahmo korostaa ilmiön (so. “ryhmän”) sisäistä rakennetta, säännönmukaisuutta ja tunnistettavuutta. Hahmo muodostuu joukosta käsiteltävän datan yksittäisiä näytteitä. Ryhmittelyn perusongelma ei täten muutu; data halutaan järjestää mielekkäisiin osiin.

Kaikki hahmot eivät ole mielenkiintoisia. Hahmojen määrä kasvaa eksponentiaalisesti suhteessa näytteiden lukumäärään (n alkioisesta joukosta saadaan 2^n osajoukkoa), joten kaikkien hahmojen läpikäynti ei ole mielekästä koneelle eikä ihmiselle. On määriteltävä mitta hahmon mielenkiintoisuudelle, relevanssille. On selvää, ettei absoluuttista mittaa ole. Tämä

[1] Huomaa kontrasti toiseen yleiseen ongelmaan, hahmontunnistukseen (*pattern recognition*).

ei kuitenkaan romuta ryhmittelyn mielekkyyttä, sillä käyttäjä voi määritellä mitan tapauskohtaisesti. Hahmojen eksponentiaalisesta määrästä johtuen jo keskikokoisesta aineistosta löytyy tukea käytännössä jokaiselle relevanssimäärittelylle, joten yhtä totuutta ei ryhmittely kerro. Ellei käyttäjä määrittele relevanssiaan tietoisesti, on tulosten tulkinta lähes mielivaltaista.

Perinteisesti ryhmittelyä on lähestytty toisesta näkökulmasta. Hahmon relevanssi määritellään implisiittisesti näytteiden samankaltaisuuden avulla; hahmot pyritään muodostamaan siten, että ne sisältävät mahdollisimman samankaltaisia näytteitä. Probabilistisessa ryhmittelyssä samankaltaisiksi tulkitaan näytteet, jotka todennäköisesti ovat saman satunnaisprosessin (jakauman) generoimia. Toinen koulukunta määrittelee erityisen samankaltaisuusmitan, jolla näytepareja voidaan vertailla.

Kumpikin lähestymistapa on ongelmallinen. Edellinen vaatii malliluokan kiinnittämisen etukäteen, jälkimmäinen mitan määrittelyn. Käytännössä nämä *apriori* valinnat määräävät hahmojen relevanssimitan, mutta kuinka tietoisesti käyttäjä tekee valinnan? Lopulta itse ryhmittelyalgoritmi, joka ratkaisee kiinnitetyn optimointiongelman vaikuttaa lopputuloksiin vain vähän. Tämä on luonnollista, sillä optimointiongelma on mallinvalinnan jälkeen hyvin määritelty ja ratkaistavissa usein iteratiivisesti jollakin *k-means* henkisellä algoritmilla.

1.1 Relevanssi perinteisissä ryhmittelymenetelmissä

Tyypillisesti käyttäjä turvautuu laajalti käytettyyn menetelmään, joskus jopa tietämättä millaisia voimakkaitakin oletuksia tehtyyn valintaan liittyy. Tarkastellaan lyhyesti esimerkkinä kahta yleistä ryhmittelymenetelmää: normaalijakautuneita sekajakaumamalleja ja menetelmiä, jotka hyödyntävät euklidista metriikkaa samankaltaisuusmittana.

Yleisesti K-komponenttinen normaalijakaumien muodostama sekoitemalli määritellään seuraavasti

$$f(x) = \sum_{k=1}^{K} w_k f_k(x; \mu_k, \sigma_k), \tag{1}$$

missä w_k on komponentin k prioritodennäköisyys ja μ_k, σ_k ovat komponenttikohtaiset parametrivektorit. Yleisesti malli sovitetaan dataan maksimoiden *maximum likelihood* -mittaa iteratiivisesti EM-algoritmilla. EM-algoritmi alustetaan sopivilla satunnaisarvoilla ja iteroidaan kunnes yhdenkään pistetodennäköisyyden muutos ei ylitä arvoa ϵ.

Millaisia hahmoja tämä malli pitää relevantteina? EM-algoritmi pyrkii estimoimaan normaalijakauman parametrit, jakauman keskiarvon ja varianssin, siten että näytteiden todennäköisyys suhteessa tähän malliin maksimoituu (lokaalisti). Relevantteina pidetään siis hahmoja, jotka selittävät miten *esimerkiksi* koko datan voisi mahdollisesti saada tuotettua K:lla normaalijakaumalla. Parasta tapaa menetelmä ei kerro, eikä sitä onko data missään määrin normaalijakautuneiden prosessien tuottamaa. Ongelmat ovat ilmeisiä jos (ja kun) data on kohinaista eikä K:n arvoa tunneta. Tämä määritelmä hahmojen relevanssille asettaa suuria vaatimuksia näytteiden keräys- ja esikäsittelyvaiheille, jotta tulkitsijan ja menetelmän näkemykset kohtaisivat. Menetelmän luonnetta kuvaa tarkemmin [Aut02].

Jälkimmäisenä esimerkkinä tarkastellaan partitiointiin perustuvaa klusterointia, jossa näytteiden samankaltaisuutta mitataan euklidisella metriikalla. Euklidinen mitta saadaan erikoistapauksena yleisestä L_k normista kun $k = 2$,

$$L_k(x, y) = \sum_{i=1}^{d} (\|x^i - y^i\|^k)^{1/k}. \tag{2}$$

Luonnollisesti mitta olettaa, että näytteiden piirteet (komponentit) ovat metrisesti tulkittavissa, vieläpä globaalisti[2]. Lisäksi mitta käsittelee kaikkia komponentteja saman arvoisina, joten komponenttien arvot on normalisoitava, jotteivat suuret arvot dominoisi lopputulosta. Tyypillisistä probabilistisista menetelmistä poiketen metrinen mitta käsittelee muuttujia jatkuvina, mikä vaatii erityistä koodausta kategorisille ei-jatkuville muuttujille. Vektorirepresentaation valinta on siis syytä harkita tarkkaan, jotta näytteiden similariteettitulkinta ei vinoudu metriikassa. Yleisesti similariteettimitta toteuttaa harvoin metrisiä aksioomia, sillä erityisesti kolmioepäyhtälölle voi olla vaikea löytää luonnollista tulkintaa.

Euklidisen metriikan käyttö perustunee pitkälti sen luonnolliseen intuitiiviseen tulkintaan ryhmittelyssä. Näytteet kuuluvat samaan hahmoon kun ne ovat *lähellä* toisiaan; läheisyyden käsite on helppo visualisoida mielessään ensimmäisissä kolmessa ulottuvuudessa. Intuitio alkaa kuitenkin korkeammissa ulottuvuuksissa horjua. Yllättäen kontrasti läheisten ja kaukaisten pisteiden välillä pienenee nopeasti ulottuvuuksien kasvaessa [AHK01]. On luonnollista olettaa, että tasaisesti jakautuneessa datassa lyhimmän ja pisimmän välimatkan erotus kasvaisi lineaarisesti ulottuvuuksien lisääntyessä. Näin ei kuitenkaan euklidisen metriikan tapauksessa käy, vaan jo 20 ulottuvuuden jälkeen erotus ei enää merkittävästi kasva. Sen sijaan fraktionaalisilla metriikoilla kun $0 < k \leq 1$ kontrasti säilyy luotettavammin [AHK01].

Tulos on toki teoreettinen, mutta ei kaukaa haettu. Useat reaalimaailman aineistot sisältävät suunnilleen tasaisesti jakautuneen kohinakomponentin, joka käyttäytyy edellisten tulosten mukaisesti. Samoin empiiriset kokeet tukevat teoriaa [AHK01]. Käytännössä euklidisen metriikan tuottama relevanssi hahmoille voi olla hyvin arvaamaton. Kvantisoitu läheisyys metrisessä mielessä voi olla kaukana kvalitatiivisesta tulkinnasta, eikä metriseenkään läheisyyteen voi korkeissa ulottuvuuksissa varauksetta luottaa. Lisäesimerkkejä tästä ongelmasta antaa [Koh02].

Ratkaisuksi edellisiin ei kelpaa automaattinen ryhmittelyn laadun validointi. Validoijalle tulisi tällöin eksplikoida käyttäjän relevantteina pitämät ominaisuudet - jos tämä onnistuisi, mikä estää käyttämästä tätä eksplikaatiota suoraan hahmojen valintaperusteena?

Edelliset esimerkit osoittivat kuinka hankalaa relevanttien hahmojen aikaansaaminen voi olla perinteissä ryhmittelytraditiossa. Normaalijakaumia koskeva tarkastelu pätee myös muihin malliperheisiin, jotka jakavat samat perusongelmat. Euklidisen mitan käyttö periytyy muutamasta perusmenetelmästä, erityisesti *k-means* ja *SOM* [Koh95] algoritmeista, laajaan joukkoon näiden johdannaisia. Ongelma ei kuitenkaan korjaannu malliperhettä tai menetelmää vaihtamalla, sillä koko ryhmittelyprosessi piirteenirrotuksesta, esikäsittelystä, diskretoinnista representaation valintaan vaikuttaa implisiittisesti relevanssimäärittelyyn. Koko prosessin määrittely koherentisti relevanssin säilyttäen on suunnaton haaste. Oleellisesti ongelmaa voi helpottaa vain suoraviivaistamalla prosessia ja käyttämällä menetelmiä, jotka antavat käyttäjän määritellä relevanssin eksplisiittisesti.

2 Information Bottleneck

Edellisen luvun lähtökohta oli mahdollisimman neutraali: hahmokandidaatiksi hyväksyttiin mikä tahansa osajoukko näytteitä. Luvussa näytettiin kuinka epävarmalla pohjalla mielenkiintoisten osajoukkojen valinta perinteisessä ryhmittelytraditiossa usein on. Omine teoreettisine oletuksineen menetelmät toimivat moitteettomasti, mutta painottavat tulkitsijan näkökulmasta epäolennaisuuksia. Lopputuloksena potentiaalisesti mielenkiintoisia hahmoja

[2]Jos näytteen eräs piirre on pituus, on kahden avaruudessa lähekkäisen arvon esim. 160cm vs. 165cm vertailu mielekästä, mutta onko avaruuden äärilaidoilla olevilla näyteillä 5cm vs. 160km mielekästä tulkintaa samassa mielessä?

hukataan.

Ryhmittelyn juuret ovat syvällä informaatioteoriassa. Tietoliikenteen kannalta mielenkiintoinen tehtävä oli kvantisoida vektorijoukko mahdollisimman pienellä hävikillä mahdollisimman pieneen tilaan. Kysymys on häviöllisestä pakkauksesta; pyritään löytämään koodikirja (*codebook*), jonka sisältämät prototyyppivektorit minimoivat hävikin, jotta mitataan keskimääräisellä neliövirheellä. Ongelman ratkaisee generalisoitu *Lloyd-Max* -algoritmi eli *k-means*. Mitä vähemmän ja mitä "tiukempia" ryhmiä saadaan aikaan sitä pienempään tilaan (ja pienemmällä hävikillä) voidaan näytteet pakata. Ryhmittely helpottaa myös häviötöntä pakkausta, sillä itse näytteen tallettamisen sijaan voidaan tallettaa näytteen sijainti ryhmänsä keskipisteeseen nähden. Parhaimmillaan saavutettu etu pakkauksessa on huomattava.

Ehkä historian oikusta nykyinen tiedonlouhintatraditio pitää ryhmittelyongelmaansa samana. Kuitenkin eräs tiedonlouhinnan perusteesejä on, että käsiteltävä aineisto sisältää potentiaalisesti mielenkiintoisia hahmoja, mutta suinkaan kaikki näytteet eivät ole kiinnostavia. Käsiteltävä pakkausongelma saa täten uusia piirteitä: näytteet tulee pakata mahdollisimman tiivisti siten, että mahdollisimman paljon *oleellisesta* informaatiosta säilyy.[3] Ongelma on vastaava kuin esimerkiksi äänen tai kuvan häviöllisessä pakkauksessa. MP3 minimoi informaatiohävikin ihmisen kuuloaistin erottelukyvyn puitteissa, mutta hukkaa kaiken tämän ulkopuolella representaation minimoimiseksi. MP3 valitsee mielenkiintoiset hahmot datasta.

2.1 Motivaatio

Edellisen kappaleen muotoilu ryhmittelyongelmalle (häviölliselle pakkaukselle) on jo melkein teoreettisesti lähestyttävissä. Vain määrittely relevanssille uupuu. Koska mielenkiintoinen hahmo on kuvausvoimaltaan suuri ja on informatiivinen tutkittavan ilmiön kannalta, on relevanssiksi luonnollista valita ilmiö itsessään. Kuvataan ilmiötä satunnaismuuttujalla $Y \in \mathbf{Y}$ ja ryhmiteltäviä näytteitä satunnaismuuttujalla $X \in \mathbf{X}$. Oletetaan, että on olemassa stokastinen kuvaus joukolta $\mathbf{X}$ joukkoon $\mathbf{Y}$. Tämän kuvauksen olemassaolo on koko formalisoinnin perusta. Oletetaan, että X on (eräs) syy Y:lle. Nyt halutaan löytää $\mathbf{X}$:stä sellaiset hahmot, joilla mahdollisimman hyvin voi kuvata (ennustaa) $\mathbf{Y}$:tä. Hypoteesin mukaan hahmoja löytyy, sillä ennustus käsittelee tiettyjä hahmoja yhtäläisesti. Eli ennustus ei ole sensitiivinen yksittäisille näytteille vaan niiden muodostamille ekvivalenssiluokille (so. hahmoille) [Sha01].

Seuraavassa motivoidaan menetelmää konkreettisella esimerkillä dokumenttien ryhmittelystä. Varsinainen johdatus menetelmän soveltamisesta dokumenttiryhmittelyyn löytyy lähteestä [SlT00]. Merkitään D:lla dokumenttien joukkoa ja W:lla niissä esiintyvien sanojen joukkoa. Yleisesti oletetaan, että dokumenteissa esiintyvät sanat kuvaavat dokumentin aihetta. Tällöin dokumentti voidaan esittää sanojensa ehdollisena todennäköisyysjakaumana

$$p(w|d) = \frac{n(w|d)}{\sum_{w \in W} n(w|d)}, w \in W, d \in D, \tag{3}$$

missä $n(w|d)$ on sanan w esiintymiskertojen lukumäärä dokumentissa d. Samaan hahmoon halutaan dokumentit, joilla nämä jakaumat ovat samankaltaisia eli oletuksen mukaan kuvaavat tiettyä aihetta. Ei ole kuitenkaan ilmeistä, kuinka jakaumien samankaltaisuutta mitattaisiin välttäen ensimmäisessä luvussa mainitut ongelmat.

Edelliset jakaumat voidaan kuvata matriisina $D \times W$, jossa siis halutaan ryhmitellä rivit (dokumentit). Perinteisessä ryhmitteluyssä rivit ryhmiteltäisiin vertailemalla niitä toisiinsa

[3] Tästä näkökulmasta ryhmittely kytkeytyy algoritmiseen informaatioteoriaan ja Kolmogorov-kompleksisuuteen. Tästä voi kehitellä myös perustellun formalisoinnin kausaliteetille [Sha01]. Viaton ryhmittely saa arvaamattoman syvällisiä piirteitä.

	w_1	w_2	w_3
d_1	0.5	0.5	0.0
d_2	0.5	0.48	0.02
d_3	0.0	0.02	0.98

Taulukko 1: Esimerkki dokumenttien ja sanojen yhteisjakaumasta

jollain valitulla samankaltaisuusmitalla. Merkitään näitä rivejä järjestämällä saatuja dokumenttihahmoja $\bar{D}$. Jotta lähestymistapojen erot tulisivat selville, katsotaan yksityiskohtaisesti mitä *Information Bottleneck* -kustannusfunktio oikein optimoikaan.

Information Bottleneck ei oleta annetuksi perinteisten ryhmittelymenetelmien tapaan jotain erillistä samankaltaisuusmittaa. Tämän sijaan oletetaan tunnetuksi jokin yhteisjakauma $p(x, y)$, jossa ryhmiteltevä muuttuja (tässä dokumentit D) on toisena jäsenenä. Dokumenttiryhmittelyn tapauksessa toinen jäsen on luontevasti sanojen joukko W.

Taulukko 1. kuvaa esimerkkitilannetta: Jos tehtävänä on ennustaa sanojen esiintymisiä, ei rivien d_1 erottelu d_2 tuo tehtävään juuri mitään apua[4]. Tämä johtuu luonnollisesti jakauminen samankaltaisuudesta. Koska siis ennustuskyky ei muutu samankaltaisia rivejä sulauttaessa, voidaan ennustuskyvyn hävikkiä käyttää luontevasti kustannusfunktiona ryhmittelyssä. Itse ryhmittelytehtävän kannalta tämä valinta ei oleellisti muuta mitään – tässäkin lopputuloksena ovat samankaltaisten rivien muodostamat hahmot. Kuitenkin teoreettisesti ja käytännössäkin on lähestymistapa mielivaltaista mittaa perustellumpi: Samankaltaisuusmitta on datalähtöinen ja käyttäjä voi vaikuttaa suoraviivaisesti siihen, mitä hahmoja pidetään tärkeinä valitsemalla haluamansa yhteisjakauman. Ennenkaikkea menetelmä ei sisällä oletuksia datan generoivasta mallista tai avaruuden rakenteesta.

Seuraavassa esitetään saadulle hyvinmääritellylle ryhmittelyongelmalle formalisointi *Information Bottleneck* -menetelmän [TPB99] mukaan. Esitys olettaa informaatioteorian peruskäsitteet tunnetuksi, johdatusta alaan antaa mm. [Mac02].

2.2 Formalisointi

Annettuna kaksi satunnaismuuttujaa $\mathbf{X}$ ja $\mathbf{Y}$. Halutaan löytää stokastinen kuvaus $p(z|x)$, missä $z \in \mathbf{Z}$ on koodisana koodikirjassa, jokaiselle $x \in \mathbf{X}$. Kuvaus muodostaa pehmeän ryhmittelyn näytteiltä koodikirjalle (hahmojoukkoon). Kuvauksen säilyttämää informaatiota voidaan mitata keskinäisinformaatiolla (*mutual information*, MI), joka määritellään yleisesti

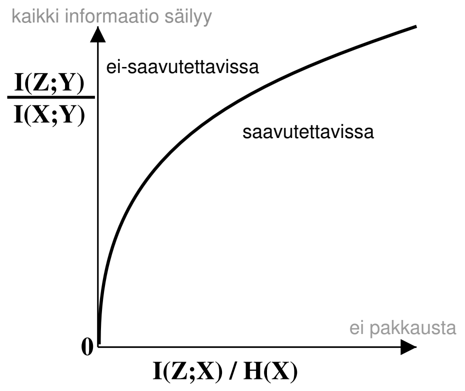
$$I(A; B) = H(A) - H(A|B) = H(B) - H(B|A). \tag{4}$$

Tässä H on Shannonin entropia. MI on siis symmetrinen. Suurimmillaan MI on silloin kun kumpikin muuttuja sisältää paljon informaatiota yksinään, mutta yhdessä ne ovat redundantteja. $I(A; B)$ siis kertoo, paljonko muuttujan A tietäminen vähentää muuttujan B epävarmuutta eli paljonko A sisältää infomaatiota B:stä. Erityisesti keskinäisinformaatio lähtö- ja kuvajoukon välillä on siis

$$I(X; Z) = \sum_{x \in X} \sum_{z \in Z} p(x, z) \log \frac{p(z|x)}{p(z)}. \tag{5}$$

Koska $I(X; Z)$ kuvaa yleisesti pakkauksessa säilyvää informaatiota, halutaan tätä minimoida mahdollisimman kompaktin representaation aikaansaamiseksi.

[4]Muodollisesti siis $I(\{d_1 \cup d_3\}; W) \lessapprox I(\{d_1 \cup d_2 \cup d_3\}; W)$.



Kuva 1: Esimerkki hävikkifunktiosta

Tähän asti tilanne vastaa alkuperäistä vektorikvantisaatiota, joskin on määriteltävä vielä hävikkimitta kuvaamaan $p(z|x)$:n kadottamaa informaatiota. Yhdessä $I(X;Z)$ ja hävikkimitta muodostavat minimoitavan funktion. Alkuperäinen teoreettinen muotoilu tälle minimointiongelmalle on esitetty Shannon-Kolmogorovin hävikkiteoriassa (*rate-distortion theorem*), joka antaa yleiselle tapaukselle seuraavan muotoilun

$$\mathcal{F}[p(z|x)] = I(X;Z) + \beta \langle d(x,z) \rangle_{p(x,z)}, \tag{6}$$

missä $\mathcal{F}$ kuvaa optimoitavaa funktiota. Tässä $d(x,z)$ on haluttu hävikkimitta ja β parametri (Lagrangen kerroin), jolla tasapainoa säilytettävän informaation ja pakkauksen välillä voi säädellä.

Information Bottleneck -menetelmän ydinajatus on määritellä mielekäs, yleinen ja perusteltu hävikkimitta. Alkuperäisen motivaation mukaisesti halutaan säilyttää maksimaalinen määrä informaatiota ennustettavasta muuttujasta. Informaation säilymistä ryhmittelyssä voidaan mitata jälleen keskinäisinformaatiolla, nyt ennustettavan muuttujan ja koodikirjan välillä

$$I(Z;Y) = \sum_y \sum_z p(y,z) \log \frac{p(y,z)}{p(y)p(z)}. \tag{7}$$

Luonnollisesti säilytettävä informaatio on kompromissi pakkauksen tiiviyden kanssa (ks. kuva 1). Kuvassa pystyakseli kuvaa säilyvää informaatiota ja vaaka-akseli häviöllisen pakkauksen tasoa. Hävikkifunktio ei ole lineaarinen, vaan datakohtaisesti ylöspäin kupera kuvan esittämällä tavalla.

Lopullinen optimointiongelma saa siis muodon

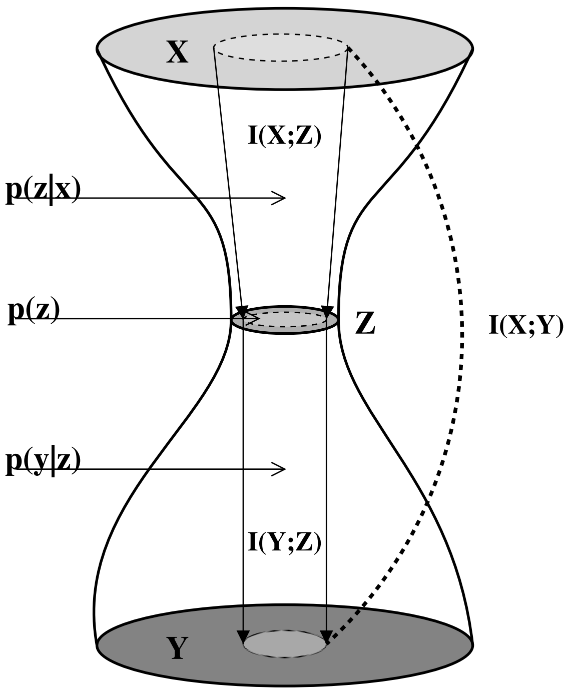
$$\mathcal{F}[p(z|x)] = I(X;Z) - \beta I(Z;Y). \tag{8}$$

Tästä on helppo nähdä, että kun $\beta = 0$ on kvantisaatio mielivaltainen suhteessa muuttujaan Y. Toisaalta kun $\beta \to \infty$ voidaan relevanttia informaatiota säilyttää mielivaltaisen paljon.[5] Kuvassa 1. parametrin β vaihtelu vastaa kulkua käyrällä. Käy ilmi, että edelliselle optimointiongelmalle löytyy eksakti ratkaisu, sillä

$$\frac{\delta \mathcal{F}}{\delta p(z|x)} = 0, \text{kun } p(z|x) = \frac{p(z)}{Q(x,\beta)} exp(-\beta D_{KL}(p(y|x) \parallel p(y|z))). \tag{9}$$

Ratkaisussa ei Kullback-Leibler divergenssiä erikseen oletettu, vaan se muodostuu osana optimointiongelman ratkaisua. Tässä mielessä se on luonnollinen valinta hävikkimitaksi, eli $d(x,z) = D_{KL}(p(y|x) \parallel p(y|z))$.

[5]Tilanteessa $\beta \to \infty$ itseasiassa Z pitää sisällään kaiken informaation (*sufficient statistics*) Y:n ennustamiseen.[Sha01]



Kuva 2: *Information Bottleneck* -kaavojen suhteet

Johdos päätyy seuraavaan yhtälöryhmään

$$p(z) = \sum_{x \in X} p(x)p(z|x) \tag{10}$$

$$p(y|z) = \frac{1}{p(z)} \sum_{x \in X} p(z|x)p(x)p(y|x) \tag{11}$$

$$p(z|x) = \frac{p(z)}{Q(x, \beta)} exp(-\beta D_{KL}(p(y|x) \parallel p(y|z))). \tag{12}$$

Lisäksi oletetaan dokumenttien priorijakauma tasaiseksi, $p(x) = \frac{1}{|X|}$.

Yhteenvetona *Information Bottleneck* -menetelmän keskeiset käsitteet on koottu kuvaan 2. Vaalea soikio kuvaa relevanttia informaatiota, joka halutaan säilyttää mahdollisimman täydellisenä. Joukot X ja Y voivat myös sisältää paljon ryhmittelyn kannalta epärelevanttia informaatiota (tummat soikiot vaalean ympärillä). Hahmojen joukko Z toimii pullonkaulana sisältäen vain X:n relevantin osan. Keskinäisinformaatioiden suhteen tavoitellaan tilannetta $I(Y;Z) \lessapprox I(X;Z) \lessapprox I(Y;X)$.

2.3 Mielekkyys

Palataan aluksi esimerkkiin dokumenttiryhmittelystä. Tavoitteenahan oli tilanne $I(\bar{D};W) \lessapprox I(D;W)$ eli hahmojen tulisi säilyttää ennustuskykynsä sanoihin nähden, mutta mitä tämä oikeastaan käytännössä tarkoittaa? Puretaan saatuja kaavoja, alkaen keskinäisinformaatiosta dokumenttien ja sanojen välillä

$$I(D;W) = \sum_{d \in D} \sum_{w \in W} p(d)p(w|d) \log \frac{p(w|d)}{p(w)}. \tag{13}$$

Koska dokumenttien priorijakauma $p(d)$ oletetaan tasaiseksi, erottelee $p(w|d)$ dokumentit. Yhdelle dokumentti-sana -parille keskinäisinformaatio on suurimmillaan kun sana esiintyy vain tässä dokumentissa; siis koko $I(D;W)$ on suurimmillaan kun dokumenttien sanastoissa ei ole lainkaan päällekäisyyttä. Luonnollisesti tällaista materiaalia ei voi ryhmitellä eikä pakata. Keskinäisinformaation säilyttäminen on kuitenkin hyvin perusteltua, sillä sen säilyminen takaa dokumenttien erottuvuuden so. luokkien säilymisen. Vastaavasti voidaan kaava kirjoittaa auki hahmoille $\bar{D}$ ja sanoille,

$$I(\bar{D};W) = \sum_{\bar{d}\in\bar{D}} \sum_{w\in W} p(\bar{d})p(w|\bar{d}) \log \frac{p(w|\bar{d})}{p(w)}. \tag{14}$$

Tähän kaavaan voidaan soveltaa *Information Bottleneck*:n kaavoja 10-12 (merkitään $z = \bar{d}$, $x = d$ ja $y = w$). Lisäksi oletetaan käytettävän kovaa ryhmittelyä, eli $p(z|x) \in \{0,1\}$. Nyt suoraviivaisen siistimisen jälkeen saadaan,

$$I(\bar{D};W) = \sum_{\bar{d}\in\bar{D}} \sum_{w\in W} \left(\sum_{d\in\bar{d}} p(d)p(w|d) \right) \log \frac{\frac{1}{|\bar{d}|}\sum_{d\in\bar{d}} p(w|d)}{p(w)}. \tag{15}$$

Tämä muistuttaa suuresti kaavaa 13, nyt vain tekijät lasketaan kaikista tietyn hahmon dokumenteista yksittäisten dokumenttien sijaan. Oleellista on huomata kaavassa 3 määritellyn ehdollisen todennäköisyyden $p(w|d)$ asema, joka määrittelee jakauman. Jotta kaavat 15 ja 13 tuottaisivat suunnilleen saman tuloksen, kuten toivotaan, on hahmon sisällä tietyn sanan lukumäärän suhde kaikkiin sanoihin oltava lähes sama kuin hahmon sisältämillä dokumenteilla. Eli hahmon sisällä dokumenttien sanajakaumat eivät saa juurikaan vaihdella eli jakaumien on oltava mahdollisimman samankaltaisia.

Kannattaa huomata, ettei *Information Bottleneck* sisällä luonnostaan Occamin-partaveitsi -periaatetta, joka voisi auttaa ryhmien lukumäärän valinnassa. Mitä enemmän ryhmiä on sitä uskollisemmin informaatiota säilyy.

Kun epäyhtälö $I(Y;Z) \lessapprox I(Y;X)$ lähenee asymptoottisesti ekvivalenssia, voidaan olettaa, että valitut hahmot kertovat ilmiöstä täsmälleen sen mitä koko data asiasta voi sanoa. Tämä tekee kustannusfunktiosta hyvin mielekkään tulkitsijan kannalta. Kannattaa kuitenkin huomata, että tarkasteltavan ilmiön ja ryhmiteltävän datan suhde on probabilistinen ja perustuu yhteisjakaumaan $p(x,y)$. Tämä on kaikki mitä menetelmä tietää maailman olemuksesta, mikä ei aina riitä.

Teoreettinen muotoilu on suhteellisen läpinäkyvä, eikä mielivaltaisia oletuksia datan generoivasta mallista tai similariteettimitoista tehdä. On kuitenkin otettava huomioon, ettei tähänkään menetelmään sisälly taikaa. Huonolla onnella otokset $\mathbf{X}$ ja $\mathbf{Y}$ ovat epäluotettavia ja tulokset sen mukaisia. Menetelmä ei myöskään vapauta käyttäjää ajattelusta, sillä ennustettavan muuttujan Y huolimaton valinta voi käytännössä johtaa mielivaltaisiin tuloksiin. Esimerkkinä tästä edellämainittu sanasäkkirepresentaatio, jossa Y:ksi valittiin sanojen joukko. Kuitenkaan käyttäjän relevanssi on dokumenttien ryhmittely aiheittain, ei sanoittain, joten tulokset voivat olla arvaamattomia.

3 Sarjallinen ryhmittely

Edellisessä määriteltiin Information Bottleneck -kustannusfunktio ryhmittelylle. Muotoilussa ryhmittely nähtiin stokastisena kuvauksena näytteiden joukolta koodikirjaan, missä $p(z|x)$ on jakauma yli hahmojen. Ryhmittelytraditiossa tälle on useita nimiä: *soft clustering*, *fuzzy clustering* tai *convex combination*. Ryhmittelyalgoritmin kannalta tämä lähestymistapa on kuitenkin kovaa klusterointia, jossa $p(z|x) \in \{0,1\}$, ratkaisevasti vaikeampi. Informaatiota edellinen sisältää jälkimmäistä paljon enemmän, mikä tekee puolestaan koodikirjan kardinaliteetin (hahmojen lukumäärän) valinnan huomattavasti työläämmäksi.

Käytännön syistä päädytäänkin käyttämään kovaa klusterointia. Tästäkin huolimatta IB:tä käyttävän hierarkisen ryhmittelyn aikavaatimus on kuutiollinen [SlT99], mikä tekee menetelmän soveltamisen mahdottomaksi. Seuraavassa esitellään tuore algoritmi, jonka luvataan keskimäärin suoriutuvan huomattavasti tätä paremmin.

Hierarkinen ryhmittely on houkutteleva ratkaisu hahmojen etsintäongelmaan, sillä hahmojen lukumäärää ei tarvi etukäteen kiinnittää. Kuitenkin laskennallisesti lähestymistapa on usein kestämätön. Lisäksi jos pisteytysfunktio (ajatellaan nyt ongelmaa pisteytysfunktion (*score function*) maksimointina, ei kustannusfunktion minimointina) tähtää maksimaaliseen informaation säilyttämiseen kohdemuuttujasta mahdollisimman kompaktilla representaatiolla (vrt. Occamin partaveitsi), tulee hahmojen lukumäärä implisiittisesti kiinnitettyä.

Seuraavassa esiteltävä sarjallinen ryhmittely [SFT02] ei tähtää optimaaliseen tulokseen, vaan etsii ahneesti pisteytysfunktion lokaalisti maksimoivan näytejakelun K:lle hahmolle. Algoritmin idea on yksinkertainen: Edetään näyte näytteeltä iteratiivisesti ja sulautetaan näyte hahmoon, joka maksimoi pisteytysfunktion näytteen ja hahmon välillä. Selkeyden vuoksi ajatellaan tarkasteltava näyte omaksi yksialkioiseksi hahmokseen, jolloin voidaan puhua vain hahmojen pisteytyksestä.

Pisteytysfunktiota merkitään $\mathcal{S}(Z)$ hahmoille Z. Sanotaan, että $\mathcal{S}$ on pilkottavissa jos $\mathcal{S}(Z)$ voidaan määritellä yksittäisten hahmojen $Z = \{z_1, ..., z_K\}$ summana $\mathcal{S}(Z) = \sum_i \mathcal{S}(\{z_i\})$. Tässä tapauksessa kahden hahmon sulautuksen vaikutus lopulliseen pisteytykseen voidaan laskea

$$d_{\mathcal{S}}(z_i, z_j) = \mathcal{S}(\{z_i\} \cup \{z_j\}) - \mathcal{S}(\{z_i\}) - \mathcal{S}(\{z_j\}). \tag{16}$$

Algoritmi alustetaan jakamalla näytteet $\mathbf{X}$ satunnaisesti hahmoihin $Z = \{z_1, ..., z_K\}$. Jokaisella askeleella irrotetaan joku näyte hahmostaan $z(x)$ omaksi hahmokseen. Tämän jälkeen sulautetaan valittu näyte uuteen hahmoon z^{uusi} siten, että $z^{uusi} = \arg\min_{z \in Z} d_{\mathcal{S}}(x, z)$. Sulautus tuottaa uuden hahmojoukon Z^{uusi}, jossa hahmojen lukumäärä on taas haluttu K. Sulautus kasvattaa hahmojoukon pisteytystä, sillä selvästi $\mathcal{S}(Z^{uusi}) \geq \mathcal{S}(Z)$. Iterointia jatketaan kunnes saavutetaan kierroksille annettu katto $maxL$ tai vain $\epsilon|X|$ näytettä vaihtaa hahmoaan.

Algoritmin aikavaativuus on luokkaa $O(LKD|X|)$, missä L on kierrosten lukumäärä, K hahmojen, D näytteen x ulottuvuus ja $|X|$ näytteiden lukumäärä. Kuten muutkin satunnaisjakelusta aloittavat algoritmit, on tämäkin syytä ajaa useampaan kertaan ja valita ajoista maksimaalisen $\mathcal{S}(Z)$ antava jakelu. Koska tyypillisesti $LKD \ll |X|$, on aikavaatimus neliöllistä parempi.

Algoritmi ei ota kantaa pisteytysfunktioon $\mathcal{S}$, kunhan se on pilkottavissa. Kun pisteytysfunktioksi valitaan Information Bottleneck, saadaan $\mathcal{S}(Z) = I(Z;Y)$ ja näytteen x representaatioksi $p(x, y)$. Käytännössä saadaan

$$d_{\mathcal{S}}(x, z) = (p(x) + p(z)) \cdot JS(p(y|x), p(y|t)), \tag{17}$$

missä JS(p,q) on Jensen-Shannonin divergenssi[6], joka määritellään

$$JS(p, q) = \pi_1 D_{KL}(p \parallel \bar{p}) + \pi_2 D_{KL}(q \parallel \bar{p}), \tag{18}$$

[6]Jensen-Shannon -divergenssillä on käytännöllinen symmetriaominaisuus toisin kuin Kullback-Leibler -divergenssillä

missä p,$\bar{p}$ ja q ovat kontekstissa

$$\{p, q\} \equiv \{p(y|x), p(y|z)\}$$
$$\{\pi_1, \pi_2\} \equiv \{\frac{p(x)}{p(x)+p(z)}, \frac{p(z)}{p(x)+p(z)}\}$$
$$\bar{p} = \pi_1 p(y|x) + \pi_2 p(y|t). \tag{19}$$

3.1 Vertailua

Sarjallinen ryhmittely muistuttaa osin *k-means*:ia. Oleellinen ero on hahmojen päivityksessä. *K-means* päivittää hahmot kaikille näytteille rinnakkain, siinä missä sarjallinen ryhmittely päivittää hahmot näyte näytteeltä. Tästä johtuen *k-means* ei voi hyödyntää kaavan 9 kaltaista konstruktiota osoittaakseen, että pistefunktion arvo todellakin kasvaa iteraatioiden myötä. Sarjallisella ryhmittely jakaa myös ominaisuuksia (inkrementaalisen) EM-algoritmin kanssa.

Sarjallista ryhmittelyä testattiin dokumenttiryhmittelykokeilla, jossa Reuters-materiaalia ja uutisryhmiä ryhmiteltiin aiheittain. Materiaali oli esiluokiteltua, joten tuloksia voitiin mitata oikeisiin ryhmiin osuneiden dokumenttien prosenttiosuutena. Ryhmien lukumäärä asetettiin vastaamaan luokkien todellista lukumäärää. Tuhannen dokumentin aineistolla sarjallinen ryhmittely Information-Bottleneck -kustannusfunktiolla (sIB) tuottaa parikymmentä prosenttiyksikköä parempia tuloksia kuin *k-means*, kun käytössä on sanafrekvenssirepresentaatio. Kun representaatio vaihdetaan tf-idf:ään, supistuu ero alle kymmeneen prosenttiyksikköön. Tulokset ovat kuitenkin keskiarvoja useasta datasta, joista kukin ryhmiteltiin useampaan kertaan ja tuloksista valittiin paras. *K-means*:in datakohtaiset tulokset vaihtelivat sIB:a enemmän. Ehkä hieman yllättäen kaikissa kokeissa sIB:n tulokset oli muutaman prosenttiyksikön päässä naiivi-Bayes luokittelijasta. Tarkempi kuvaus koeasetelmasta löytyy lähteestä [SFT02].

Aineiston sisältäessä vain 500 dokumenttia, on kaula sIB:n ja muiden menetelmien (naiivi-Bayestä lukuunottamatta) välillä merkittävä. *K-means*:n tulokset jäävät tässä asetelmassa keskimäärin yli kolmekymmentä prosenttiyksikköä heikommiksi. Nämä tulokset kertovat jotain myös dokumenttiryhmittelyn laadusta. Data on kohinaista, mutta tietyt termit korreloivat kuitenkin voimakkaasti tiettyjen aihepiirien kanssa. Vaikuttaakin siltä, että erityisesti kohinaisessa tilanteessa, jossa *k-means* ei kykene hahmoja erottamaan, on eksplikoidusta relevanssista hyötyä. Kannattaa kuitenkin huomata, että tf-idf -representaatio on kehitetty juuri parantamaan aihepiirien erottelukykyä metrisissä avaruuksissa, mikä voidaan tulkita mutkan kautta relevanssieksplikaatioksi. Ilman tätä painotusta ovat tulokset heikkoja.

Kannattaa huomata, että koeasetelmassa pyrittiin dokumentteja ryhmittelemään aihepiireittäin sanajakaumiensa mukaan. Sanajakaumat eivät välttämättä kuvaa millään tapaa dokumenttien sisältöä tai aihepiiriä, joskin tietyillä aineistoilla aihepiirien ja sanaston välillä on selkeä korrelaatio. Koko totuutta ryhmittelymenetelmän tai kustannusfunktion luonteesta tämänkaltainen koe ei kerro, mutta on kyllä tulkittavissa suhteessa muihin samaa informaatiota käyttäviin menetelmiin.

4 Teknisiä huomioita

On vaikea sanoa kertovatko edellä esitetyt tulokset ennemmin algoritmin herkkyydestä lokaaleille minimeille vai kustannusfunktion laadusta. Selvää on, että pahimmillaan tulokset voivat jäädä kauaskin optimaalisesta. Toinen, tähän liittyvä, avoin kysymys on algoritmin konvergointinopeus. Alkuperäinen lähde [SFT02] ei ota tähän kantaa, mutta alustavasti tekijöiden suosittamilla lopetustestin arvoilla vaikuttaisi algoritmin konvergointinopeus olevan EM-algoritmin luokkaa.

Kustannusfunktiossa käytetty Jensen-Shannon divergenssi kuvaa uskottavuutta, että kaksi jakaumaa ovat peräisin samasta lähteestä. Tässä mielessä algoritmi toimii liittäen näytteitä yksi kerrallaan todennäköisimpään lähteeseensä. Olisi mielenkiintoista tietää, mikä menetelmän suhde on ryhmittelyyn EM-algoritmilla multinomijakaumaoletuksin. Ilmeisesti Kullback-Leibler -divergenssin käytöllä hävikkimittana on läheinen suhde suurimman uskottavuuden (*maximum likelihood*) maksimointiin [SiK02].

Kiusallisinta esitetyssä algoritmissa on luopuminen pehmeästä klusteroinnista ja hahmojen lukumäärän (mielivaltainen) kiinnittäminen. Nämä oletukset väistämättä jättävät tuloksen kauas optimaalisesta. Hieman toisenlainen lähestymistapa ryhmittelyyn eksplikoiduilla relevansseilla on *Learning metrics* [SiK02], joka pitää jakaumaa $p(z|x)$ jatkuvana. Muutenkin kannattaa huomata, että oleellista *Information Bottleneck* -menetelmässä on ennemminkin sen teoreettinen muotoilu kustannusfunktiona ja ajatus datalähtöisestä samankaltaisuusmitasta. Sarjallinen ryhmittely on eräs käytännöllinen menetelmä tämän hyödyntämiseen, muttei välttämättä lainkaan paras mahdollinen. Tästä huolimatta myös konkreettiset ryhmittelytulokset ovat lupaavia.

Kuriositeettina mainittakoon vielä teknisesti mielenkiintoinen yhteys laskennalliseen mekaniikkaan (*computational mechanics*), jossa aivan toisista lähtökohdista aivan toisin tavoittein (ei mitään ryhmittelyyn liittyvää), päädytään muistittomia transduktereita hyväksikäyttäen muotoilemaan malli, joka suuresti muistuttaa Information Bottleneckia [Sha01]. Information Bottleneckista poiketen tässä mallissa pyritään löytämään pienin optimaalinen joukko Z, mikä vastaa aiemmassa muotoilussa tilannetta $\beta \to \infty$. Tämä muotuilu tarjoaisi kätevän ylärajan hahmojen lukumäärälle K, joten tältäkin suunnalta voisi IB:a käyttävää ryhmittelyä kehittää.

Viitteet

[AHK01] Aggarwal C., Hinneburg A., Keim A., On the surprising behavior of distance metrics in high dimensional space, *Proc. 8th International Conference on Database Theory*, Lontoo, Iso-Britannia, 2001, 420-434.

[Aut02] Autio I., Sekoitemallit ja EM-algoritmi klusteroinnissa, Teoksessa *Klusterointimenetelmät-seminaari*, Raportti C-2002-54, Helsingin yliopisto, tietojenkäsittelytieteen laitos, marraskuu 2002, 127-138.

[Koh02] Kohonen J., Aliavaruusklusterointi, Teoksessa *Klusterointimenetelmät-seminaari*, Raportti C-2002-54, Helsingin yliopisto, tietojenkäsittelytieteen laitos, marraskuu 2002, 21-31.

[Koh95] Kohonen T., *Self-Organizing Maps.* Springer, Berliini, 1995.

[Mac02] MacKay D.J.C., *Information Theory, Inference and Learning Algorithms.* Versio 2.4.1. Julkaisematon teos, http://www.inference.phy.cam.ac.uk/mackay/itprnn/book.html. [25.2.2002]

[Sha01] Shalizi C., *Causal Architecture, Complexity and Self-Organization in Time Series and Cellular Automata*, väitöskirja, Univerity of Wisconsin, Madison, 2001.

[SiK02] Sinkkonen J., Kaski S., Clustering based on conditional distributions in an auxiliary space. *Neural Computation*, 14, 2002, 217-239.

[SlT99] Slonim N., Tishby N., Agglomerative Information Bottleneck. *Advances in Neural Information Processing Systems*, 12, 1999, 617-623.

[SlT00] Slonim N., Tishby N., Document clustering using word clusters via the information bottleneck method. *Proc. 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Ateena, Kreikka, 2000, 208-215.

[SFT02] Slonim N., Friedman N., Tishby N., Unsupervised document classification using sequential information maximization. *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, Suomi, 2002, 129-136.

[TPB99] Tishby N., Pereira F., Bialek W., The Information Bottleneck Method. *Proc. 37th Allerton Conference on Communication and Computation*, 1999.