TEKNILLINEN KORKEAKOULU
Tietotekniikan osasto
Teknillisen fysiikan koulutusohjelma

Mikko Kurimo

Adaptiivisten vektorikvantisointi-menetelmien ja kätkettyjen Markov-mallien kombinaatioita puheentunnistuksessa

Diplomityö tehty teknillisen matematiikan syventymiskohteessa
Työn valvoja vs.prof. Jukka Ruusunen
Työn ohjaaja TkT Kari Torkkola

Otaniemi 24.1.1992

TEKNILLINEN KORKEAKOULU **DIPLOMITYÖN TIIVISTELMÄ**

Tekijä:	Mikko Kurimo
Työn nimi:	Adaptiivisten vektorikvantisointimenetelmien ja kätkettyjen Markov-mallien kombinaatioita puheentunnistuksessa
Päivämäärä:	24.1.1992 **Sivumäärä:** 80
Osasto:	Tietotekniikan osasto
Professuuri:	Mat-2 Sovellettu Matematiikka
Laitos:	Matematiikan ja systeemianalyysin laitos
Työn valvoja:	vs. prof. Jukka Ruusunen
Työn ohjaaja:	tekn. tri. Kari Torkkola

Tässä työssä tutkitaan jatkuvan havaintojakauman kätkettyjen Markov-mallien yhdistämistä adaptiiviseen vektorikvantisointiin perustuvaan puheentunnistusjärjestelmään. Tutkimus liittyy Teknillisen korkeakoulun informaatiotekniikan laboratoriossa tutkijaprofessori Teuvo Kohosen johdolla tehtävään puheentunnistustutkimukseen.

Foneemit mallitetaan peräkkäisinä puhesignaalin tiloina. Tilat määritetään puhesignaalista laskettujen hetkellisten kepstrikertoimien muodostamien piirrevektorien ja tilojen välisten siirtymien todennäköisyysjakaumien avulla. Piirrevektorijakaumaa approksimoidaan monen osittain päällekkäisen normaalijakauman yhdistelmällä. Mallit opetetaan kerätyn puheaineiston perusteella käyttäen vektorikvantisointimenetelmiä normaalijakaumien huippujen optimaaliseen sijoitteluun ja iteratiivista Baum-Welch –estimointia muiden parametrien määrittämiseen.

Tehdyt kokeet osoittavat, että piirrevektorien todennäköisyysjakaumat puhesignaalin tilamallissa ovat mutkikkaita ja niiden approksimaatiot vaativat paljon parametreja. Parhaaksi strategiaksi osoittautui keskittyminen enemmän jakaumien huippujen paikallistamiseen kuin huippujen muodon kuvaamiseen. Itseorganisoituvan piirrekartan ja oppivan vektorikvantisoinnin käyttö Baum-Welch –estimoinnin ohella jatkuvien Markov-mallien opetuksessa tuotti parempia tunnistustuloksia kuin pelkkä Baum-Welch –opetus. Tunnistuksessa saadut tulokset olivat tarkempia kuin puheentunnistusjärjestelmässä tällä hetkellä käytössä olevilla diskreetin havaintojakauman Markov-malleilla, mutta eivät kuitenkaan täysin vertailukelpoisia.

Tämä diplomityö on tehty Teknillisen korkeakoulun informaatiotekniikan laboratoriossa vuonna 1991. Haluan kiittää tutkijaprofessori Teuvo Kohosta työni rahoittamisesta ja mahdollisuudesta työskennellä laboratoriossa, TkT Kari Torkkolaa diplomityöni ohjaamisesta sekä kaikkea laboratorion henkilökuntaa hauskasta työskentelyilmapiiristä ja lukuisista hyvistä neuvoista. Lisäksi kiitän matematiikan ja systeemianalyysin vs. professoria TkT Jukka Ruusuista työn valvojana toimimisesta.

Otaniemessä, 24.1.1992

Mikko Kurimo

Sisällysluettelo

1 Johdanto **9**

1.1 Puheentunnistuksen nykytila 9

1.2 Merkittävimpiä puheentunnistusjärjestelmiä 9

1.3 Puheentunnistustutkimus TKK:n informaatiotekniikan laboratoriossa . . . 10

1.4 Kätketyt Markov-mallit puheentunnistuksessa 10

1.5 Vektorikvantisointi foneemien mallittamisessa 10

1.6 Tämän työn sisältö 11

2 Puheentunnistus **12**

2.1 Puhe 12

2.1.1 Puhuminen 12

2.1.2 Puhesignaali 12

2.1.3 Kuuleminen 14

2.1.4 Puheen ymmärtäminen 14

2.2 Foneemit 14

2.2.1 Foneemien luokittelu 15

2.2.2 Foneemit eri kielissä 16

2.2.3 Foneemien erottaminen toisistaan 16

2.2.4 Foneemien kirjoittaminen 16

2.3 Puheentunnistimien luokittelu 16

2.3.1 Puheen kerääminen 16

2.3.2 Kieli 17

2.3.3 Sanavarasto 17

2.3.4 Tunnistettavat osat 18

2.3.5 Puhujan valinta 19

2.4 Esimerkki puheentunnistusjärjestelmästä – SPHINX 19

2.5 Automaattinen puheentunnistus 19

2.6 Puheentunnistuksen historiaa 20

2.7 Puheentunnistimen sovelluksia 21

3 Otaniemen puheentunnistusjärjestelmä **22**

3.1 Järjestelmän kehitys 22

3.2 Laitteisto 23

3.3 Toimintaperiaate 23

3.3.1 Yleistä 23

3.3.2 Puhesignaalin esikäsittely 23

3.3.3 Piirteiden laskeminen 24

3.3.4 Kvasifoneemien määrittäminen 25

3.3.5 Foneemien tunnistaminen kvasifoneemijonosta 26

3.3.6 Tulosten parantaminen paikallisen kontekstin avulla 27

3.4 Suorituskyky 27

4 Kätketyt Markov-mallit **29**

4.1 Mallin kuvaus 29
4.2 Esimerkki kätketystä Markov-mallista 30
4.3 Havaintojonon todennäköisyys 32
4.4 Todennäköisin tilajono 32
4.5 Parametrien estimointi 33
4.6 HMM:t puheentunnistuksessa 35

5 Vektorikvantisointi ja itseorganisoituvat piirrekartat 37

5.1 Piirrekartat ja vektorikvantisointi 37
5.1.1 Piirteet 37
5.1.2 Vektorien samanlaisuuden mittaaminen 38
5.1.3 Vektorikvantisointi 39
5.1.4 Kvantisointikoodikirjan koko ja muoto 40
5.1.5 Koodikirjalta vaadittavat ominaisuudet 41
5.2 Itseorganisaatio 41
5.2.1 Itseorganisointialgoritmi 41
5.2.2 Topologinen ympäristö 42
5.2.3 Opetusvakion merkitys 43
5.3 Oppiva vektorikvantisointi eli LVQ 44
5.3.1 LVQ:n päämäärä 44
5.3.2 LVQ1 44
5.3.3 LVQ2 44
5.3.4 LVQ3 45

6 Vektorikvantisoinnin yhdistäminen jatkuviin Markov-malleihin 46

6.1 Miksi? 46
6.2 Puheen foneettisen kuvauksen tuottaminen 47
6.2.1 Tunnistuksen vaiheet 47
6.2.2 Koemateriaali 48
6.2.3 Menetelmän testaus 48
6.3 Puheen foneettinen malli 49
6.3.1 Sanan malli 49
6.3.2 Foneemin malli 50
6.4 Piirrevektorin tiheysfunktio 51
6.4.1 Tiheysfunktion approksimointi normaalijakaumien avulla 51
6.4.2 Yksinkertaistettu malli 53
6.4.3 Yhteisen kovarianssimatriisien käyttö osajakaumien välillä 54
6.4.4 Tiheysfunktioapproksimaatioiden vertailu 54
6.5 Vertailua vektorikvantisointiin 55
6.5.1 Koodikirjavektorit ja normaalijakaumat 55
6.5.2 Kvantisointivirhe ja havaintotodennäköisyys 56
6.5.3 Koodikirjan osittaminen 56
6.6 Mallin parametrien opettaminen 57
6.6.1 Siirtotodennäköisyyksien alkuarvot 57
6.6.2 Jakaumien itseorganisointi 57
6.6.3 Erottelukyvyn parantaminen LVQ:n avulla 60
6.6.4 Parametrien uudelleenestimointi 62
6.6.5 Foneemitransitioiden estimointi 66
6.7 Todennäköisimmän tilajonon käyttö opetuksessa 67

6.7.1 Todennäköisin havainnot tuottanut tilajono 67

6.7.2 Todennäköisimmän tilasekvenssin avulla tehty LVQ 68

6.7.3 Todennäköisimmän tilasekvenssin avulla tehty itseorganisointi 68

6.8 Mallin parantaminen 69

6.8.1 Tilojen transitiotodennäköisyyksien painottaminen 69

6.8.2 Foneemimallien joustavuuden lisääminen 71

6.8.3 Kiellettyjen trigrammien estäminen 72

6.8.4 Kaksoisfoneemien tunnistaminen 73

7 Yhteenveto **75**

Lähdeluettelo **77**

Luku 1

Johdanto

1.1 Puheentunnistuksen nykytila

Automaattisen puheentunnistuksen kehitys on edennyt maailmalla jo siihen vaiheeseen, että puheentunnistusta hyödyntäviä sovelluksia esiintyy monilla aloilla. Tärkeä sovellusalue on puhelimet, joissa puheentunnistimien käyttö vaihtelee äänivalintaisista puhelimista pörssikursseja soittajan valinnan mukaan kertovaan automaattiseen puhelinvastaajaan. Myös erilaisia yksinkertaisia käskyjä ymmärtäviä automaatteja on sovellettu useisiin eri tarkoituksiin.

Puheentunnistuksen tutkimus on tietokoneiden laskentatehon ja nopeasti käytettävän muistikapasiteetin kasvun takia jatkuvasti lisääntynyt. Suosittuja tutkimusalueita ovat sovellusten lisäksi sopeutuminen uuteen puhujaan, hermoverkkomallit, järjestelmien opettaminen ja optimointi, kielten mallittaminen, tunnistuksen perusyksiköt ja tunnistettavien piirteiden valinta ja käsittely.

1.2 Merkittävimpiä puheentunnistusjärjestelmiä

Puheentunnistustutkimus yliopistoissa ja suurissa tutkimuslaitoksissa on tuottanut lukuisia hyviä puheentunnistusjärjestelmiä, jotka suoriutuvat melko hankalistakin tunnistustehtävistä, kuten jatkuvasta puheesta ja kymmenien tuhansien sanojen sanastosta. Näiden järjestelmien kaupallista toteuttamista hankaloittavat kuitenkin niissä käytettyjen algoritmien vaatima suuri laskentateho ja muistikapasiteetti. Mainitsemisen arvoisia järjestelmiä ovat muunmuassa Dragon Dictate [Baker 1989], Sphinx [Lee 1988] ja Tangora [Averbuch 1987].

1.3 Puheentunnistustutkimus TKK:n informaatiotekniikan laboratoriossa

TKK:n informaatiotekniikan laboratoriossa puheentunnistus on ollut tärkeä tutkimuskohde hahmontunnistuksen ja hermoverkkojen käytännön sovelluksena. Laboratoriossa on keskitytty erityisesti suomenkielisen puheen tunnistukseen. Tunnistuksen lähtökohtana on foneemien tunnistaminen ja siten puheen foneettisen kuvauksen tuottaminen. Lisäksi on tutkittu menetelmiä, joilla foneettinen kuvaus muunnetaan kirjoitettua tekstiä vastaavaan muotoon.

1.4 Kätketyt Markov-mallit puheentunnistuksessa

Lähes kaikki jatkuvaa puhetta ja laajaa sanavarastoa tunnistavat puheentunnistusjärjestelmät (mm. aiemmin mainitut Dragon, Sphinx ja Tangora) käyttävät nykyään kätkettyjä Markov-malleja (hidden Markov models, HMM). Mallien suosio perustuu siihen, että niiden matemaattinen perusta tunnetaan hyvin ja mallien parametrit voidaan opettaa yksinkertaisin ja tehokkain menetelmin tallennetusta puhedatasta. HMM:illä voidaan luontevasti yhdistää puheen hetkellisen spektrin mallittaminen stationaarisena stokastistisena prosessina ja näiden prosessien temporaaliset suhteet Markovin ketjulla.

HMM:ien soveltamisessa puheentunnistukseen on ilmennyt sekä teoreettisia että käytännöllisiä ongelmia. Markovin ketjujen perusoletuksen mukaan peräkkäiset tilanvaihdot ovat riippumattomia. Puhesignaalissa esiintyy kuitenkin myös pidempiaikaisia riippuvuuksia. Laskettaessa puhesignaalin hetkellisiä spektrejä käytetään yleensä osittain päällekkäisiä aikaikkunoita. Tilojen todennäköisyyksien laskukaavat eivät täten ole tarkkoja, koska niissä peräkkäiset havainnot oletetaan riippumattomiksi.

Käytännössä edellisiä vakavampi ongelma on kuitenkin parametriestimaattien alkuarvojen määritys. Kaavat antavat parametreille todennäköisyyden paikallista maksimia läheneviä arvoja, mutta todellinen maksimitodennäköisyys saavutetaan vain alkuarvoja lähimmän maksimin ollessa globaali. Foneemien tilojen todennäköisyysjakaumat ovat osoittautuneet hyvin hankaliksi, joten hyvien alkuarvojen valinta on ratkaisevan tärkeää. Erityisesti tämän ongelman ratkaiseminen on ollut tämän työn lähtökohtana kehitettäessä foneemimallien opetusmenetelmiä.

1.5 Vektorikvantisointi foneemien mallittamisessa

Puheessa esiintyvien foneemien akustiset piirteet muodostavat varsin monimutkaisen jakauman. Syntyviin äänteisiin vaikuttavat monet ympäröiviin muihin äänteisiin, puhujaan ja puhetilanteeseen liittyvät tekijät. Tunnistuksen kannalta oleellinen informaatio siivilöityy tehokkaasti vektorikvantisoimalla signaalista laskettujen piirteiden muodostamat vektorit. Kvantisoinnin jälkeen voidaan havainnot helposti luokitella käyttäen hyväksi aiemmin kerätyn aineiston tilastollisia ominaisuuksia. Monissa puheentunnistusjärjestelmissä onkin päädytty vektorikvantisointiin foneemien mallittamisen apuna.

Eri luokkien vektorien erottelun sijasta vektorikvantisoinnilla voidaan ryhmitellä vektorit myös luokkien sisällä. Tässä työssä vektorikvantisointia on sovelletaan mahdollisimman hyvän ja erottelukykyisen jakauma-approksimaation tuottamiseksi foneemin tilojen tiheysfunktioille. Vektorikvantisoinnin menetelmiä käytetään sijoiteltaessa lineaarikombinaation osajakaumat oikeille paikoilleen vastaamaan sopivasti todellisen jakauman tihentymiä.

1.6 Tämän työn sisältö

Tämän työn toisen luvun alussa kuvataan puhumalla tapahtuvan ajatusten vaihdon eri vaiheita. Hieman tarkemmin tarkastellaan foneemia, kielen perusyksikköä. Puheentunnistusongelman erilaisista lähestymistavoista johtuvaa erityyppisten tunnistinten suurta joukkoa jäsennellään tutkimalla muutamia järjestelmien luokittelutapoja. Automaattisesta puheentunnistuksesta esitellään lyhyesti historiaa ja puheentunnistimien tärkeimpiä sovelluksia.

Kolmannessa luvussa kuvataan työn lähtökohtana oleva puheentunnistusjärjestelmä. Järjestelmän toiminnan ymmärtäminen on erityisen tärkeää arvioitaessa saavutettujen tulosten tasoa, sillä monivaiheisen prosessin tuloksia vertailtaessa on aina otettava huomioon kaikkien käytettyjen vaiheiden toteutukset. Vertailtavien menetelmien paremmuus on siis järjestelmäkohtainen ja yhtä järjestelmää käyttämällä tehtyjä johtopäätöksiä ei voida sellaisenaan siirtää muihin järjestelmiin.

Kätkettyjen Markov-mallien periaatteet ja mallien käytössä tarvittavat menetelmät selostetaan luvussa neljä. Havaintoihin parhaiten sopiva malli haetaan Viterbi-algoritmin avulla ja mallien opettaminen esimerkkinäytteitä käyttäen ratkaistaan iteratiivisella Baum-Welch –algoritmilla.

Luvussa viisi pohditaan yleisesti ilmiöitä kuvaavien piirteiden valintaa ja piirrevektorien käsittelyssä sovellettavaa vektorikvantisointia. Hankalien piirrevektorijakaumien approksimointimenetelmänä käsitellään itseorganisoituvia piirrekarttoja. Piirrevektoreihin perustuvan luokittelun optimointialgoritmiksi sopivia oppivia vektorikvantisointimenetelmiä esitellään myös lyhyesti.

Työn tärkein osa eli tässä työssä kehitetyt foneemimallit ja opetusmenetelmät esitellään luvussa kuusi. Luvuissa neljä ja viisi selostetuista menetelmistä kuvataan käytetyt sovellukset ja tehtyjen tunnistuskokeiden tulokset.

Viimeisessä luvussa on yhteenveto saaduista koetuloksista ja aikaisempien lukujen pohjalta tuloksista vedetyt johtopäätökset.

Luku 2

Puheentunnistus

2.1 Puhe

2.1.1 Puhuminen

Puhe on puhujan tuottamaa vaihtelevaa paineaaltoa, joka kulkee puhujan suusta vastaanottajan korvaan. Puhesignaali syntyy paineen vaihtelusta ajan funktiona.

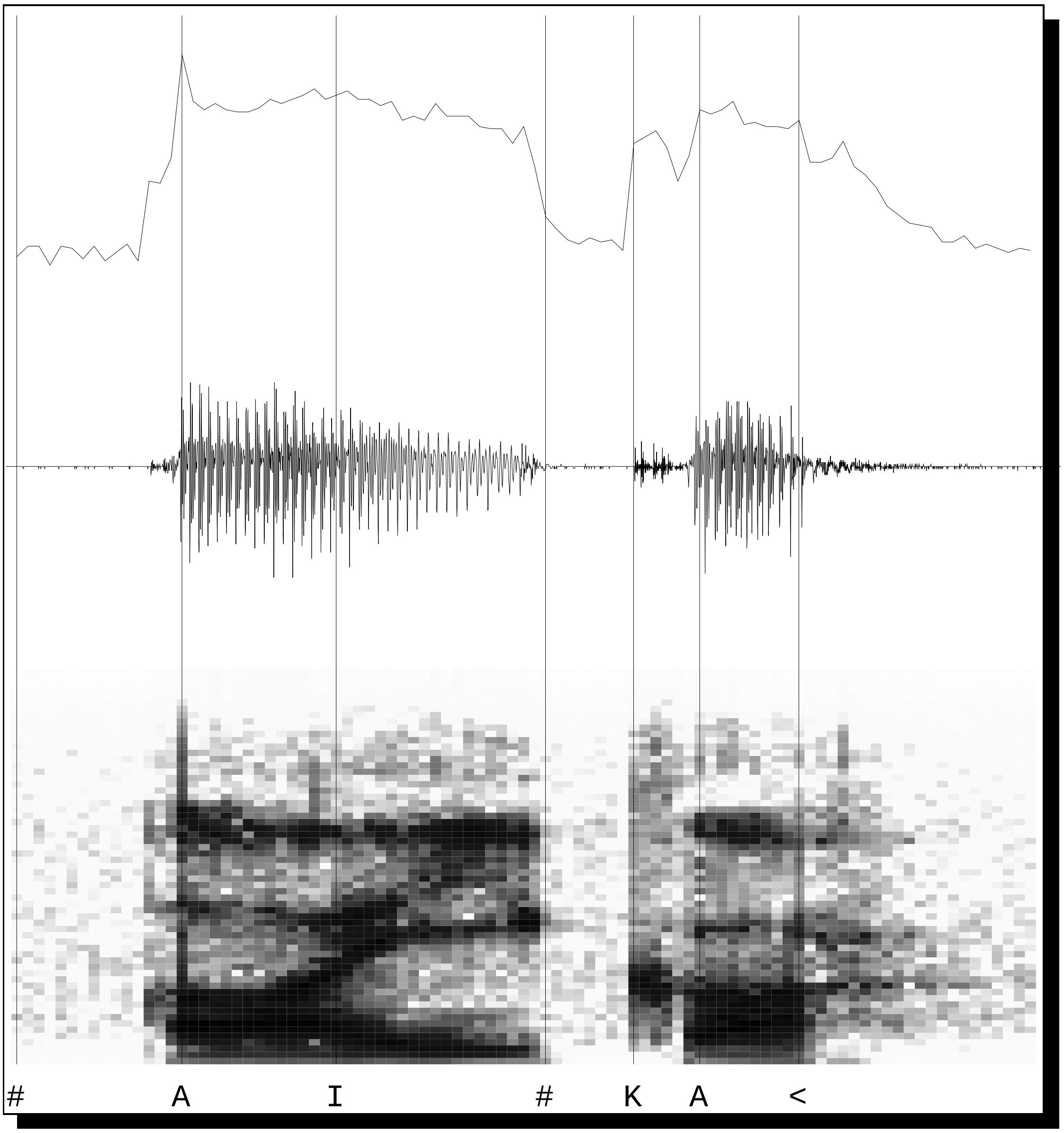
Ilmavirta syntyy puhujan keuhkoissa ja haluttu äänne tuotetaan kurkunpään äänihuulten ja ääniväylän avulla. Äänihuulten jaksollinen avaaminen ja sulkeminen tuottaa soinnillisia äänteitä. Äänihuulten ollessa kiinni tai osittain kiinni syntyvät soinnittomat äänteet.

Ääniväylä on akustinen suodin, jolla voidaan säädellä ääniaallon eri taajuuksien energiajakaumaa. Ääniväylä koostuu nielusta, suuontelosta, nenäontelosta ja huulista. Aikuisella miehellä ääniväylän pituus on noin 17cm ja sen muotoa muuttelemalla voidaan toisia ääniaaltojen taajuuksia voimakkaasti vahvistaa ja toisia vaimentaa. Suurin osa muodonmuutoksista tehdään puheessa kielen, huulten ja hampaiden avulla. Äänen syntymekanismia on käsitelty tarkemmin mm. viitteessä [O'Shaughnessy 1987].

2.1.2 Puhesignaali

Puhesignaalin aaltomuoto on esitetty keskimmäisenä kuvassa (2.1). Alempana kuvassa on signaalin hetkellinen tehospektri ajan funktiona laskettuna 10 ms välein. Vaaka-akselilla on aika ja pystyakselilla taajuudet 50 – 6400 Hz 100 Hz välein. Kuvapisteiden tummuus vastaa spektrikomponentin voimakkuutta. Ylhäällä olevaa spektrin hetkellisen kokonaistehon eli ns. RMS-arvon käyrää käytetään mm. sanojen erottamiseen puheesta.

Tehospektrin huippukohdat vastaavat ääniväylän resonanssikohtia eli formantteja. Puheen ymmärtämisessä muutaman ensimmäisen formantin sijainti ja suhteellinen amplitudi ovat



Kuva 2.1: *Esimerkki puhesignaalista. Keskimmäinen kuvaaja näyttää signaalin aaltomuodon, alimmainen signaalista 10 ms välein lasketun spektrin ja ylimmäinen spektrin tehon eli ns. RMS-arvon. Kuvan tietojen perusteella on foneemien alkukohdat merkitty pystyviivoilla.*

erityisen tärkeitä. Puhesignaalille on luonteenomaista, että signaalin piirteet säilyvät lähes samanlaisina muutaman kymmenen millisekunnin ajan, mutta 50 – 100 ms välein tapahtuu merkittäviä muutoksia. Puheen spektrin suuret muutokset johtuvat ääniväylän muodonmuutoksista eri äänteiden välillä.

2.1.3 Kuuleminen

Ihmisen puheentuottomekanismi ja vastaavasti kuuloaisti eli puhesignaalin dekoodauskyky ovat todennäköisesti kehittyneet rinnakkain parhaan mahdollisen yhteistoiminnan aikaansaamiseksi. Korva poimii signaalista erityisesti 200 – 5600 Hz taajuiset muutokset, jotka sisältävät viestinnän kannalta tärkeimmän informaation.

Korvan uloimmaisen osan tehtävänä on ohjata paineaallot rumpukalvolle, josta välikorva muuntaa painevaihtelut mekaaniseksi liikkeeksi. Sisäkorva lähettää mekaanisia värähtelyitä vastaavat sähköiset hermoimpulssit kuulohermoja pitkin aivoihin. Korvan toimintaa ja psykoakustisia ominaisuuksia on tutkittu tarkkaan ja siitä on olemassa paljon hyviä kuvauksia mm. viitteessä [O'Shaughnessy 1987]. Toistaiseksi ei kuitenkaan tiedetä, miten hermoimpulsseihin koodattua puheen aika- ja taajuusinformaatiota käytetään kielellisen ilmaisun tuottamiseen.

2.1.4 Puheen ymmärtäminen

Korvan lähettämien hermoimpulssien muuntaminen foneemeiksi ja sanoiksi sekä viestitettyjen ajatusten tulkitseminen on monimutkainen prosessi, jota on vaikea jakaa selkeästi peräkkäisiin vaiheisiin. Puhesignaalista ei voida myöskään tarkkaan erotella, miten eri foneemit on siihen koodattu. Suodattamalla signaalista erilaisia taajuuskaistoja ja lisäämällä erilaisia häiriöitä on voitu kokeellisesti todeta [O'Shaughnessy 1987], että puhe on hyvin redundanttia eli se sisältää monia erilaisia piirteitä, joiden perusteella kukin foneemi on tunnistettavissa.

Puhetta kuunteleva ihminen käyttää akustisen informaation lisäksi myös lukuisia muita keinoja viestin selvittämiseksi. Tuttu puhuja ja aihe auttavat puheen tulkintaa samoin kuin erilaiset ilmeet ja eleet. Jopa täysin kuuro henkilö voi harjaantua tunnistamaan puhetta huulien liikkeitä seuraamalla.

2.2 Foneemit

Foneemi on puhutun kielen rakenteen perusyksikkö ja pienin yksikkö, joka erottaa ilmauksien merkityksen. Mikäli jonkin sanan joku foneemi vaihdetaan, voi koko lauseen merkitys muuttua. Foneemien puheessa esiintyviä reaalisaatioita sanotaan äänteiksi. Kirjoitetussa tekstissä foneemit erotetaan / –merkeillä, esimerkiksi /a/.

2.2.1 Foneemien luokittelu

Foneemit luokitellaan yleensä niiden ääniväylällä olevan muodostumiskohdan perusteella. Kaksi pääluokkaa ovat vokaalifoneemit ja konsonanttifoneemit. Vokaaleissa ääniväylä on melko avoin ja konsonanteissa lähes suljettu jostakin kohdasta. Alkuperäisessä suomen kielessä on 8 vokaalia: /a/, /e/, /i/, /o/, /u/, /y/, /ä/, /ö/ ja 13 konsonanttia: /d/, /ŋ/, /h/, /j/, /k/, /l/, /m/, /n/, /p/, /r/, /s/, /t/ ja /v/. Lisäksi muista kielistä otettujen lainasanojen mukana on tullut uusia konsonanttifoneemeja kuten /b/, /f/, /g/ ja /š/.

	Etuvokaalit		Takavokaalit	
	Illabiaalit	Labiaalit	Illabiaalit	Labiaalit
Suppeat	/i/	/y/		/u/
Puolisuppeat	/e/	/ö/		/o/
Väljät	/ä/		/a/	

Taulukko 2.1: *Suomen kielen vokaalifoneemit*

Vokaalit luokitellaan ääntämispaikan mukaan etu- ja takavokaaleihin ja ääniväylän supistumisasteen perusteella suppeisiin, puolisuppeisiin ja väljiin. Huulten asennon mukaan vokaalit jaetaan vielä labiaaleihin ja illabiaaleihin. Suomenkielen vokaalien luokittelu on taulukossa (2.1) [Laaksonen et al. 1988].

	Labiaalit	Dentaalit	Palato-velaarit	Laryngaalit
Klusiilit				
soinnittomat	/p/	/t/	/k/	
soinnilliset	/b/	/d/	/g/	
Frikatiivit	/f/	/s/, /š/		/h/
Lateraalit		/l/		
Tremulantit		/r/		
Nasaalit	/m/	/n/	/ŋ/	
Puolivokaalit	/v/		/j/	

Taulukko 2.2: *Suomen kielen konsonanttifoneemit*

Konsonantit jaetaan ääniväylän sulkeutumiskohdan mukaan labiaaleihin (huulet), dentaaleihin (hampaat ja kielen kärki), palatovelaareihin (kitalaki ja kielen keski- tai takaosa) ja laryngaaleihin (kurkunpää). Konsonantit luokitellaan myös sen perusteella, miten ääniväylän läpi kulkevaa ilmavirtaa rajoitetaan. Näitä luokkia ovat klusiilit, frikatiivit, lateraalit, tremulantit, nasaalit ja puolivokaalit. Esimerkiksi klusiileissa ääniväylä on suurimman osan ajasta täysin suljettuna ja vasta lopuksi ilmavirta purkautuu nopeasti ulos. Suomenkielen konsonanttien luokittelu on taulukossa (2.2) [Laaksonen et al. 1988].

Foneemeista suurin osa on soinnillisia. Soinnittomia ovat vain konsonanteista frikatiivit ja osa klusiileista.

2.2.2 Foneemit eri kielissä

Kaikilla maapallon tutkituilla kielillä on jonkinlainen foneemijärjestelmä. Foneemeja on yleensä noin 20 – 40 kappaletta ja ne jaetaan vokaaleihin ja konsonantteihin. Foneemipohjainen puhe on ilmeisestikin jollakin tavalla optimaalinen suullisen informaation välityskeino, koska kehitys on johtanut samantyyppisiin ratkaisuihin eri puolilla maapalloa, vaikka osa kansoista on elänyt viime aikoihin asti melko eristyneissäkin oloissa.

2.2.3 Foneemien erottaminen toisistaan

On vaikea sanoa tarkasti, minkä perusteella ihmiskuulo lopulta suorittaa foneemien erottelun. Puhe on foneemien erottamisen suhteen varsin redundanttia eli tunnistamiseen vaadittava informaatio esiintyy siinä monilla eri tavoilla. Tärkein erottava tekijä lienee kuitenkin ääniväylän resonanssitaajuuksien ja niiden siirtymien havaitseminen. Ympäröivien foneemien muodostaman kontekstin vaikutus tunnistuspäätökseen on myös suuri; johtaahan konteksti koartikulaatioilmiöiden kautta samankin foneemin erilaisiin ilmenemismuotoihin. Puhenopeus ja -voimakkuus, foneemien kesto ja tietyt välivaiheet äänteen muodostuksessa auttavat ilmeisesti myös foneemien erottelussa samoin kuin visuaalinen informaatio kuten huulten liikkeet.

2.2.4 Foneemien kirjoittaminen

Suomen kielessä jokaista foneemia vastaa oma merkkinsä ja merkit äännetään aina samalla tavalla muutamia poikkeuksia lukuunottamatta. Tavallisimmat poikkeukset ovat /ŋ/ ja /p/:n edellä /m/:ksi ääntyvä /n/. Suomen kielessä pitkä äänne merkitään aina kahdella ja lyhyt yhdellä kirjaimella.

Useimmissa muissa kielissä ääntämisen ja kirjoituksen välillä on suuriakin eroja (esimerkiksi ranskassa ja englannissa) ja foneemit kirjoitetaan erilaisilla merkkiyhdistelmillä. Osa kirjaimista saattaa tietyissä yhteyksissä myös jäädä kokonaan ääntymättä.

2.3 Puheentunnistimien luokittelu

2.3.1 Puheen kerääminen

Puheentunnistimet voidaan luokitella erilaisten puheentallennustapojen perusteella erillisten sanojen, korostetun puheen ja jatkuvan puheen tunnistimiin. Erillisiä sanoja kerättäessä kukin sana talletetaan erikseen ja sanan alku ja loppu on tarkkaan määritettävissä. Korostetussa puheessa sanat äännetään erityisen huolellisesti ja niiden väliset erot pyritään osoittamaan selkeällä painotuksella.

Puheentunnistimet eroavat toisistaan myös ulkoisten häiriöiden sietokyvyssä. Sovelluksen kannalta on merkittävä ero siinä, tarvitaanko puheen tallennukseen hiljainen laboratorioympäristö vai voidaanko se tehdä esimerkiksi ulkona tai tehdashallissa. Samoin on miellyttävää, jos tunnistinta voidaan käyttää luonnollisella puheella, eikä tarvitse puhua hitaasti ja ääntää erityisen huolellisesti.

2.3.2 Kieli

Eri kielien automaattisessa puheentunnistuksessa suurimmat erot ovat sanojen muodoissa ja foneemeissa. Esimerkiksi englannin kielessä sanat esiintyvät pieniä poikkeuksia lukuunottamatta aina samassa muodossa, joten monissa sovelluksissa rajoitetun sanaston tunnistimet ovat täysin riittäviä, kun valitaan riittävän suuri sanasto. Hyvä vastaesimerkki on suomen kieli, jossa useimmilla sanoilla on lukuisia keskenään hyvinkin erilaisia muotoja (esim. käsi, kättä, kädellä jne.).

Useimmissa kielissä puhe ja kirjoitettu kieli eroavat toisistaan huomattavasti: kaikkia kirjaimia ei äännetä ja monien ääntäminen vaihtelee eri sanoissa. Suomessa puhe ja kirjoitus vastaavat harvinaisen tarkasti toisiaan (eroja on muutamia esimerkiksi sana “kengät”).

Sanojen erottaminen toisistaan on joissakin kielissä vaikeaa, koska ne pyritään lausuttaessa sitomaan toisiinsa (esim. ranska). Myös voimakas äänenpainon vaihtelu sanan sisällä hankaloittaa tunnistusta joidenkin foneemien muuttuessa pelkäksi muminaksi tai kadotessa kokonaan.

2.3.3 Sanavarasto

Hallittavan sanaston perusteella puheentunnistimet ovat jaettavissa pienen, suuren ja rajoittamattoman sanavaraston tunnistimiin.

Pienen sanavaraston tunnistimet erottavat yleensä erikseen lausutuista komentosanoista muutamia kymmeniä tai satoja sanoja vertaamalla niitä muistissa olevien sanojen piirteisiin. Piirteet voivat olla hetkellisiä spektrikomponentteja tai niistä edelleen laskettuja muutoksia ja toistoja kuvaavia kertoimia. Sanaston pienuuden vuoksi jokaisesta sanasta voidaan tallettaa useita esimerkkityyppejä ja tunnistuksessa päästään lähes 100%:n tarkkuuteen. Tunnistimia voidaan toteuttaa myös puhujariippumattomasti. Käyttökohteena on erilaiset selvästi rajoitetut komentosanastot ja esimerkiksi lausuttujen numeroiden tunnistus.

Suurella sanavarastolla tarkoitetaan yleensä kymmeniä tuhansia sanoja, joilla pyritään kattamaan jokapäiväisessä puheessa esiintyvät sanat. Sanaston koon takia vaaditaan tehokkaita vertailualgoritmeja ja vähäparametrisia malleja sanojen käsittelyssä. Tunnistusta hankaloittaa suuresti puhekielessä esiintyvien sanojen eri muodot, jotka lisäävät vaadittavan muistitilan moninkertaiseksi. Ongelmia aiheuttavat myös toisiaan muistuttavat sanat, jotka puheessa äännetään lähes samalla tavalla ja erotetaan toisistaan lähinnä rakenteiden ja lauseyhteyden avulla.

Rajoittamattoman sanavaraston tunnistimissa lähtökohtana on sanaa pienempien puheen yksiköiden, kuten tavujen ja foneemien, tunnistus. Talletettavien mallien lukumäärää voidaan näin vähentää, mutta ongelmaksi muodostuu toisaalta puhutun sanan jakaminen osiin ja toisaalta oikean sanan kokoaminen tunnistetuista osista. Tunnistuksen hierarkkisuuden vuoksi tunnistuksessa syntyviä virheitä on vaikea automaattisesti korjata; ei voida yksiselitteisesti erotella todellisia ja mahdottomia sanoja.

2.3.4 Tunnistettavat osat

Puheentunnistimen totetuttamisen kannalta tärkeä kysymys on sopivan tunnistettavan perusyksikön valinta puhesignaalista. Perusyksiköiden tunnistuksen onnistuminen vaikuttaa ratkaisevasti koko puheentunnistuksen lopputulokseen, sillä tulostettavaa tekstiä muokataan jälkikäsittelyssä enää vain tunnistettujen perusyksiköiden ja tunnistettavan aineiston yleisien ominaisuuksien perusteella. Perusyksikön valinta on siis tehtävä huolella ottaen huomioon tunnistettava aineisto ja tunnistimen käyttötarkoitus. Mahdollisia vaihtoehtoja ovat esimerkiksi sanat, tavut, foneemiparit ja foneemit.

Sana on paras valinta perusyksiköksi, jos tunnistustehtävä rajoittuu tiettyyn ennaltatunnettuun sanajoukkoon, joka ei ole liian laaja. Normaalin suomenkielisen puheen tunnistaminen laatimalla kullekin sanalle oma malli (tai useita eri malleja) ei esimerkiksi ole mielekästä.

Tavujen erottaminen puheesta on huomattavasti vaikeampaa kuin sanojen, sillä tavut sulautuvat usein toisiinsa varsin kiinteästi. Erilaisten tavujen lukumäärä suomenkielessä on selvästi sanojen määrää pienempi, mutta niitäkin lienee silti yli kymmenen tuhatta.

Foneemien suurin etu tunnistettavina yksikköinä on ilman muuta niiden vähäinen määrä. Suomenkielisessä puheessa foneemeja on 21, kun vieraskielisiä lainasanoja ei oteta huomioon. Useimmissa ihmisten käyttämissä kielissä foneemeja on 20 – 40, esimerkiksi englannin kielessä 40 [O'Shaughnessy 1987]. Jatkuvan puhesignaalin osittaminen diskreetiksi foneemijonoksi ei kuitenkaan ole helppoa. Foneemien kestoaika vaihtelee suuresti ja sama foneemi saattaa kuulostaa eri ympäristössä hyvinkin erilaiselta koartikulaation vuoksi. Koartikulaatioilmiössä ääniväylän asento ympäröivissä äänteissä vaikuttaa syntyvään äänteeseen ja peräkkäisten äänteiden väliset erot pyrkivät tasoittumaan.

Koartikulaatioilmiöiden huomioimiseksi on puheentunnistusjärjestelmissä käytetty foneemien sijasta foneemipareja eli difoneja tai ympäristösidonnaisia foneemeja eli trifoneja [Lee 1988]. Difonien haitat ovat samanlaisia kuin tavujen: mallitettavia yksiköitä on paljon ja niiden erottelu puheesta on vaikeaa. Trifonimalleissa kullakin foneemilla on eri mallit eri naapurifoneemeja varten, joten koartikulaatio on hyvin mallitettu. Esimerkiksi englannin kielessä viereisten foneemien vaikutus toisiinsa on ääntämisessä merkittävä. Trifonien käyttö johtaa kuitenkin varsin suureen mallimäärään, joista monet ovat turhia, koska osa naapurifoneemeista aiheuttaa samanlaisen koartikulaation.

2.3.5 Puhujan valinta

Pyrittäessä yhä parempaan tunnistustulokseen on puheentunnistusjärjestelmä säädettävä puhujan ominaisuuksien mukaan, koska eri henkilöiden puhetavassa on suuria eroja. Kun tunnistusta varten lasketaan puhesignaalista hetkellisiä ominaisuuksia, jotkut saman puhujan eri foneemeista otetut näytteet saattavat jopa muistuttaa enemmän toisiaan kuin eri puhujien samasta foneemista otetut näytteet. Hyvän tunnistustuloksen saamiseksi on siis puhesignaalin matemaattinen malli laskettava kullekin puhujalle erikseen. Tarkassa mallien säädössä on kuitenkin aina vaarana, että mallista tulee liian herkkä, jolloin pienetkin muutokset puheessa sotkevat tunnistuksen.

Puhujasta riippuvien ja riippumattomien tunnistusjärjestelmien välimuotoina ovat toisaalta tietyn ryhmän jäsenten puhetta tunnistavat systeemit ja toisaalta systeemit, jotka kykenevät nopeasti omaksumaan puhujanvaihdoksen, eikä opetusta tarvitse aloittaa uudelleen alusta lähtien.

2.4 Esimerkki puheentunnistusjärjestelmästä – SPHINX

SPHINX on kehitetty Carnegie Mellonin yliopistossa USA:ssa. Järjestelmä tunnistaa puhujariippumattomasti jatkuvaa puhetta soveltaen laajaa sanavarastoa. Tässä kappaleessa kuvataan SPIHINX:in perusversio hyvin lyhyesti. Tarkempi selvitys on esimerkiksi viitteessä [Lee 1988].

Puheen hetkellisiksi piirteiksi valitaan 10 millisekunnin välein lasketut LPC-kepstrikertoimet. Kukin piirrevektori vektorikvantisoidaan lähimpään koodivektoriin 256 vektorin kvantisointikoodikirjan mukaan. Kvantisointikoodikirja on laadittu hierarkkisella LBG-algoritmilla [Linde et al. 1980]. Tunnistuksen perusyksikkönä käytetään kolmen äänteen muodostamaa trifonia. Samalla tavalla vaikuttavien foneemiympäristöjen yhdistämisellä vähennetään opetettavia malleja. Kätkettyjä Markov-malleja soveltaen yhdistetään äänteet sanoiksi ja sanat lauseeksi.

Puhujariippumattomuus toteutetaan jakamalla puhujat puheominaisuuksien perusteella ryhmiin, joille opetetaan omat mallit. Tunnistuksessa on siis ensimmäiseksi valittava mihin ryhmään puhuja kuuluu.

2.5 Automaattinen puheentunnistus

Automaattisessa puheentunnistuksessa on tavoitteena usein jäljitellä maailman parhaan puheentunnistimen – ihmisen kuuloaistin – toimintaa. Osittain tässä on onnistuttukin; puhesignaalista kyetään nykyisin menetelmin laskemaan riittävän tiheästi ja riittävällä tarkkuudella tehospektrit ym. signaalin sisältämä tunnistusinformaatio. Lopullisesti ratkaisematta on kuitenkin vielä, miten tätä informaatiota käytetään optimaalisesti ja mitä osaa siitä tarvitaan kunkin äänteen tunnistamiseen.

Hyviin tunnistustuloksiin pyrittäessä joudutaan, varsinkin käsiteltäessä laajaa sanavarastoa, käyttämään paljon muistitilaa puheyksikköjen malleihin ja laskentakapasiteettia tunnistukseen. Tämä on vielä viime vuosiin asti heikentänyt reaaliaikaisen tunnistuksen tarkkuutta.

Ihmisen kuunnellessa puhetta toimii kuuloaisti jatkuvasti saumattomassa yhteistyössä muiden aistien ja kokemusperäisen päättelyn kanssa. Ympäristöä, puhujaa ja puheenaihetta seuraamalla aivot pystyvät jatkuvasti luomaan eräänlaisen ennusteen tunnistettaville sanoille. Esimerkiksi televisiota katsottaessa pystyy ihminen välittömästi huomaamaan, jos puhe ei vastaa kuvan tilannetta. Tämän kaltaista assosiointia ei automaattisen puheentunnistuksen tueksi ole vielä kyetty luomaan.

Verrattaessa automaattisen puheentunnistuksen ja -tuottamisen vaikeutta on esitetty, että ihmisen puheentunnistuskykyä vastaavan järjestelmän kehittäminen olisi yhtä vaikeaa kuin syntetisaattorin, joka täysin rajoituksitta tuottaisi tekstistä puhetta, jota ei voida erottaa ihmisen puheesta [O'Shaughnessy 1987].

2.6 Puheentunnistuksen historiaa

Ensimmäiset toimivat puheentunnistimet kehitettiin vasta 1950-luvulla Yhdysvalloissa. Niillä saatiin lupaavia tuloksia yksittäin lausuttujen numeroiden ja pienestä sanavarastosta valittujen sanojen tunnistuksessa. Sanaston laajennus ei kuitenkaan onnistunut. Tuloksien parantamiseksi rakennettiin myöhemmin tekoälyohjelmia, joilla pyrittiin hallitsemaan kielioppimalleja.

1970-luvulla Yhdysvalloissa aloitettiin laaja projekti puheentunnistustutkimuksen koordinoimiseksi ja saavutettujen tulosten vertailemiseksi. Projektista vastasi suuri amerikkalainen tutkimusorganisaatio DARPA (*Defense Advanced Research Projects Agency*). Samaan aikaan Japanissa oli oma kansallinen hahmontunnistusprojekti PIPS. Yliopistojen ja valtiollisten tutkimusyksikköjen lisäksi myös suurten yksityisten yritysten (mm. IBM, AT&T Bell, Hitachi, Fujitsu ja NEC) tutkimuslaitoksissa on tutkittu puheentunnistusta.

1980-luvulla puheentunnistustutkimus sai uutta vauhtia tehokkaiden tietokoneiden, kätkettyjen Markov-mallien ja keinotekoisten hermoverkkoalgoritmien kehittymisen takia. Hermoverkkojen adaptiivisuus, assosiointiominaisuudet, kyky luoda optimaalisia luokittelusääntöjä esimerkkien avulla ja soveltuvuus rinnakkaislaskentaan ovat osoittautuneet hyödyllisiksi monilla aivojen toimintaa jäljittelevillä informaatiotekniikan aloilla.

2.7 Puheentunnistimen sovelluksia

Automaattisten puheentunnistimien käytännön sovelluksia on toistaiseksi rajoittanut järjestelmien korkea hinta ja riittämätön suorituskyky. Nykyiset kaupalliset järjestelmät vaativat yleensä tehokkaan laitteiston, mutta ne eivät kuitenkaan suoriudu kuin tiettyjen tarkkaan määriteltyjen sanojen erottamisesta. Kaikki tavallisimmat sanat eri muodoissaan ja useiden henkilöiden puhumina erottavat tunnistimet eivät toistaiseksi kykene toimimimaan reaaliajassa tekemättä liikaa virheitä. Esimerkiksi sanelun perusteella toimiva kirjoituskone, joka tekee useita kirjoitusvirheitä joka rivillä, ei vielä syrjäytä tavallista kirjoituskonetta. Mikäli työskentely kuitenkin tapahtuu tilanteessa, jossa kirjoittaminen olosuhteiden vuoksi on vaikeaa, virheitäkin tekevästä puheentunnistimesta on apua. Jos jatkuvaa puhetta tunnistava kone saadaan tuotettua kohtuuhinnalla sellaiseksi, että ihminen kykenee saaduista tuloksista päättelemään viestin sisällön tunnistusvirheistä huolimatta, siitä on huomattavaa hyötyä esimerkiksi kuulovammaisille.

Kommunikaatio tietokoneen kanssa laajenee merkittävästi automaattisen puheentunnistimen avulla. Muutoksia ei todennäköisesti tapahdu ohjelmakoodin tallentamisessa, mutta tiedon syöttäminen voi helpottua huomattavasti. Esimerkiksi pitkien tavara- tai nimiluetteloiden tallentaminen sujuisi kevyesti, sillä onhan puhuminen ihmiselle luontevampi tiedonvälitystapa kuin kirjoittaminen. Puhuminen sujuu ongelmitta myös silloin kuin kädet on sidottu muuhun työhön tai valaistus on puutteellinen. Nopean ja vähän virheitä tekevän puheentunnistimen avulla voidaan kehittää keskustelevia tietokoneohjelmia, joissa konetta ohjataan puheella ja vastaukset saadaan näytölle tai tuotetaan puhesyntetisaattorilla. Tällaisille sovelluksille on varmasti odotettavissa lukuisia käyttökohteita, mikäli puheentunnistimen hinta saadaan pudotetuksi sopivalle tasolle.

Puheen perusteella toimiva automaattinen kielenkääntäjä on varsin houkutteleva sovellus, mutta se vaatii erittäin tehokkaan puheentunnistimen lisäksi myös nopeaa kielten rakenteiden hahmottamista. Saattaa vierähtää vielä monia vuosia ennenkuin tavarataloissa myydään taskulaskinta muistuttavia puheenkääntäjiä, jotka tunnistavat reaaliaikaisesti satoja eri kieliä ja kääntävät tuloksen halutulle kielelle [Vonnegut 1989].

Luku 3

Otaniemen puheentunnistusjärjestelmä

3.1 Järjestelmän kehitys

Puheentunnistustutkimusta on tehty Teknillisen korkeakoulun informaatiotekniikan laboratoriossa vuodesta 1975 lähtien.

Alkuvaiheessa käytettiin foneemien tunnistukseen oppivaa aliavaruusmenetelmää, mutta nykyään foneettinen tunnistus perustuu oppivaan vektorikvantisointiin. Keskeinen ongelma tunnistuksessa on ollut myös tunnistettavien piirteiden valitseminen ja irroittaminen puhesignaalista. Tehokkaiden signaaliprosessoreiden avulla voidaan signaalista laskea nopeasti Fourier-muunnos käyttäen FFT-algoritmia. FFT-spektristä saatu logaritmoitu hetkellinen tehospektri onkin vakiintunut puheen piirteiden perustaksi. Spektrikertoimien sijasta käytetään nykyään tehospektristä edelleen muuntamalla saatuja kepstrikertoimia.

Tietokoneiden laskentatehon nopea kehitys on vaikuttanut merkittävästi tunnistusmenetelmien valintaan; kun laboratorioon on saatu uusia huomattavasti voimakkaampia koneita, on myös voitu siirtyä käyttämään uusia raskaampia menetelmiä ja selvittämään vaikeampia ongelmia, kuten rajoittamattoman sanaston ja jatkuvan puheen tunnistusta. Ohjelmiston kannalta ratkaiseva muutos oli muutama vuosi sitten tapahtunut siirtyminen henkilökohtaisten työasemien käyttöön.

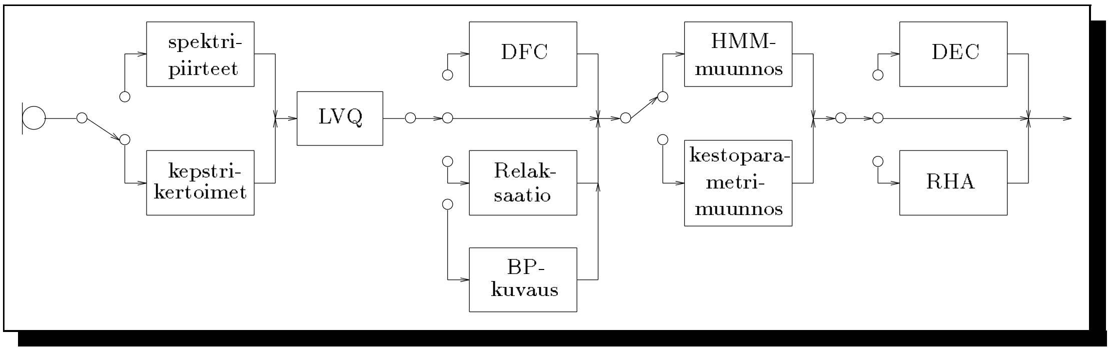
Tuorein kuvaus puheentunnistusjärjestelmästä [Torkkola et al. 1991a] on julkaistu Otaniemessä kesäkuun lopussa pidetyssä kansainvälisessä keinotekoisten hermoverkkojen konferensissa (ICANN-91), jossa järjestelmän toimintaa myös julkisesti demonstroitiin.

3.2 Laitteisto

Puheaineisto kerätään ja esikäsitellään mikrotietokoneeseen asennetulla signaaliprosessorikortilla, joka suoriutuu myös piirteiden laskennasta. Ethernet-verkon välityksellä tiedot siirtyvät Silicon Graphicsin Iris 4D –työasemalle. Työasema laskee vektorikvantisoinnin ja muodostaa hetkellisistä tunnistustuloksista puheen foneettisen kuvauksen. Jälkikäsittelyllä foneettinen kuvaus muunnetaan tekstiksi. Työaseman grafiikkaominaisuuksien avulla voidaan tarkkailla ja ohjata tunnistuksen vaiheita ja näyttää tunnistettu teksti selkeästi.

Tunnistusjärjestelmän tietokoneohjelmat on toteutettu C-kielisenä ohjelmistopakettina, jossa kutakin prosessointivaihetta vastaa oma ohjelmansa. Järjestelmän tavoitteena on joustavien yleiskäyttöisten ohjelmien kokonaisuus, jossa ohjelmia ja työvaiheita voidaan helposti siirrellä laitteesta toiseen tarpeen vaatiessa ja uusien algoritmien ominaisuuksien testaaminen vanhojen ohjelmien kanssa on helppoa.

3.3 Toimintaperiaate



Kuva 3.1: *Otaniemen puheentunnistusjärjestelmän lohkokaavio*

3.3.1 Yleistä

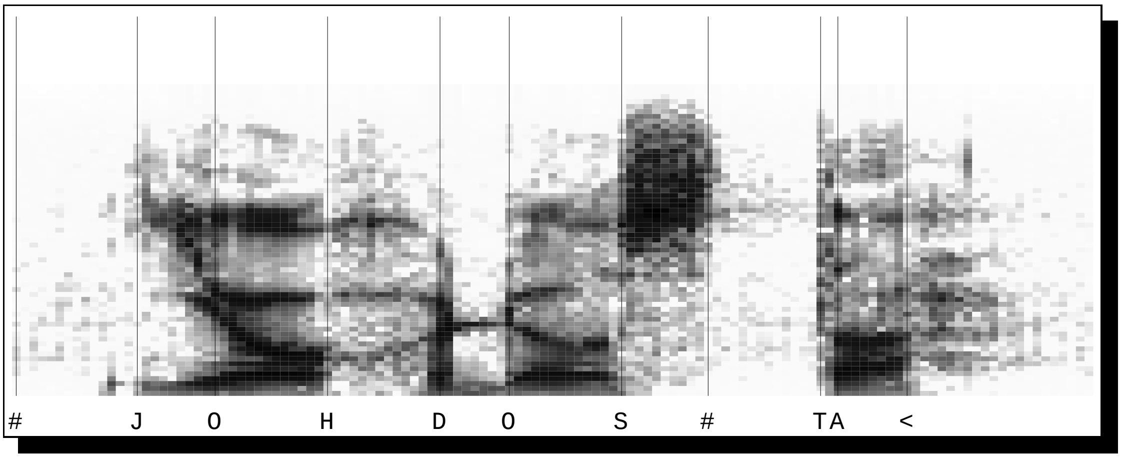
Puheentunnistusjärjestelmä toimii reaaliaikaisesti tunnistaen puhutusta suomen kielestä foneemit ja tuottaen niistä tekstiä. Järjestelmä on puhujariippuva, mutta se on helposti opetettavissa myös uusille puhujille.

3.3.2 Puhesignaalin esikäsittely

Puhe digitoidaan 16 bittisellä A/D -muuntimella käyttäen 25.6 kHz:n näytteenottotaajuutta ja desimoidaan 12.8 kHz taajuudelle, joka on riittävän tarkka puheentunnistukseen. Korostamalla korkeita taajuuksia kompensoidaan niillä esiintyvää spektrin intensiteetin heikkenemistä. Korkeat taajuudet ovat tärkeitä erityisesti konsonanttien tunnistuksessa. 128 näytteen (10 ms) välein lasketaan 256 peräkkäisen puhesignaalinäytteen (20 ms) muodostama Hamming-ikkuna ($N = 256$).

$$w_M(n) = 0.54 - 0.46 \cos \frac{2\pi n}{N-1}\,, \quad n = 0, 1, \cdots, N-1 \tag{3.1}$$

Hamming-ikkunointi korostaa ikkunan keskellä sijaitsevia näytteitä ja estää voimakkaiden spektripiikkien vuotamisen lähitaajuuksille.



Kuva 3.2: *Puhesignaalista on 10 ms välein laskettu tehospektrit. Vaaka-akselina on aika, pystyakselina taajuus (50 – 6400 Hz) ja spektrikomponenttien voimakkuutta esittää kuvapisteiden tummuus. Foneemien alkukohdat on merkitty pystyviivoilla eri vaiheiden hahmottamisen helpottamiseksi. # –merkki tarkoittaa hiljaisuutta sanan alussa tai klusiilin edellä ja < sanan loppua.*

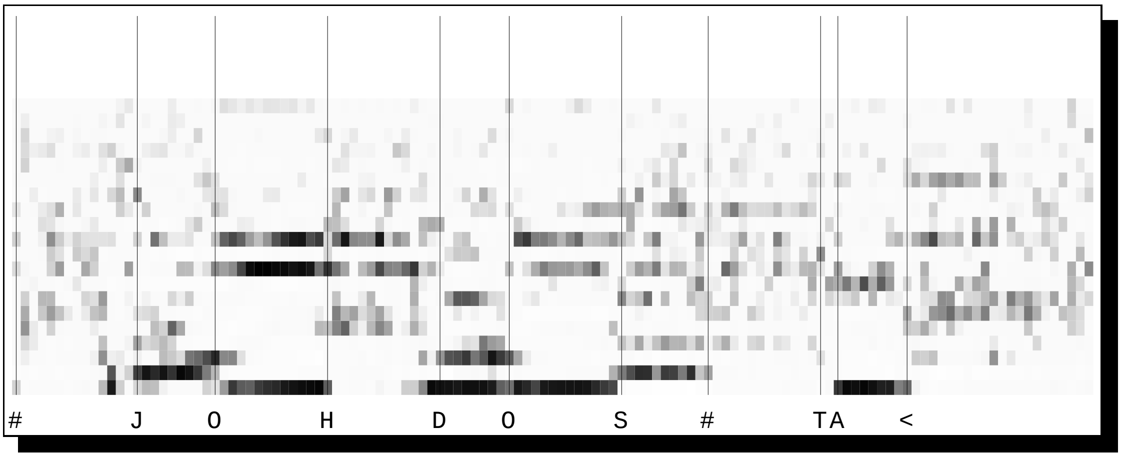
Ikkunoidusta signaalista lasketaan 256 pisteen FFT, josta saadaan 128-komponenttinen logaritmoitu tehospektri. Kuvassa 3.2 tehospektrit ovat pystysuuntaisia raitoja, joissa pisteet ovat sitä tummempia, mitä suurempi on komponentin arvo.

3.3.3 Piirteiden laskeminen

Spektrivektorista suodatetaan 22 spektrikomponenttia X_k käyttäen osittain päällekkäisiä kolmionmuotoisia kaistasuodattimia. Suodattimet noudattavat kuulon fysiologisiin ja psykoakustisiin ominaisuuksiin perustuvaa Mel-asteikkoa, joka jakaa puheen taajuusalueen alapään (50 – 1000 Hz) 10 tasamittaiseen ja yläpään (1 – 5 kHz) 12 eksponentiaalisesti pitenevään osaan.

Piirrevektorina käytetään 20-komponenttista kepstriä, joka saadaan tekemällä suodatetulle tehospektrille diskreetti kosinimuunnos [Davis et al. 1980].

$$MFCC_i = \sum_{k=1}^{N} X_k \cos\left[i\left(k - \frac{1}{2}\right)\frac{\pi}{N}\right], i = 1, 2, \ldots, M, \tag{3.2}$$



Kuva 3.3: *Puhesignaalista on 10 ms välein laskettu kepstrit. Hetkellisten kepstrikomponenttien voimakkuutta esittää kuvapisteiden tummuus. Foneemien alkukohdat merkitty pystyviivoilla eri vaiheiden hahmottamisen helpottamiseksi. /</ on sanan loppu.*

N on suodattimien ja M kepstrikertoimien lukumäärä. Painottamalla kepstrikertoimia sopivasti [Kaski 1990] esimerkiksi kohotettua sinikäyrää vastaavilla kertoimilla, saadaan poistettua ylimääräistä kohinaa [Tohkura 1987]. Hetkellisiä kepstrejä on kuvassa 3.3.

Piirrevektoriin voidaan myös liittää spektrin hetkellinen kokonaisteho eli ns. RMS-arvo, joka on kuvattu ajan funktiona ylimpänä kuvassa 2.1.

3.3.4 Kvasifoneemien määrittäminen

Piirrevektorit tunnistetaan vertaamalla niitä muistissa oleviin mallivektoreihin ja valitsemalla kullekin piirrevektorille lähimpänä olevan mallivektorin foneemitunniste. Näitä sata kertaa sekunnissa saatavia tunnistettuja piirrevektoreita sanotaan kvasifoneemeiksi ja yhtä sanaa vastaavia tunnistuksia kvasifoneemijonoksi.

Mallivektorien muodostamiseen käytetään itseorganisoituvaa karttamenetelmää (SOM) ja oppivaa vektorikvantisointia (LVQ) [Kohonen 1982] ja [Kohonen 1986b]. Karttamenetelmä kuvaa moniulotteiset piirrevektorit kaksiulotteiselle tasolle siten, että etäisyyssuhteet säilyvät ja kuvaustarkkuus on paras siellä missä vektoritiheys on suurin. LVQ:n avulla mallivektorit sijoitellaan niin, että kvasifoneemien luokittelutarkkuus on mahdollisimman hyvä.

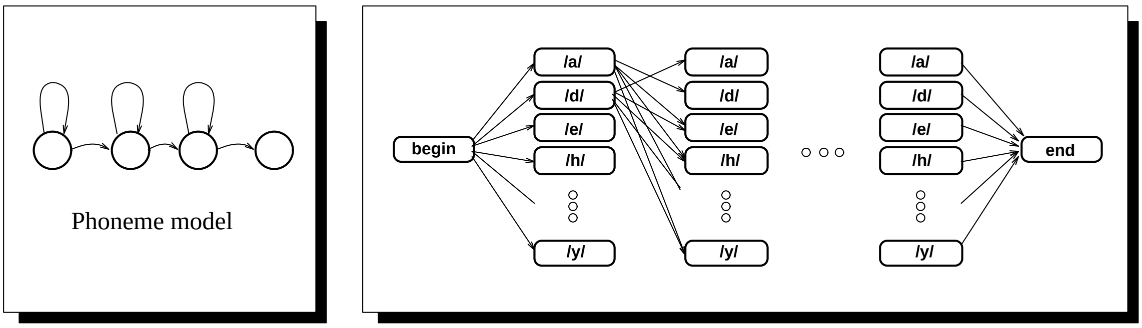
Aluksi mallivektorit valitaan satunnaisesti. Karttaa opetetaan opetusaineistosta valituilla piirrevektoreilla valitsemalla kutakin opetusvektoria lähinnä oleva mallivektori ja muuttamalla sitä ja sen naapuriyksikköjä enemmän opetusvektorin kaltaiseksi. Kartan yksiköiden väliset etäisyydet määritellään kokonaislukuina pitkin verkkoa, joka muodostuu yhdistettäessä jokainen yksikkö kuuteen naapuriinsa. Opetuksen edetessä muutoksia pienennetään ja opetettavien naapurien määrää vähennetään. Opetusaineistona on saman puhujan kolmeen eri kertaan puhumat 311 sanaa.

Mallivektoreita vastaavien tunnisteiden löytämiseksi käydään opetusaineisto uudelleen läpi ja valitaan kutakin opetusvektoria lähimpänä oleva mallivektori. Mallivektorin tunnisteeksi valitaan se foneemitunnus, jota vastaavista opetusvektoreista on useammalle valittu lähimmäksi kyseinen mallivektori.

Vektoriluokittelukyvyn parantamiseksi karttaa opetetaan vielä oppivalla vektorikvantisoinnilla. Opetusaineisto käydään vielä kerran läpi, ja kutakin opetusvektoria lähimpänä olevaa mallivektoria muutetaan enemmän opetusvektorin kaltaiseksi, jos tämän mallivektorin foneemitunniste on sama kuin opetusvektorin. Jos foneemitunnukset eivät ole samoja muutetaan mallivektoria vastakkaiseen suuntaan.

Opetuksen pituus ja muutosten pienentäminen opetuksen edistyessä ovat samoja kuin itseorganisoivassa opetuksessa. Soinnittomien klusiilien /k/, /p/ ja /t/ erottamista varten opetetaan erillinen kartta [Kangas 1988].

3.3.5 Foneemien tunnistaminen kvasifoneemijonosta



Kuva 3.4: *Foneemimallin rakenne ja yhdistetty malli HMM-dekoodauksessa.*

Kvasifoneemit tulkitaan diskreetin havaintojakauman kätkettyjen Markov-mallien ulostulosymboleiksi [Torkkola et al. 1991a]. Jokaisen foneemin mallina on erillinen nelitilainen ketjumainen Markov-malli, jossa sallitut siirtymät tapahtuvat ketjussa edeltävästä tilasta seuraavaan tilaan (kuva 3.4). Kullakin tilalla on oma ulostulojakauma, joka koostuu siis eri kvasifoneemien havaintotodennäköisyyksistä.

Tilojen diskreetit ulostulojakaumat ja transitiotodennäköisyydet opetetaan mallille iteratiivisesti Baum-Welch –estimoinnilla [Rabiner 1989]. Menetelmän tavoitteena on saada laskettua parametriarvot, jotka tuottavat käytettyä opetusdataa vastaavat kvasifoneemijonot suurimmalla uskottavuudella (Maximum Likelihood –estimointi). Iteratiivisessa algoritmissa määrätään vuorotellen saatujen parametrien avulla eri tilojen hetkelliset todennäköisyydet ja todennäköisyyksien avulla uudet parametrit.

Foneemimalleja voidaan yhdistellä peräkkäin kokonaisen sanan mallittamiseksi. Sanan mallissa alkuun ja loppuun lisätään hiljaisuuden mallit ja foneemimallit liitetään toisiinsa siten, että edellisen mallin lopputilasta voidaan siirtyä seuraavan alkutilaan.

Foneemien väliset siirtotodennäköisyydet estimoidaan laskemalla peräkkäisten foneemien frekvenssit laajasta tekstiaineistosta (kymmeniä tuhansia sanoja). Todennäköisin foneemiyhdistelmä etsitään Viterbi-haulla [Rabiner 1989]. Hakumenetelmä perustuu dynaamisen optimointiin; näytteen alkuhetkestä lähtien määrätään kullekin mahdolliselle tilalle todennäköisin edeltävä tilasekvenssi ja kun on päästy näytteen loppuun valitaan sallittuun lopputilaan päätyvistä tilasekvensseistä todennäköisin. Pitkien foneemien erottamiseksi lasketaan haun yhteydessä myös viipymisajat kussakin foneemimallissa.

Kätkettyjen Markov-mallien sijasta segmentoinnissa voidaan käyttää kestoparametreja, jotka määräävät foneemien pituudet [Kohonen et al. 1984]. Foneemi c tunnistetaan, mikäli n_c:n mittaisessa kvasifoneemijonossa on vähintään m_c kappaletta tunnusta c. Jos peräkkäin tunnistetaan kaksi kertaa sama foneemi ja tunnistuspäätöksen aiheuttaneet kvasifoneemisekvenssit ovat riittävän kaukana toisistaan, tulkitaan havainnot kaksoisfoneemiksi. Jokaista foneemia kohden tarvitaan kolme parametria, joista kolmas kuvaa siis foneemin maksimipituutta. Uusimmat testitulokset [Torkkola et al. 1991a] osoittavat kuitenkin, että kätketyt Markov-mallit antavat parempia tuloksia.

3.3.6 Tulosten parantaminen paikallisen kontekstin avulla

Koartikulaatioilmiöitä ja muita systemaattisia virheitä voidaan korjata DEC-algoritmilla (Dynamically expanding context) [Kohonen 1986a] tuotetuilla yhteysherkillä säännöillä. Yhteysherkkä säännöstö sisältää esimerkkien avulla generoituja yksittäisten symbolien muunnossääntöjä, joiden avulla dekoodatuista foneemiyhdistelmistä saadaan oikeita sanoja.

Samanlaista menetelmää voidaan soveltaa myös kvasifoneemijonojen tarkasteluun. Koartikulaatioilmiöiden ja hetkellisten epäpuhtaiden äännähdysten aiheuttamien poikkeavien symbolien muuntaminen helpottaa foneemien dekoodausta ja tuottaa tarkemman tunnistustuloksen. Menetelmää sanotaan dynaamisesti tarkentuvaksi kontekstiksi (Dynamically focusing context) [Torkkola 1990].

Kvasifoneemien paikallinen konteksti on mahdolista ottaa huomioon opettamalla neuraaliverkko kuvaamaan kvasifoneemi ympäristöineen ideaaliseksi kvasifoneemiksi. Tähän on sovellettu mm. back-propagation –algoritmia [Torkkola et al. 1990]. Ideaalisten kvasifoneemien määrittäminen relaksaatioalgoritmilla, joka perustuu toisiaan lähellä olevien tunnistustulosten välisten riippuvuuksien malleihin [Kurimo 1990], ja edellä mainittujen menetelmien vertailua on selvitetty viitteessä [Torkkola et al. 1991b].

3.4 Suorituskyky

Tässä työssä on tutkittu erillisten suomenkielisten sanojen foneettisen kuvauksen tuottamista noudattaen rajoittamattoman sanaston periaatteita. Suomen kielen foneemit on jaettu 21 luokkaan, joista soinnittomat klusiilit /k/, /p/ ja /t/ on useimmissa testeissä kuitenkin luettu samaan luokkaan niiden erottamisen hankaluuden vuoksi. Lainasanoista peräisin olevat äänteet /b/, /f/ ja /g/ on jätetty luokittelematta.

Puhuja	oikein%	virhe%
kt	94.8	8.1
jk	91.3	14.9
pu	91.6	11.0

Taulukko 3.1: *Tunnistuskeskiarvot eri puhujilla tunnistettaessa 311 sanan testiaineistoa, jossa soinnittomat klusiilit on luettu samaan luokkaan.*

Testiaineiston tunnistamiseksi on järjestelmä opetettu saman puhujan kolmella riippumattomalla opetusaineistolla. Neljän eri testin keskiarvot kullekin puhujalle on esitetty taulukossa 3.1. Tulokset on tuotettu kokoonpanolla (vrt. kuva 3.1), johon kuuluu kepstripiirteet, LVQ ja segmentointi yhden koodikirjan HMM:ien avulla. Parempan segmentointitulokseen olisi päästy useammalla koodikirjalla, kuten erillisellä kpt-tunnistimella ja rms-arvojen luokittelulla, mutta vertailujen helpottamiseksi tunnistus haluttiin toteuttaa mahdollisimman yksinkertaisesti. Foneettista kuvausta on verrattu suoraan puhuttuihin sanoihin suorittamatta mitään virheitä korjailevaa jälkikäsittelyä (esim. DEC).

Luku 4

Kätketyt Markov-mallit

4.1 Mallin kuvaus

Kätketyt Markov-mallit (engl. hidden Markov models, HMM) on todennäköisyyksiin perustuva tapa tutkia aikasarjoja. Aikasarjassa havainnot tehdään ajan suhteen diskreetisti, mutta itse havainnot voivat olla peräisin myös jatkuvasta jakaumasta. Kätketty Markov-malli koostuu kahdesta osasta: äärellisen tilajoukon Markovin ketjusta ja äärellisestä määrästä ulostulojakaumia. Markovin ketju tuottaa jonon tiloja eli tilojen välisen polun, ja tiloihin liittyvät ulostulojakaumat muuttavat tilajonon aikasarjaksi. Havaittu aikasarja antaa siis tietoa, jonka perusteella eri tilajonoille voidaan laskea todennäköisyydet. Mallia sanotaan "kätketyksi", koska tilojen välistä polkua ei voida suoraan havainnoida [Poritz 1988].

Kätketty Markov-malli on täydellisesti määrätty, jos tunnetaan [Rabiner 1989]:

1. Systeemin mahdolliset tilat (N kpl) ja niiden väliset siirtotodennäköisyydet.

$$a_{ij} = P(q_{t+1} = j | q_t = i) \quad i, j = 1, \cdots, N \tag{4.1}$$

   Systeemin tilaa aikana $t = 1, \cdots, T$ merkitään jonolla $Q = q_1, q_2, \cdots, q_T$.

2. Systeemistä saatavien havaintojen jakaumat kussakin tilassa.

$$b_j(O) = P(O_t = x | q_t = j) \quad j = 1, \cdots, N \tag{4.2}$$

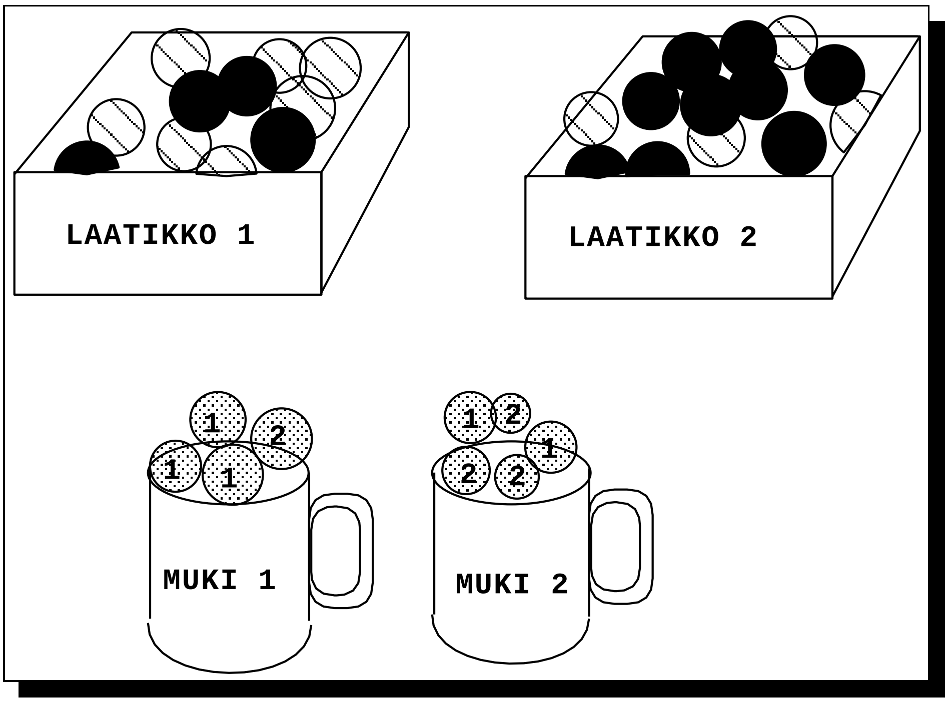
   Systeemistä saatuja havaintoja merkitään jonolla $O = O_1, O_2, \cdots, O_T$. Mikäli havainnot ovat diskreettejä (M kpl), tunnetaan kunkin arvon todennäköisyys erikseen kussakin tilassa eli $x = 1, \cdots, M$. Jatkuvasta havaintojakaumasta tunnetaan tiheysfunktion parametrit jokaisessa tilassa.

Jos kätketty Markov-malli tunnetaan, saadaan eri tilajonojen todennäköisyydet Bayesin kaavalla käyttäen havaintojen todennäköisyyttä systeemin eri tiloissa (kaavat 4.4 ja 4.5).

$$P(Q|O) = \frac{P(O|Q)P(Q)}{P(O)} \tag{4.3}$$

Jos malli on tuntematon, se voidaan iteratiivisesti estimoida lähtien liikkeelle parametrien alkuarvoista. Parametrien avulla lasketaan eri tilojen todennäköisyydet kullakin hetkellä ja yhdistämällä ne systeemin tilasta saatuihin havaintoihin estimoidaan mallin parametrit uudelleen. Parannetuilla parametreilla saadaan taas totuudenmukaisemmat tilatodennäköisyydet, joista vuorostaan paremmat parametrit. Iteratiivinen estimointi tuottaa kohti suurimman uskottavuuden mallia [Liporace 1982] suppenevan parametrijonon. Suurimman uskottavuuden malli on parametriyhdistelmä, joka suurimmalla todennäköisyydellä tuottaa saadut havainnot.

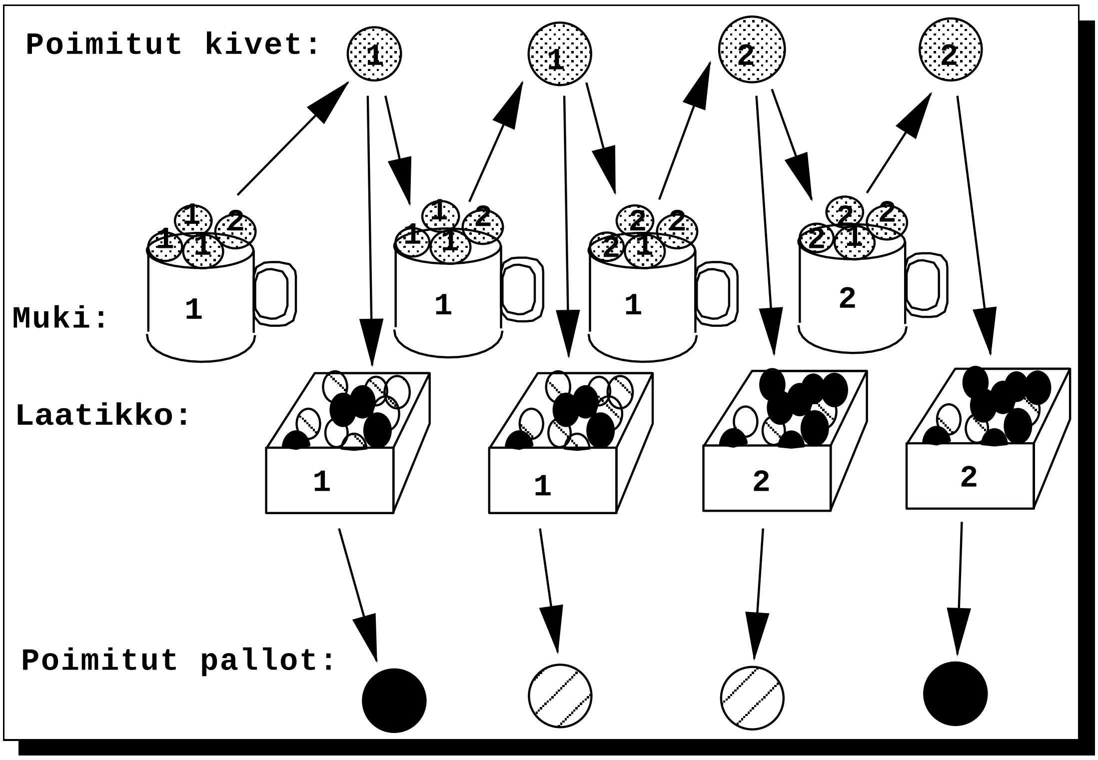
4.2 Esimerkki kätketystä Markov-mallista



Kuva 4.1: *Laatikoissa on mustia ja valkoisia palloja. Mukeissa on 1-kiviä ja 2-kiviä.*

Oletetaan että kahdessa laatikossa on mustia ja valkoisia palloja. Mustien ja valkoisten pallojen lukumäärien suhde tunnetaan kummassakin laatikossa erikseen. Oletetaan lisäksi, että kahteen mukiin laitetaan kiviä, joista osa on merkitty numerolla 1 ja osa numerolla 2 (kuva 4.1). Erinumeroisten kivien suhde kummassakin mukissa on tunnettu [Poritz 1988].

Poimitaan kivi mukista 1, ja jos kivessä on numero 1, poimitaan pallo laatikosta 1 ja seuraava kivi mukista 1. Jos taas kivessä on numero 2, pallo poimitaan laatikosta 2 ja seuraava kivi mukista 2. Pallon väri kirjataan muistiin.



Kuva 4.2: *Poimitun kiven numero määrää systeemin tilan eli laatikon ja mukin numeron.*

Nostetun kiven numero osoittaa systeemin tilan, eli kumpi pallolaatikko on käytössä. Pallon väri puolestaan on havainto systeemin tilasta, sillä eriväristen pallojen suhde on erilainen eri laatikoissa.

Toistetaan koetta poimien kiviä ja palloja aina edellisen kiven numeron mukaisesta mukista ja laatikosta (kuva 4.2). Kivet ja pallot palautetaan noston jälkeen aina alkuperäiseen astiaansa, jotta saatavien numeroiden ja värien todennäköisyydet säilyvät muuttumattomina. Pallojen värit kirjataan muistiin poimintajärjestyksessä, mutta kivien numerot unohdetaan.

Kun palloja on nostettu yhteensä T kappaletta, on generoitu peräkkäisten havaintojen jono $O = O_1, O_2, \cdots, O_T$, jossa havainto O_t on pallon t väri. Poimittujen kivien numeroita, jotka siis kertovat, mistä laatikosta kukin pallo poimittiin, ei merkitty muistiin, joten ne muodostavat tuntemattoman tilajonon $Q = q_1, q_2, \cdots, q_T$.

Systeemin malli sisältää parametrit $b_1(M)$, $b_1(V)$, $b_2(M)$, $b_2(V)$ eli mustien ja valkoisten pallojen osuudet laatikossa 1 ja 2, sekä parametrit a_{11}, a_{12}, a_{21}, a_{22} eli 1- ja 2-numeroisten kivien osuudet mukeissa 1 ja 2. Itseasiassa $b_i(O)$ kertoo todennäköisyyden, jolla tilassa i tehdään havainto O ja a_{ij} siirtymistodennäköisyyden tilasta i tilaan j.

4.3 Havaintojonon todennäköisyys

Todennäköisyys, että ajassa T saadaan havainnot $O = O_1, O_2, \cdots, O_T$, kun systeemin tila samaan aikaan noudattaa jonoa $Q = q_1, q_2, \cdots, q_T$ on

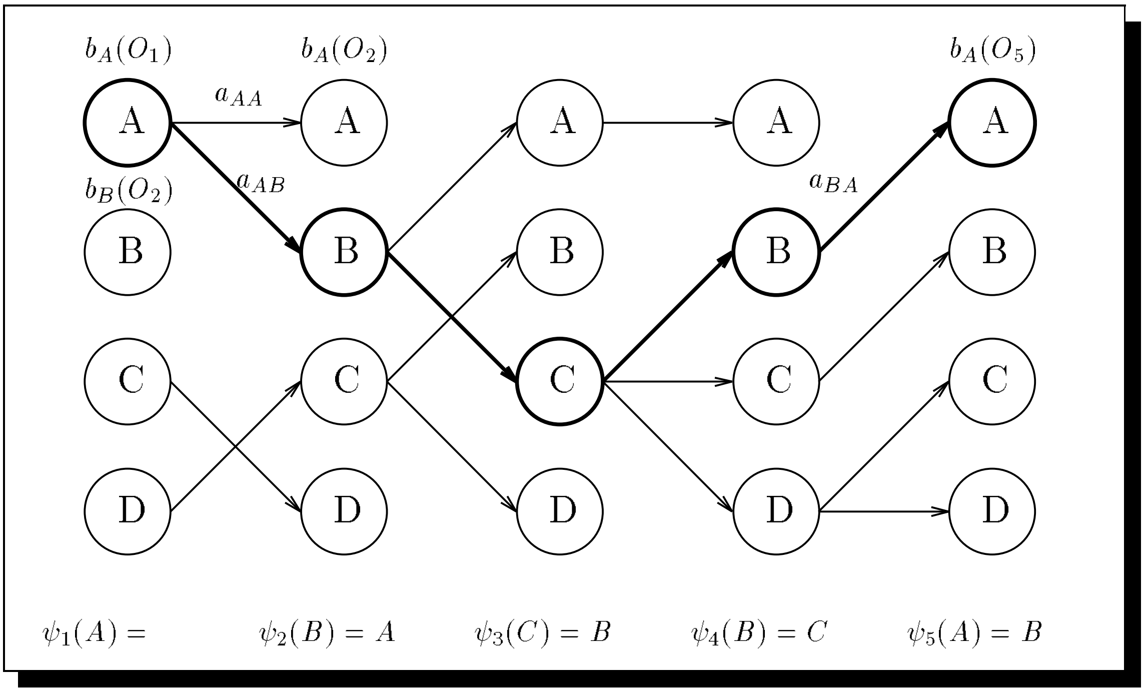
$$P(O|Q) = \prod_{t=1}^{T} b_{q_t}(O_t) \tag{4.4}$$

Havainnot oletetaan riippumattomiksi toisistaan.

Koska systeemin todellisesta tilasta ei ole muuta tietoa kuin em. havainnot ja tiloja säätelevä Markov-malli, havaintojonon todennäköisyys tuntemattomalla tilajonolla on summa kaikkien mahdollisten tilajonojen todennäköisyyksistä tilansiirtotodennäköisyyksillä a_{ij} painotettuna.

$$P(O) = \sum_{all\ Q} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(O_t) \tag{4.5}$$

4.4 Todennäköisin tilajono



Kuva 4.3: *Viterbi-haku. Systeemillä on neljä mahdollista tilaa A, B, C ja D. Kuvassa viisi perättäistä ajanhetkeä, jolloin systeemin tila on määrättävä. Kuhunkin hetkelliseen tilavalintaan on vedetty nuoli todennäköisimmästä edeltäjästä. Tummennettu reitti on todennäköisin polku.*

Todennäköisin polku systeemin eri tilojen välillä saadaan tehokkaasti haettua dynaamiseen optimointiin perustuvalla Viterbi-haulla [Rabiner 1989] (kuva 4.3).

Viterbi-haun idea on valita järjestyksessä jokaisella aika-askeleella kullekin mahdolliselle tilalle todennäköisin edeltävä tilajono. Todennäköisimmän tilaan j hetkellä t päättyvän tilajonon todennäköisyys määritellään rekursiivisesti

$$\delta_t(j) = b_j(O_t) \max_i \delta_{t-1}(i) a_{ij} , \quad t = 1, \cdots, T \quad i, j = 1, \cdots, N \tag{4.6}$$

$\delta_0(j) = 1.0$, jos tila j on sallittu alkutila ja 0 muuten. Todennäköisimmäksi tilajonoksi valitaan siis todennäköisin kombinaatio edelliseen ajanhetkeen mennessä todennäköisimmästä jonosta ja siirtymisestä seuraavaan tilaan. Todennäköisimmät tilajonot talletetaan taulukkoon ψ.

$$\psi_t(j) = \arg\max_i \delta_{t-1}(i) a_{ij} , \quad t = 1, \cdots, T \quad i, j = 1, \cdots, N \tag{4.7}$$

Taulukoidaan siis kullakin aika-askeleella jokaisen tilan todennäköisin edeltäjä, kuten taulukossa (4.1) on tehty. Kun kuhunkin lopputilaan j päättyvät todennäköisimmät jonot $\delta_T(j)$ on laskettu, valitaan niistä suurin $\delta_T(j^*)$, jossa j^* on sallittu lopputila q_T^*. Todennäköisin tilajono $Q^* = q_1^*, q_2^*, \cdots, q_T^*$ (merkitty paksulla viivalla kuvaan (4.3)) saadaan purkamalla taulukosta ψ

$$q_t^* = \psi_{t+1}(q_{t+1}^*) , \quad t = T-1, T-2, \cdots, 1 \tag{4.8}$$

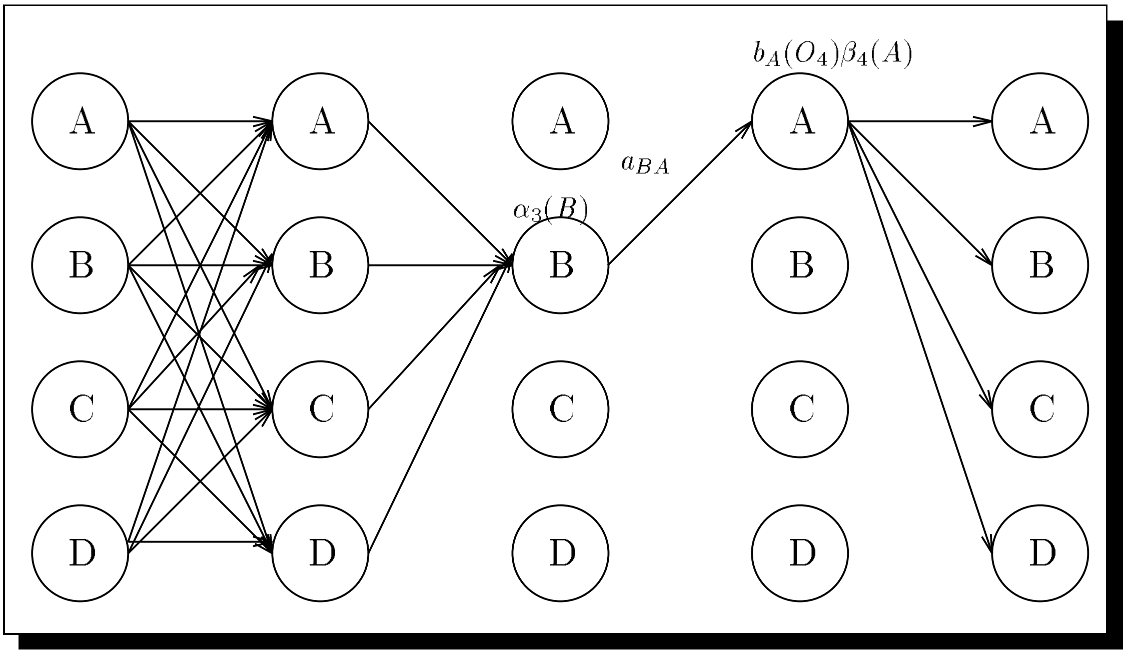
	ψ_1	ψ_2	ψ_3	ψ_4	ψ_5
A		A	B	A	B
B		A	C	C	C
C		D	B	C	D
D		C	C	C	D
	q_1^*	q_2^*	q_3^*	q_4^*	q_5^*
Q^*	A	B	C	B	A

Taulukko 4.1: *Todennäköisimmät tilajonot. Taulukosta voidaan eri ajanhetkillä lukea kunkin tilan todennäköisin edeltäjä. Viimeisellä rivillä on todennäköisintä lopputilaa vastaava tilajono.*

4.5 Parametrien estimointi

Systeemin tilojen väliset siirtotodennäköisyydet a_{ij} voidaan opettaa mallille esimerkkien avulla. Opetuksen tavoitteena on saavuttaa siirtotodennäköisyydet, jotka maksimoisivat opetusjonojen todennäköisyydet [Liporace 1982]. Iteratiivinen Baum-Welch –algoritmi [Rabiner 1989] sopii tähän tarkoitukseen hyvin.

Ensin lasketaan opetusjonojen havaintotodennäköisyydet $P(O)$. Kaava 4.5 on kuitenkin laskennallisesti liian raskas, koska se vaatii opetusjonon todennäköisyyden laskemisen kaikilla mahdollisilla tilajonoilla (N^T kpl). Baum-Welch -algoritmi ratkaisee ongelman dynaamisen ohjelmoinnin periaatteella (kuva 4.4) vaiheittain (kuten Viterbi-algoritmi edellä).



Kuva 4.4: *Baum-Welch –algoritmi kontekstin huomioivien tilatodennäköisyyksien laskemiseksi. Kuvassa on laskettu todennäköisyys, että ajanhetkellä 3 siirrytään tilasta B tilaan A.*

Baum-Welch –algoritmin todennäköisyyksien laskukaava (4.9) eroa Viterbi-algoritmin vastaavasta kaavasta (4.6) siten, että seuraavan tilan todennäköisyyteen vaikuttaa todennäköisimmän jonon sijasta kaikkien edeltävien jonojen summa. $\alpha_t(j)$ määritellään rekursiivisesti

$$\alpha_t(j) = b_j(O_t)\sum_{i=1}^{N}\alpha_{t-1}(i)a_{ij}\,, \quad t = 1,\cdots,T \quad j = 1,\cdots,N \tag{4.9}$$

$\alpha_0(j) = 1.0$, jos tila j on sallittu alkutila ja 0 muuten. $\alpha_t(j)$ on siis todennäköisyys, että on tehty havainnot $O_1, O_2, \cdots, O_t$ ja päädytty tilaan j hetkellä t. Koko opetusjonon havaintotodennäköisyys on siis eri lopputiloihin päätyvien jonojen todennäköisyyksien summa.

$$P(O) = \sum_{i=1}^{N}\alpha_T(i) \tag{4.10}$$

Vastaavasti määritellään $\beta_t(i)$ rekursiivisesti

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}b_j(O_{t+1})\beta_{t+1}(j)\,, \quad t = T-1, T-2,\cdots,1 \quad i = 1,\cdots,N \tag{4.11}$$

$\beta_T(i) = 1.0$, jos tila i on sallittu lopputila ja 0 muuten. $\beta_t(i)$ on siis todennäköisyys, että tehdään havainnot $O_{t+1}, O_{t+2}, \cdots, O_T$ lähtien tilasta i hetkellä t.

α ja β todennäköisyyksien avulla lasketaan $\xi_t(i,j)$, joka on todennäköisyys, että ollaan tilassa i hetkellä t ja siirrytään tilaan j.

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \tag{4.12}$$

Todennäköisyys, että ollaan hetkellä t tilassa i, saadaan helposti laskemalla yhteen $\xi_t(i,j)$:t kaikilla j:n arvoilla.

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{4.13}$$

Sijoittamalla kaava (4.12) (4.13):een nähdään, että kyseessä todellakin on todennäköisyys tilalle i hetkellä t.

$$\gamma_t(i) = \frac{\alpha_t(i)\sum_{j=1}^{N} a_{ij} b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\alpha_t(i)\sum_{j=1}^{N} a_{ij} b_j(O_{t+1})\beta_{t+1}(j)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \tag{4.14}$$

Tilasta i tilaan j siirtymien määrän odotusarvo jaettuna kaikkiin tiloihin siirtymisten odotusarvolla antaa hyvän estimaatin tilojen väliselle siirtymätodennäköisyydelle a_{ij}.

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T}\xi_t(i,j)}{\sum_{t=1}^{T}\gamma_t(i)} \tag{4.15}$$

Tilasta i saatavien havaintojen jakauman parametreille saadaan suurimman uskottavuuden estimaatti jakamalla tilassa i saatujen symbolin v_k havaintojen määrän odotusarvo kaikkien kyseisessä tilassa käyntien odotusarvolla.

$$\hat{b}_i(v_k) = \frac{\sum_{t=1}^{T}\begin{cases}\gamma_t(i) & \text{kun } O_t = v_k\\ 0 & \text{muuten}\end{cases}}{\sum_{t=1}^{T}\gamma_t(i)} \tag{4.16}$$

4.6 HMM:t puheentunnistuksessa

Useimmissa menestyksellisissä puheentunnistusjärjestelmissä käytetään puheen mallina kätkettyjä Markov-malleja [Lee 1988], [Levinson 1983] jne. HMM:illä voidaan luontevasti yhdistää puheen hetkellisen spektrin mallittaminen stationaarisena stokastistisena prosessina ja näiden prosessien temporaaliset suhteet Markovin ketjulla.

Puheen hetkellisestä spektristä lasketut piirrevektorit voidaan kuvata diskreetillä symbolijakaumalla vektorikvantisoinnin avulla ja estimoida edellä kuvatulla menetelmällä (kaava 4.16) jakauman parametrit. Tämäntyyppistä lähestymistapaa voidaan kutsua vektorikvantisointi-HMM:iksi tai diskreeteiksi HMM:iksi [Rabiner et al. 1983], [Torkkola et al. 1991a].

Toinen lähestymistapa on mallittaa piirrevektorit jatkuvalla jakaumalla, esimerkiksi normaalijakaumalla tai usean normaalijakauman painotetulla summalla. Jatkuvan jakauman parametrit voidaan estimoida suurimman uskottavuuden menetelmällä kuten diskreetin jakauman parametritkin ja estimointikaavoja on johdettu mm. viitteissä [Juang 1985], [Liporace 1982] ja [Bahl et al. 1983].

Jatkuvan ja diskreetin havaintojakauman yhdistelmiäkin on kokeiltu piirrevektorijakauman approksimoinnissa. Esimerkiksi ns. puolijatkuvilla (semi-continuous) malleilla on saatu hyviä tuloksia [Huang et al. 1989b], [Huang et al. 1989a]. Puolijatkuvissa malleissa on vektorikvantisoinnin koodikirjavektorit korvattu normaalijakaumilla, joiden keskikohta yhtyy koodikirjavektoriin. Lähimpään koodikirjavektoriin kvantisoinnin sijasta lasketaan kunkin jakauman todennäköisyydet painotettuna jakaumaa vastaavan ulostulosymbolin todennäköisyydellä.

Erityisesti jatkuvan jakauman HMM:iä käytettäessä on riittävän hyvien alkuarvojen määrittäminen jakaumille osoittautunut mallien opetuksen kannalta tärkeäksi (ja vaikeaksi). Tilojen todennäköisyysjakaumat ovat monihuippuisia ja niiden kuvaamiseksi tarvitaan usein monen (jopa yli kymmenen) normaalijakaumien yhdistelmä. Tällöin on huomattu, että piirrekomponenttien kovariansseja eri jakaumilla ei saada luotettavasti estimoitua, koska riittävän kattavaa puheaineistoa ei ole kerätty. Täydellisten kovarianssimatriisien sijasta onkin päädytty käyttämään diagonaalisia matriiseja [Rabiner 1989]. Kun osajakaumia on paljon, niiden hyödyntämiseksi vaaditaan hyvät alkuarvot, jotta Baum-Welch –algoritmi toimisi kunnolla. Hyviä menetelmiä alkuarvojen laskemiseksi on kirjallisuudessa esitetty niukasti. Viitteessä [Juang et al. 1990] on esitetty eräs K-means –menetelmään [MacQueen 1967] pohjautuva algoritmi. Tämän työssä on kehitetty menetelmä käyttää itseorganisoituvia piirrekarttoja [Kohonen 1982] ja oppivaa vektorikvantisointia [Kohonen 1986b].

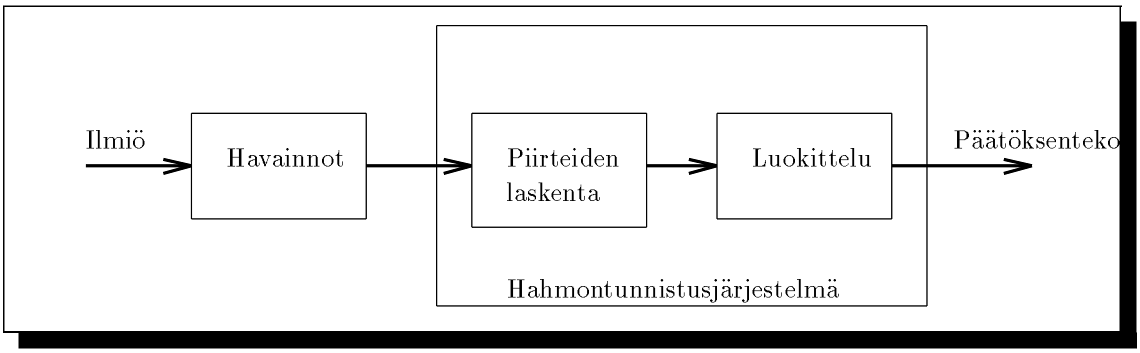
Luku 5

Vektorikvantisointi ja itseorganisoituvat piirrekartat

Tässä luvussa esitellään menetelmiä, joilla saavutetaan mahdollisimman hyvä ja erottelukykyinen jakauma-approksimaatio foneemin tilasta saatujen havaintojen tiheysfunktiolle. Jatkuvaa havaintojakaumaa käytettäessä voidaan soveltaa vektorikvantisointimenetelmiä sijoiteltaessa todellisen jakauman tihentymiä kuvaavia parametrisia jakaumia sopiville paikoille. Todellista jakaumaa approksimoidaan näiden parametristen jakaumien lineaarikombinaatiolla.

5.1 Piirrekartat ja vektorikvantisointi

5.1.1 Piirteet



Kuva 5.1: *Hahmontunnistustehtävän päävaiheet.*

Hahmontunnistuksen toinen pääongelma on ilmiöiden luokittelun lisäksi sopivien ilmiötä mahdollisimman hyvin kuvaavien ja luokitteluun sopivien piirteiden valinta ja niiden irroittaminen tehdyistä havainnoista. Piirreirroituksen tavoite on muodostaa yleensä hyvin korkeadimensioisen havaintoavaruuden kuvaus matalampidimensioiseen piirreavaruuteen säilyttäen samalla mahdollisimman suuren osan havaintojen sisältämästä informaatiosta.

Piirreirroituksen avulla luokittelun laskennallinen kompleksisuus vähenee usein ratkaisevasti, jolloin automaattiseen luokitteluun kuluva aika ja vaadittava muistitila saadaan pidettyä mielekkäänä. Samalla saadaan luokittelijan parametrit estimoitua luotettavammin, koska opetukseen käytettävää esimerkkiaineistoa on enemmän kutakin parametria kohden. Tällöin myös luokitteluun perustuvat päätökset ovat luotettavampia, mikäli piirreirroituksessa ei pääse häviämään liikaa luokittelun kannalta oleellista informaatiota.

Hyvälle piirreirroitusmenetelmälle on tyypillistä, että piirteet eivät korreloi voimakkaasti toistensa kanssa. Piirteiden välisen riippumattomuuden etu on siinä, että samaa informaatiota ei ole sisällytetty moneen komponenttiin vaan kukin komponentti ilmentää erillaisia ominaisuuksia. Mikäli useampi piirre kuvaa samaa ominaisuutta, on ehkä mahdollista vähentää vielä piirteiden määrää menettämättä kuitenkaan oleellista informaatiota.

Luokittelijan toteuttamista helpottaa, jos piirteet ovat suunnilleen samaa suuruusluokkaa ja niiden merkitsevyys ilmenee komponenttien saamien arvojen varianssista. Kun samaan luokkaan kuuluvien näytteiden piirteet ovat keskenään melko samanlaisia ja eri luokkien näytteet tuottavat keskenään hyvin erilaisia piirteitä on tehokkaiden luokittelusääntöjen muodostaminen helppoa. Luokittelijan on kuitenkin pystyttävä vertailemaan eri piirteissä vallitsevien erojen merkityksiä, jotta luokittelupäätös voidaan tehdä vaikeissakin tilanteissa.

Vaikeissa hahmontunnistusongelmissa (esimerkiksi puheentunnistuksessa) on tavallista, että on tehtävä hyvin huolellinen piirreirroitus, jotta ongelma saadaan yleensä ratkaistua tyydyttävällä tarkkuudella. Silloin piirreirroitus voi viedä huomattavan osan koko tehtävän suoritukseen käytettävästä ajasta ja sen laskennalliseen toteutukseen on kiinnitettävä erityistä huomiota. Piirteiden irroituksen on siis oltava laskennallisesti mahdollisimman kevyt operaatio ja se pitäisi olla totetutettavissa tietokoneella yksinkertaisesti ja tehokkaasti.

5.1.2 Vektorien samanlaisuuden mittaaminen

Luokittelutehtävissä kuluu valtaosa laskentatyöstä vektorien samanlaisuuden mittaamisiin. Vektorien vertailun perusteella joudutaan tekemään päätös siitä, mihin luokkaan kuuluvia näytevektoreita tunnistettava vektori eniten muistuttaa tai minkä luokan todennäköisyysjakaumaan se parhaiten sopii.

Mitä vaikeampi luokitteluongelma on sitä huolellisemmin on vektorien mittausmenetelmä valittava, sillä luokkien määrän ja piirrevektorien dimensioiden kasvaessa vektorien vertailuun kuluvan ajan osuus lisääntyy nopeasti. Erityisesti jos luokkien todennäköisyysjakaumat ovat monimutkaisia, on niitä kuvattava monella vektorilla, joihin saadun havainnon piirrevektoria joudutaan vertaamaan. Vertailun pitää siis olla nopea, mutta koska sen vaikutus luokittelun onnistumiseen on merkittävä, vertailu on suoritettava huolellisesti.

Tavallisimpia vektorien samanlaisuuden mittaamiseen käytettyjä menetelmiä on vektorien välinen euklidinen etäisyys (tai sen neliö).

$$d(x(t), m_i) = \|x(t) - m_i\|^2 = \sum_{k=1}^{K}(x_k - m_{ik})^2 \tag{5.1}$$

Tässä $x(t) = [x_1 x_2 \cdots x_K]^T$ on hetkellä t saadusta havainnosta laskettu piirrevektori ja m_i vertailuvektori.

Euklidista etäisyyttä yleistetään määräämällä kullekin vektorin komponentille oma painokerroin, joka kuvaa siinä komponentissa olevan eron merkitsevyyttä suhteessa kokonaiseroon. Painokerroin voi olla esimerkiksi σ_{ik}:n, eli komponentin k varianssin luokkaan i kuuluvilla näytevektoreilla, käänteisluku, jolloin komponentissa, jossa yleensä poikeamat ovat pieniä, sattuva suuri poikkeama on merkittävä.

$$d_D(x(t), m_i) = \sum_{k=1}^{K} \frac{(x_k - m_{ik})^2}{\sigma_{ik}^2} \tag{5.2}$$

Jos näytteitä on riittävästi voidaan kunkin luokan näytteille estimoida täydelliset kovarianssimatriisit ja käyttää Mahalanobis-etäisyyttä [Therrien 1989].

$$d_M(x(t), m_i) = (x(t) - m_i)^T \Sigma_i^{-1}(x(t) - m_i), \tag{5.3}$$

Mahalanobis-etäisyys ottaa huomioon piirrevektorien komponenttien keskinäiset riippuvuudet ja antaa luotettavan kuvan näytteiden samanlaisuudesta, vaikka piirrevektorin komponentit korreloisivatkin keskenään. Ongelma on kuitenkin se, että varsinkin jos piirrevektorien dimensio on korkea, joudutaan estimoimaan paljon parametreja ja hyvien tulosten saavuttamiseen vaaditaan laajaa opetusaineistoa.

Tehokas ja nopea tapa yhtä pitkien vektorien vertailuun on laskea vektoreiden sisätulo.

$$d_S(x(t), m_i) = x(t)^T m_i \quad m_i^T m_i = 1 \ \forall i \tag{5.4}$$

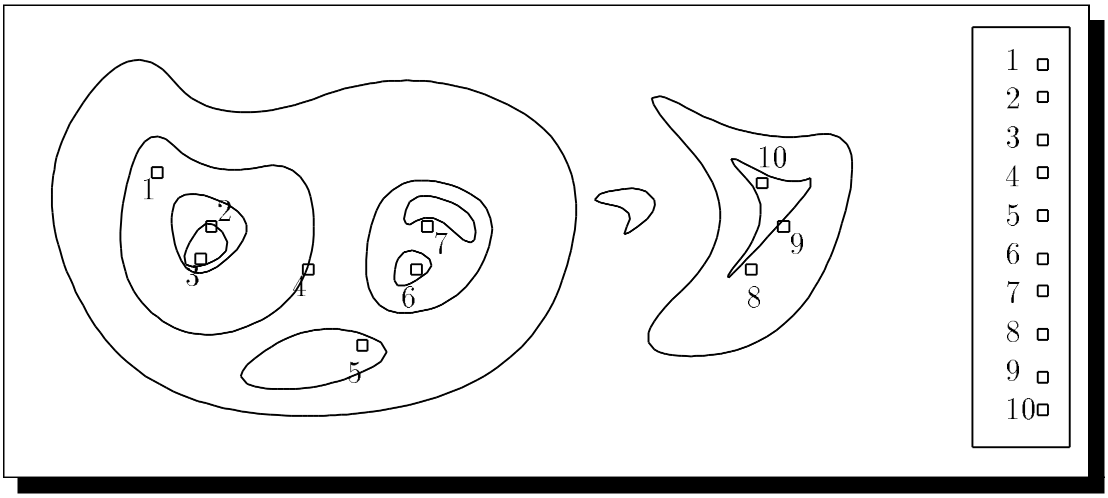
Samansuuntaisten pituudeltaan ykkösiksi normeerattujen vektoreiden sisätulo on 1, toisiaan vastaan kohtisuorien 0 ja vastakkaissuuntaisten -1. Menetelmä on kevyt ja se antaa valmiin sopivaksi skaalatun tuloksen, mutta käyttöä hankaloittaa jokaisen vektorin pituuden normalisointi.

5.1.3 Vektorikvantisointi

Vektorikvantisoinnissa (VQ) jatkuvaa todennäköisyysjakaumaa noudattavaa vektoria x approksimoidaan äärellisellä määrällä koodikirjavektoreita $m_i, i = 1, \cdots, N$ (kuva 5.2). Vektori x kuvataan valitun etäisyysmitan mukaan lähimmällä koodikirjavektorilla. Tavallisesti etäisyysmitaksi valitaan euklidinen etäisyys eli $\|x - m_i\|$.

Koodikirjavektorit pyritään sijoittamaan siten että kvantisointivirhe eli lähimmän vektorin etäisyys x:stä ($d(x, m_c)$) olisi mahdollisimman pieni. Minimoitava lauseke on tällöin muotoa

$$E = \int d(x, m_c)p(x)dx \tag{5.5}$$



Kuva 5.2: *Kaksiulotteisen todennäköisyysjakauman kuvaaminen yksiulotteisen koodikirjan kymmenellä koodikirjavektorilla. Jakauman tiheimmät kohdat on kuvattu "korkeuskäyrillä" ja pienet neliöt esittävät vektoreiden koordinaatteja.*

$p(x)$ on x:n tiheysfunktio. Koska lähimmän koodikirjavektorin m_c valinta tapahtuu jokaiselle x:lle erikseen, ei minimointia voida ratkaista suljetussa muodossa, vaan on tyydyttävä iteratiivisiin ratkaisumenetelmiin.

Kun lähin koodikirjavektori m_c valitaan euklidisen etäisyyden perusteella, voidaan osoittaa, että seuraava gradienttimenetelmä lähestyy asymptoottisesti optimaalista ratkaisua.

$$m_i(t+1) = \begin{cases} m_i(t) + \alpha(t)(x(t) - m_i(t)) & \text{kun } i = c \\ m_i(t) & \text{muuten} \end{cases} \tag{5.6}$$

$x(t)$ on satunnaisesti valittu tiheysfunktiota $p(x)$ noudattava näyte ja $\alpha(t)$ on monotonisesti pienenevä positiivinen oppimiskerroin, jonka arvo on pienempi kuin yksi. Koodikirjavektoreiden alkuarvot $m_i(0)$ valitaan joko satunnaisesti tai käyttäen jotain sopivaa likimääräismenetelmää.

Kaavan 5.6 päivitykset voidaan tehdä myös valitsemalla ensin kaikki kutakin koodikirjavektoria lähinnä olevat vektorit ja päivittämällä vektoria m_i sitten keskitetysti. K-means algoritmissa [MacQueen 1967] m_i päivitetään valittujen näytteiden keskiarvoksi.

5.1.4 Kvantisointikoodikirjan koko ja muoto

Piirrevektoriavaruuden tihentymien muodot ja suurin sallittu kvantisointivirhe ratkaisevat yleensä sen, kuinka suuri koodikirja on valittava. Koodikirja laaditaan siten, että kvantisointivirhe eli kunkin moniulotteisen vektoriavaruuden pisteen etäisyys lähimmästä koodikirjavektorista olisi mahdollisimman pieni. Tällöin koodikirjavektoreiden sijoittelu approksimoi vektorien tiheysfunktiota; siellä missä vektorien esiintymistodennäköisyys on suurin tarvitaan eniten koodikirjavektoreita.

Jos halutaan, että piirrevektoriavaruudessa lähekkäiset pisteet kuvautuisivat koodikirjalla lähekkäisiksi koodivektoreiksi, on valittava koodikirjalle sellainen topologia, jossa kullakin yksiköllä on useita naapureita. Naapureita on luonnollisesti sitä enemmän mitä enemmän koodikirjalla on dimensioita ja mitä vähemmän reuna-alueita. Sopivan koon ja muodon valitsemiseksi on yleensä tehtävä useita kokeita erilaisilla koodikirjoilla.

5.1.5 Koodikirjalta vaadittavat ominaisuudet

Aikaisemmin sanotun perusteella voidaan tiivistetysti ilmaista, että hyvältä koodikirjalta edellytetään ainakin seuraavia seikkoja:

- Pieni kuvausvirhe
- Piirreavaruuden topologia säilyy
- Koodikirja on mahdollisimman pieni

Avaruuden topologian säilyttäminen eli lähekkäisten pisteiden kuvautuminen lähekkäisille yksiköille on tarpeellista, mikäli avaruuden eri osat esittävät eri luokkia ja luokat halutaan kuvata myös koodikirjalle mahdollisimman hyvin.

Koodikirjan pitäminen mahdollisimman pienenä on luonnollisesti tavoittelemisen arvoista, koska suuren koodikirjan käsittely on hidasta ja koodikirjavektoreiden sijoitteluun tarvitaan paljon ja kaikkia ominaisuuksia oikeissa suhteissa edustavaa aineistoa. Suuren koodikirjan opettaminen on lisäksi hidasta ja käytettäessä suppeaa esimerkkiaineistoa on tavallista, että koodikirja kuvaa esimerkkiaineiston pienellä virheellä, mutta sen sijaan muiden näytteiden kuvaaminen onnistuu huonommin.

5.2 Itseorganisaatio

5.2.1 Itseorganisointialgoritmi

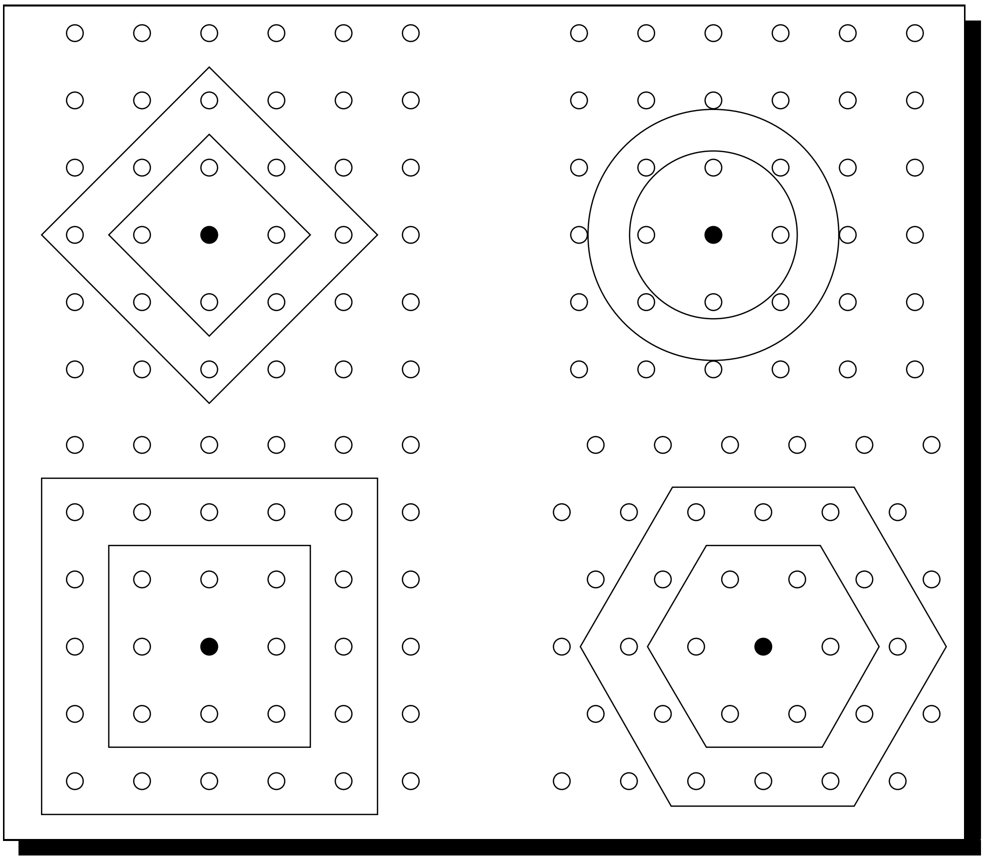
Edellä esitetyt vaatimukset täyttävän koodikirjan konstruointi käytännössä on ongelma, jolle ei löydy suoraa ratkaisua. Iteratiivisia algoritmeja on kuitenkin olemassa ja niistä tärkein on professori Kohosen kehittämä itseorganisoituva piirrekartta [Kohonen 1982], jossa koodikirjavektoreiden sijoittelua parannetaan vähitellen ja joka tuottaa valitusta kartasta piirrevektorien tiheysfunktiota mahdollisimman hyvin approksimoivan kuvauksen pyrkien samalla topologian säilyttämiseen. Prosessin matemaattinen tarkastelu on kuitenkin vaikeaa ja asymptoottisia ominaisuuksia ja optimaalisen organisoitumisen saavuttamista on toistaiseksi tutkittu vain erikoistapauksissa mm. [Kohonen 1989], [Ritter 1989], [Ritter et al. 1986] ja [Ritter et al. 1988].

Koodikirjan parantaminen tapahtuu valitsemalla piirrevektorien joukosta esimerkkivektori x ja siirtämällä lähintä koodikirjavektoria m_c ja sen naapureita hieman lähemmäksi

x:ää. Päivitettävä joukko N_c on siis koodikirjavektorin m_c naapurusto kartalle valitun topologian mielessä. Koodikirjavektorin muutoksen määrän määräävä kerroin $\alpha(t)$ saa t:n kasvaessa väheneviä skalaariarvoja väliltä $0 < \alpha(t) < 1$ ja sitä sanotaan opetusvakioksi. Päivityskaava on

$$m_i(t+1) = \begin{cases} m_i(t) + \alpha(t)(x(t) - m_i(t)) & \text{kun } i \in N_c(t) \\ m_i(t) & \text{kun } i \notin N_c(t) \end{cases} \tag{5.7}$$

5.2.2 Topologinen ympäristö



Kuva 5.3: *Erilaisia tapoja määrätä naapurusto neliöhilaksi ja heksagonaaliseksi hilaksi järjestetyillä kartoilla. Kussakin tapauksessa on esitetty kaksi erisuuruista naapurustoa.*

Valitun etäisyysmitan mukaisen lähimmän yksikön opetuksessa päivitettävä naapurusto $N_c(t)$ määritellään kartan topologian perusteella. Tavallisimmissa topologioissa kartta on järjestetty joko neliöllisen tai heksagonaalisen hilan (kuva 5.3) muotoon. Neliöhilassa naapurusto voidaan määrätä esimerkiksi yksiköiden välisien pysty- ja vaakakoordinaattierojen summana, euklidisena etäisyytenä tai suurimman koordinaattieron perusteella. Heksagonaalihilassa voidaan etäisyydet määrätä kätevästi ja nopeasti käyttäen koordinaattiakselien suuntaisten siirtymien summaa, kun käytössä on kolme akselia: vaaka-akseli ja kaksi vinoakselia. Neliöhilassa yksiköllä on neljä (tai kahdeksan) lähintä naapuria ja heksagonaalihilassa vastaavasti kuusi.

Kartan itseorganisoinnissa topologian säilyvyydestä huolehditaan parhaiten valitsemalla naapuruston säde aluksi suureksi ja pienentämällä sitä tasaisesti iteroinnin aikana. Naapurusto voi organisoitumisen alussa kattaa lähes kaikki kartan yksiköt ja loppuvaiheessa, kun järjestäytyminen on saavutettu, voidaan keskittyä vain lähimmän yksikön $N_c = \{c\}$ opetukseen.

5.2.3 Opetusvakion merkitys

Kartan järjestäytymistä voidaan kontrolloida säätelemällä opetusvakion α suuruutta. Kun α on lähellä ykköstä kartan kuvausvirheet pienenevät aluksi nopeasti, koska yksikköjen painovektorit siirtyvät lähelle opetuksessa käytettäviä piirrevektoreita. Varsin pian joudutaan kuitenkin tilanteeseen, jossa yksikköjen olisi kyettävä yleistämään useamman näytevektorin ominaisuuksia, mikä ei onnistu suurella opetusvakiolla vaan johtaa yksikköjen edestakaiseen liikehdintään.

Pienellä opetusvakiolla (0.01:n suuruusluokkaa) yksikköjen muutokset ovat pieniä ja kuvausvirhe pienenee hitaasti. Kuvauksen paraneminen kestää kuitenkin pitempään ja tuloksena on kartta, jonka yksiköiden sijoittelu approksimoi hyvin opetusnäytteiden tiheysfunktiota. Käytönnön sovellusten kannalta kartan konvergoituminen saattaa kuitenkin olla liian hidasta.

Paras oppimiskäyttäytyminen saavutetaan, kun opetusvakiota pienennetään iteroinnin aikana vähitellen. Usein käytetään iteroinnin kierroslukuun kääntäen verrannollista opetusvakiota eli $\alpha(t) \sim 1/t$. Käytännössä ei α:n pienentämisen funktionaalisella muodolla ole kuitenkaan suurta merkitystä, joten voidaan käyttää myös lineaarisesti pienenevää opetusvakiota. Teoreettisesti on todistettavissa [Ritter et al. 1988], että kartta organisoituu optimaaliseksi todennäköisyydellä 1, kun alkutila on riittävän hyvä ja $\alpha(t)$ riittävän pieni positiivinen funktio, joka toteuttaa ehdot:

$$\lim_{T \to \infty} \int_0^T \alpha(t) dt = \infty \tag{5.8}$$

$$\lim_{T \to \infty} \alpha(T) = 0 \tag{5.9}$$

On mahdollista määritellä opetusvakio riipuvaksi käytetyn naapuruston koosta siten, että lähimpiä naapureita opetetaan enemmän kuin kaukaisempia. Sopiva menetelmä on valita opetusvakion lasku edettäessä kohti naapuruston ulkoreunaa Gaussin kellokäyrän mukaan [Kohonen 1990].

Opetusvakiot voidaan asettaa myös yksikkökohtaisiksi (OLVQ1) [Kohonen et al. 1992]. Tällöin kunkin yksikön opetusvakio on aluksi sama, mutta jokaisella iteraatiokierroksella vähennetään vain näytevektoria lähimmän yksikön oppimisnopeutta. Yksikkökohtaisella opetusvakiolla varmistetaan, että eri vaiheissa tehtyjen korjausten vaikutukset lopputulokseen ovat yhtäsuuria ja organisoitumisen nopeus on paras mahdollinen. Intuitiivisesti ajatellen todennäköisyysjakauman tiheälle alueelle kulkeutunut koodikirjavektori, jota päivitetään usein, ankkuroituu näin paikoilleen nopeasti. Harvemmalle alueelle jäänyt vektori joutuu sensijaan pidempään vaeltelemaan näytteiden mukana etsien sopivaa paikkaa itselleen.

5.3 Oppiva vektorikvantisointi eli LVQ

5.3.1 LVQ:n päämäärä

Vektorikvantisaation koodikirjavektoreita käytetään usein luokitteluun, jolloin näytevektorit x luokitellaan lähimmän koodikirjavektorin luokan muokaan. Tyypillisessä tilanteessa koodikirjavektoreita on huomattavasti enemmän kuin mahdollisia luokkia, joten luokkaa edustaa useampia erilaisia vektoreita.

Luokittelun päämäärä on luokitteluvirheiden määrän minimoiminen, joten kvantisointivirheellä ei enää ole merkitystä. Luokittelussa on myös yhdentekevää mikä samaan luokkaan kuuluvista koodikirjavektoreista sijoittui lähimmäksi näytettä x.

Oppivalla vektorikvantisoinnilla [Kohonen 1986b] pyritään siirtämään koodikirjavektorit niin, että luokkien välinen erottelukyky paranisi. Viitteessä [Kohonen 1990] esitetään kolme eri tapaa (LVQ1, LVQ2 ja LVQ3) opettaa vektoreita tämän tavoitteen saavuttamiseksi. Lähtökohtana on yleensä näytevektoreiden tiheysfunktiota approksimoivan vektorikvantisoinnin antamat alkuarvot (itseorganisoituva piirrekartta, K-means tms.). Koodikirjavektoreiden edustamat luokat määrätään helpoimmin tutkimalla valmiiksi tunnistettuja näytteitä ja valitsemalla todennäköisin luokka enemmistöäänestyksellä.

5.3.2 LVQ1

Euklidista etäisyyttä käyttäen valitaan kutakin näytevektoria x lähinnä oleva koodikirjavektori m_c ja verrataan sen edustamaa luokkaa x:n todelliseen luokkaan. Koodikirjavektorit päivitetään seuraavan kaavan mukaisesti.

$$m_i(t+1) = \begin{cases} m_i(t) + \alpha(t)(x(t) - m_i(t)) & \text{kun luokka on oikea ja } i = c \\ m_i(t) - \alpha(t)(x(t) - m_i(t)) & \text{kun luokka on väärä ja } i = c \\ m_i(t) & \text{kun } i \neq c \end{cases} \tag{5.10}$$

5.3.3 LVQ2

LVQ2:ssa tarkistetaan ennen vektorien siirtämistä, että ne todella sijaitsevat luokkien rajalla. Päivityksessä korjataan sekä lähintä että toiseksi lähintä vektoria. Tarkastus toteutetaan sallimalla kaavan 5.10 mukaiset koodikirjavektorien korjaukset vain siinä tapauksessa, että lähin koodikirjavektori m_c edustaa väärää luokkaa ja toiseksi lähin $m_{c'}$ oikeaa. Lisäksi vaaditaan, että näytevektori x sijaitsee noin m_c:n ja $m_{c'}$:n puolivälissä (pieni poikkeama sallitaan esimerkiksi 10%). Näiden ehtojen toteutuessa siirretään m_c:tä pois päin x:stä ja $m_{c'}$:a kohti x:ää.

$$m_i(t+1) = \begin{cases} m_i(t) - \alpha(t)(x(t) - m_i(t)) & \text{kun } i = c \\ m_i(t) + \alpha(t)(x(t) - m_i(t)) & \text{kun } i = c' \\ m_i(t) & \text{muuten} \end{cases} \tag{5.11}$$

5.3.4 LVQ3

Kolmas LVQ-algoritmi on kehitetty pitäen tavoitteena edellisten hyvien puolien yhdistämistä ja sivuvaikutusten eliminoimista.

Koodikirjavektoreiden korjaus on aina verrannollinen näytteen x etäisyyteen korjattavasta vektorista. LVQ2:ssa (kaava 5.11) siirretään tällöin oikeaa luokkaa edustavaa koodikirjavektoria aina hieman enemmän kohti x:ää kuin väärän luokan edustajaa pois päin x:stä. Jokainen korjaus siis hieman lyhentää kahden eri luokkaan kuuluvan koodikirjavektorin välistä etäisyyttä.

Kun LVQ2:n opetussääntöä korjataan siten, että näytevektorin x sijoittuessa kahden koodikirjavektorin puoleen väliin (likimäärin), jotka molemmat edustavat oikeaa luokkaa, siirretään molempia vektoreita LVQ1- säännön mukaan (kaava 5.10), kuitenkin hieman vähemmän ($\epsilon < 1$) kuin ristiriitatapauksissa.

Siis kun m_c edustaa väärää luokkaa ja $m_{c'}$ oikeaa

$$m_i(t+1) = \begin{cases} m_i(t) - \alpha(t)(x(t) - m_i(t)) & \text{kun } i = c \\ m_i(t) + \alpha(t)(x(t) - m_i(t)) & \text{kun } i = c' \\ m_i(t) & \text{muuten} \end{cases} \tag{5.12}$$

Ja kun m_c ja $m_{c'}$ molemmat edustavat oikeaa luokkaa

$$m_i(t+1) = \begin{cases} m_i(t) + \epsilon\alpha(t)(x(t) - m_i(t)) & \text{kun } i = c \text{ tai } i = c' \\ m_i(t) & \text{muuten} \end{cases} \tag{5.13}$$

Luku 6

Vektorikvantisoinnin yhdistäminen jatkuviin Markov-malleihin

6.1 Miksi?

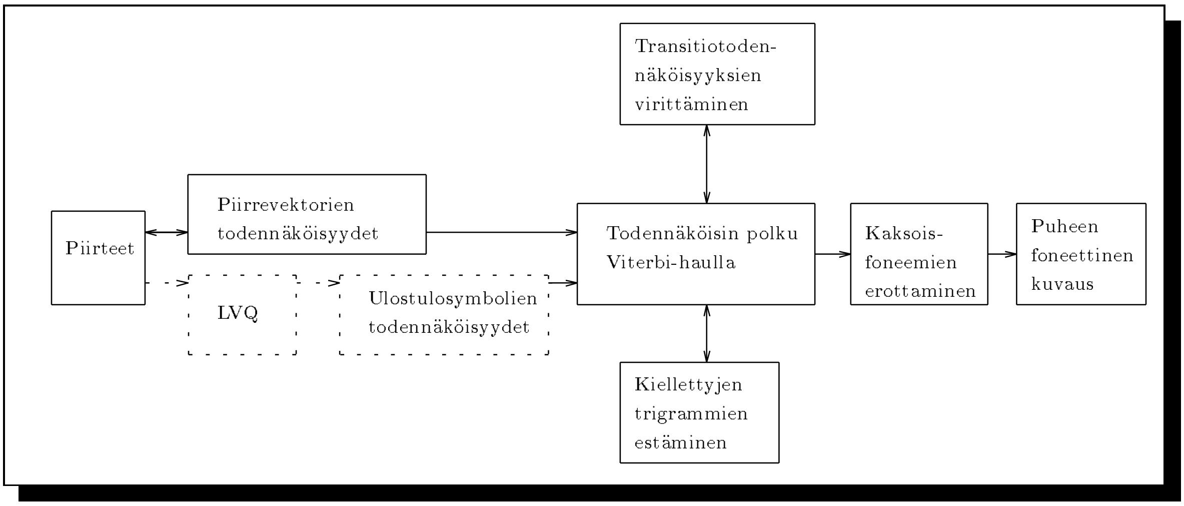
Jatkuvan havaintojakauman kätkettyjä Markov-malleja on hyvin yleisesti sovellettu puheentunnistukseen ja vieläpä erinomaisin tunnistustuloksin (mm. [Nakagawa et al. 1990] [Rabiner 1989] ja [Levinson 1983]). Silti sekä opetusmenetelmissä että itse malleissa on havaittu puutteita.

Monimutkaisia jakaumia kuvattaessa tarvitaan useiden parametristen jakaumien yhdistelmiä. Kun tavallisimmin käytetyllä Baum-Welch –algoritmilla opetetaan monen jakauman kombinaatiota, saavutettavan mallin laatu riippuu selvästi parametrien alkuarvoista. Jos alkuarvot ovat pahasti pielessä, algoritmi konvergoi hitaasti eivätkä parametrit koskaan saavuta hyviä arvoja. Tehokkaita menetelmiä estimoinnin tehostamiseksi ei kuitenkaan ole liiemmin esitetty. Parhaat menetelmät edellyttävät yleensä opetusaineiston manuaalista segmentointia tilajonoiksi. Lisäksi tilan tuottamat piirrevektorit on jollain keinolla ryhmiteltävä, jotta parametrisille osajakaumille saadaan toisistaan poikkeavat alkuarvot. Tähän on käytetty esimerkiksi K-means –algoritmia [Juang et al. 1990]. Tässä työssä kokeillaan itseorganisoituvien piirrekarttojen [Kohonen 1982] ja oppivan vektorikvantisoinnin [Kohonen 1986b] soveltamista.

Parametrien määrän valinta tilojen tiheysfunktioden malleissa on osoittautunut ongelmalliseksi puheentunnistuksessa. Tavallisesti parametriset mallit ovat normaalijakaumien lineaarikombinaatioita. Puheen piirrevektorit ovat normaalisti noin 20-dimensioisia, joten niiden normaalijakaumat tarvitsevat kukin varsin paljon parametreja. Foneemimalleja tarvitaan kielestä riippuen 20 – 40, difoni- ja trifonimalleissa lukumäärä kasvaa eksponentiaalisesti. Lisäksi kukin malli sisältää muutamia erillisiä tiloja ja kunkin tilan havaintojakauma vaatii toimiakseen useiden normaalijakaumien yhdistelmän. Käytännössä saatavilla oleva puheaineisto ei yleensä ole riittävän monipuolinen, jotta näin valtavan mallin kaikille parametreille saataisiin luotettavat estimaatit. Jostakin on siis tingittävä, sillä myös koneiden keskusmuisti ja laskentakapasiteetti asettavat rajoituksia. Tässä kuvatussa järjestelmässä foneemeilla on erilliset mallit, piirrevektorien komponentit oletetaan riippumattomiksi ja komponenttien keskinäinen painotus on kaikilla jakaumilla sama.

6.2 Puheen foneettisen kuvauksen tuottaminen

6.2.1 Tunnistuksen vaiheet



Kuva 6.1: *Foneettisen kuvauksen tuottaminen kätketyillä Markov-malleilla. Katkoviivalla piirretty reitti liittyy diskreetin jakauman HMM:iin, jotka järjestelmässä ovat tällä hetkellä käytössä. Tämän luvun kaavioissa katkoviivaa käytetään esittämään vaihtoehtoja, joita tässä työssä ei ole kokeiltu.*

Lasketaan puhenäytteestä ensin piirrevektorit kuten luvussa kolme. Seuraavaksi määritetään kvantisointivirheiden ja jatkuvien havaintojakaumien avulla havaintotodennäköisyydet $b_i(O_t)$ piirrevektoreille O_t kaikissa mahdollisissa tiloissa i.

Havaintotodennäköisyydet voidaan myös laskea (vrt. kappale 3.3.5) kvantisoimalla piirrevektori lähimpään koodikirjavektoriin ja ottamalla käyttöön sen foneemitunnusta vastaava diskreetti ulostulosymbolijakauma. Tätä menetelmää on aikaisemmin sovellettu Otaniemen puheentunnistusjärjestelmässä menestyksellisesti [Torkkola et al. 1991a].

Todennäköisin polku (taulukko 6.1) tilojen välillä saadaan soveltamalla kappaleessa 4.4 esitettyä Viterbi-algoritmia. Jos todennäköisin polku ei aina ole mielekäs, voidaan sen valintaan vaikuttaa käsittelemällä sopivasti tilansiirtotodennäköisyyksiä ja sulkemalla pois mahdottomat trigrammit (kolmen kirjaimen yhdistelmät).

Kun tilojen välinen polku on selvitetty, foneettinen kuvaus saadaan poimimalla polun varrelta kaikki foneemien väliset siirtymiset. Mikäli jonkun foneemin sisällä viivytään

```
>>>>>>>>>>>>AAAAAAAAAAAAAAAAAIIIIIIIIIIIIIIIIKKKKKKKKKKKKKKKKKAAAAAAAAAAA
0000000011230001222222222222300011122222222230000000000012222300000000122

AAAAAAAAAAAAAAAAAAAAA<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
2222222222222222222230111111111111111111111111111123
```

Taulukko 6.1: *Esimerkki tilojen välisestä polusta. Kukin kirjain kertoo käytössä olevan foneemimallin ja välittömästi sen alla oleva numero mistä tämän mallin tilasta on kyse. Yksin kirjain-numero –pari vastaa kymmenen millisekunnin mittaista ajanjaksoa.*

riittävän kauan, se tulkitaan kaksoisfoneemiksi.

6.2.2 Koemateriaali

Koemateriaalina käytetään 311 suomenkielistä sanaa, joista 112 on paikannimiä ja loput yleisiä puheessa esiintyviä sanoja. Kolme syntyperäistä suomalaista miespuolista puhujaa (kt, jk ja pu) on kukin puhunut neljään kertaan koesanaston läpi. Puhesignaalia 10 ms välein kuvaavat 21-dimensioiset piirrevektorit on muodostettu signaalin hetkellisistä kepstrikertoimista, joiden laskemisessa on käytetty luvussa kolme esitettyjä esikäsittely- ja piirreirroitusmenetelmiä, ja RMS-arvosta (kappale 3.3.3).

Mallien opetuksessa käytetään kolmea saman puhujan puhekertaa ja testauksessa neljättä. Tätä toistetaan neljä kertaa siten, että kukin puhekerta on kerran testiaineistona ja tuloksista lasketaan keskiarvo. Jos puhujaa ei erikseen mainita, kokeet on tehty kt:n puheaineistolla.

6.2.3 Menetelmän testaus

	Foneeminäytteet	Erilliset sanat		Merkityt sanat	
Puhuja	oikein%	oikein%	virhe%	oikein%	virhe%
kt	95.9	96.2	5.6	97.5	4.2
jk	91.7	93.9	7.8	94.0	7.5
pu	94.8	94.3	7.4	95.4	6.2

Taulukko 6.2: *Työssä saavutetut parhaat tulokset eri puhujilla tunnistettaessa sanoista erotettuja foneeminäytteitä, puhuttuja sanoja ja sanoja, joiden alku- ja loppuhetki on tiedetään. Prosenttiluvut ovat neljän eri puhekerran keskiarvoja. Soinnittomat klusiilit on tunnistuksessa luokiteltu samaan luokkaan.*

Puheen foneettisen kuvauksen tuottamismenetelmää testattiin tunnistamalla ensin foneeminäytteitä, jotka poimittiin yhden puhekerran sisältämistä 311 sanasta. Vastaavia muista puhekerroista poimittuja näytteitä käytettiin mallien opettamiseen. Jokaisesta foneemista pyrittiin valitsemaan 150 näytettä, mutta harvinaisempien kohdalla oli kuitenkin tyytyminen vähempään puheaineiston suppeuden takia.

Keskimmäiset sarakkeet taulukossa (6.2) kertovat, kuinka hyvin varsinainen tunnistustehtävä, testiaineiston sisältämien 311 sanan foneettisen kuvauksen tuottaminen, onnistui. Oikein% tarkoittaa oikeiden foneemien prosentuaalista osuutta ja virhe% tehtyjen foneemivirheiden määrää. Foneemivirheet syntyvät yksittäisten foneemien katoamisista, lisääntymisistä ja vaihtumisista. Virheellisen sanan sisältämä foneemivirheiden määrä määritellään pienimpänä mahdollisena virhemääränä, jotka tekemällä oikeasta sanasta päädytään tähän virheelliseen muotoon. Kyseessä on merkkijonojen välinen Levenshtein-etäisyys [Levenshtein 1966]. Nämä keskimmäiset sarakkeet ovat vertailukelpoisia sivun 28 taulukon 3.1 kanssa, jossa vastaava testi tehtiin nykyisellä järjestelmällä.

Merkittyjen sanojen tunnistaminen tarkoittaa sitä, että sanojen tarkat alku- ja loppuhetket tunnetaan. Alku- ja loppuhetket voidaan määrittää jollain muulla menetelmällä, joka käyttää esim. signaalin voimakkuutta, tai käsin.

Testitulokset kaikilla puhujilla (taulukko 6.2, erilliset sanat) ovat tarkkuudeltaan parempia kuin luvussa kolme saadut tulokset (taulukko 3.1). Nämä tulokset eivät kuitenkaan ole suoraan vertailukelpoisia, sillä diskreettien HMM:ien yhteydessä on käytetty tosiaikaisen toiminnan saavuttamiseksi pieniä karttoja.

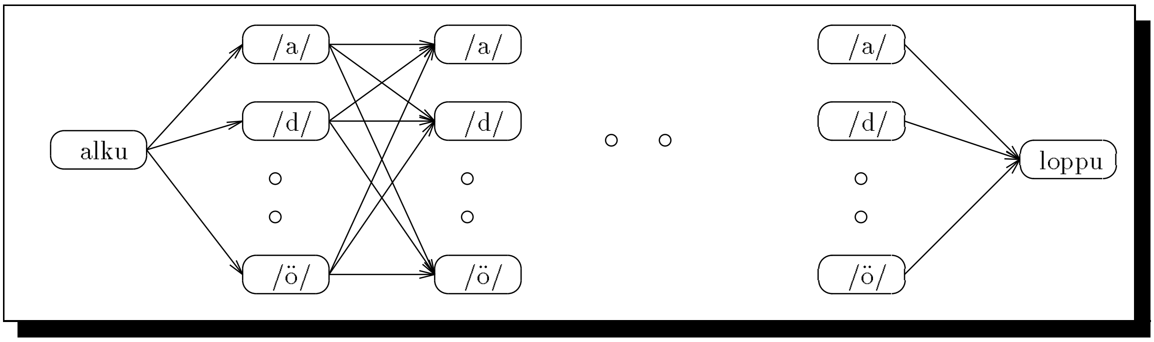
Tähän testiin on valittu tilasta saatavien havaintojen tiheysfunktioksi 25 moniulotteisen normaalijakauman yhdistelmä. Normaalijakaumien piirrekomponentit on oletettu riippumattomiksi ja niitä on kaikkia painotettu yhtä paljon. Normaalijakaumien keskiarvovektorit on sijoiteltu mahdollisimman hyville paikoille kuvaamaan tiheysfunktion huippuja käyttäen itseorganisoituvia piirrekarttoja. Markov-mallien muut parametrit on opetettu ajamalla kahdeksan kierrosta iteratiivista Baum-Welch –algoritmia. Myös huippujen sijaintia kuvaavien vektorien on annettu mukautua muiden parametrien arvojen mukaan. Seuraavissa kappaleissa on tarkemmin kuvattu, miten mallit, menetelmät ja parametrit valittiin.

6.3 Puheen foneettinen malli

6.3.1 Sanan malli

Puheen foneettisen kuvauksen tuottamiseksi havaintovektorijono paloitellaan ensin sanoja vastaaviksi osiksi. Mikäli tehtävänä on tunnistaa jatkuvaa yhtenäistä puhetta, sanojen erotteleminen ei aina ole näin yksinkertaista, mutta tässä työssä ei niihin ongelmiin puututa.

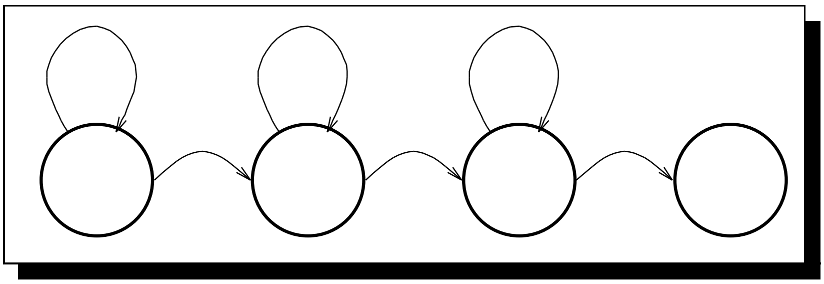
Puhuttua sanaa vastaavat havaintojonot ovat pitkiä, yleensä noin 50 – 150 peräkkäistä havaintovektoria, ja sisältävät foneemien lisäksi hieman hiljaisuutta sanan alussa ja lopussa. Foneemien mallit on luotava niin, että kunkin mallin lopputilasta määrätään siirtymätodennäköisyydet muiden foneemien mallien alkutiloihin eli estimoidaan fonee-



Kuva 6.2: *Puheesta erotetun sanan malli on yhdistelmä foneemien Markov-malleista. Foneemimallien kestot ja peräkkäisten mallien lukumäärät ovat rajoittamattomia.*

mitransitioiden todennäköisyydet. Lisäksi on tehtävä Markov-mallit alku- ja loppuhiljaisuudelle. Tilajonojen ainoa sallittu alkutila on nyt alkuhiljaisuuden mallin ensimmäinen tila ja ainoa sallittu lopputila on puolestaan loppuhiljaisuuden mallin viimeinen tila.

6.3.2 Foneemin malli



Kuva 6.3: *Foneemin Markov-malli on nelitilainen yksisuuntainen ketju. Kullakin tilalla on oma jatkuva piirrevektorijakauma ja diskreetti siirtotodennäköisyysjakauma.*

Puheen perusyksikön, foneemin, mallina on ääntämisen alkutilaa, lopputilaa ja mahdollisia välitiloja vastaava Markov-malli, joka sisältää tilojen tiheysfunktiot (kaava 6.6) ja tilojen väliset siirtymistodennäköisyydet a_{ij}.

Foneemien luonteesta johtuen Markov-mallit yksinkertaistetaan yksisuuntaiseksi ketjuksi (kuva 6.3) rajoittamalla sallittuja siirtymiä. Jos tilat numeroidaan esiintymisjärjestyksessä esimerkiksi 0,1,2 ja 3, sallittuja tilajonoja ovat vain ne, jotka alkavat 0:stä ja päättyvät 3:een ja käyvät kaikki välitilat läpi suuruusjärjestyksessä. Esimerkiksi tilajono 00011222223 on sallittu, mutta 00002211113 ja 021112222223 ovat kiellettyjä. Monimutkaisempien mallien rakentelu on tietenkin mahdollista, mutta silloin mallin estimoitavien parametrien määrä kasvaa. Mitä enemmän parametreja malli sisältää sitä laajempaa ja edustavampaa opetukseen käytettävää puheaineistoa tarvitaan. On siis valittava mahdollisimman yleinen malli, jossa kuitenkaan ei ole paljoa parametreja. Nelitilainen ketjumalli on riittävän kattava foneemimalli, sillä foneemi koostuu yleensä vain muutamasta tilasta. Kulloinkin tarvittava tilojen lukumäärä on foneemikohtainen ja voi riippua myös sanayhteydestä.

Puhuja	oikein%	virhe%
kt	95.7	6.2
jk	93.4	8.3
pu	93.7	8.0

Taulukko 6.3: *Erillisten sanojen tunnistustulokset, kun soinnittomat klusiilit /k/, /p/ ja /t/ on tunnistettu eri luokkiin.*

Suomenkielisten näytteiden perusteella on mallitettaviksi foneemeiksi valittu /a/, /d/, /e/, /h/, /i/, /j/, /k/, /l/, /m/, /n/, /o/, /p/, /r/, /s/, /t/, /u/, /v/, /y/, /ä/ ja /ö/. Vierasperäisissä sanoissa esiintyville konsonanteille ei siis ole konstruoitu mallia niiden harvinaisuuden vuoksi. Samoin /ŋ/ on jätetty mallittamatta. Muiden foneemien erotteluun sopivasti valitut piirrevektorit eivät kovin hyvin sovi soinnittomien klusiilien /k/, /p/ ja /t/:n erotteluun, joten vaikka niille tehdään kullekin oma malli ei niiden sekoittamista keskenään aina katsota virheeksi foneettisen kuvauksen tuottamisessa, sillä niitä kannattaa tarvittaessa erotella muilla keinoilla. Jos niiden sekoittaminen kuitenkin tulkitaan virheeksi testitulokset heikkenevät noin puoli prosenttiyksikköä.

Kun tunnistettaville foneemeille sekä sanan alku- ja loppuhiljaisuudelle tehdään kullekin nelitilaiset Markov-mallit, systeemi sisältää kaikkiaan 88 mahdollista puhesignaalin tilaa.

Jokaisella tilalla on oma erilainen piirrevektorien tiheysfunktio, jonka perusteella tilat ja niiden vaihtelut voidaan signaalista jäljittää. Piirrevektorin tiheysfunktiolla pyritään mallittamaan ääniväylän muotoa eli siis sitä, millaista puhesignaalia tässä asennossa oleva ääniväylä tuottaa. Koartikulaatioilmiöt ja äänteiden epäpuhtaudet tekevät piirrevektorien todennäköisyysjakaumasta yleensä monimutkaisen ja vaikeasti mallinnettavan. Hyvä tiheysfunktiomalli on kuitenkin olennainen äänteiden luokittelun kannalta.

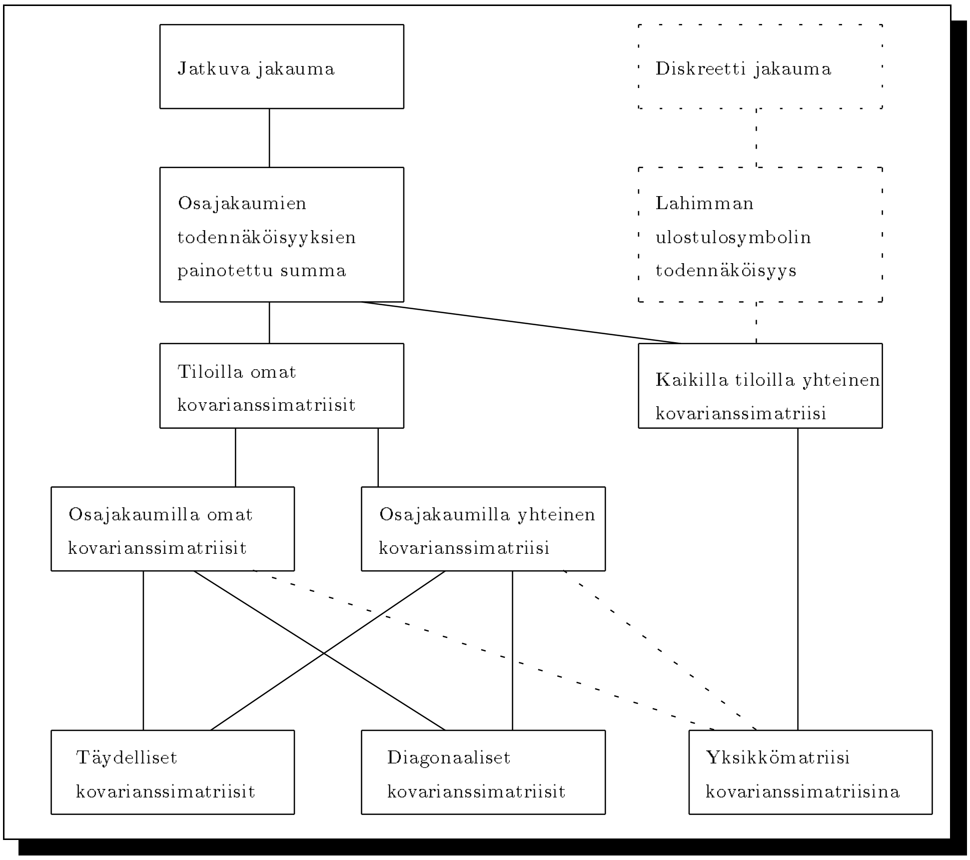
6.4 Piirrevektorin tiheysfunktio

Havainto signaalista hetkellä t on K-komponenttinen piirrevektori $O_t = (o_1, o_2, \cdots, o_K)^T$. Piirrevektorien todennäköisyysjakauma on osoittautunut mutkikkaaksi ja sen parametreja on vaikea määrittää. Jakaumaa voidaan kuitenkin approksimoida parametristen jakaumien avulla, kun valitaan riittävän monta osajakaumaa ja lasketaan niiden painotettu summa. Tehokas approksimaatiojakauma on esimerkiksi K-ulotteinen normaalijakauma.

6.4.1 Tiheysfunktion approksimointi normaalijakaumien avulla

Piirrevektorin O_t todennäköisyys K-ulotteisessa normaalijakaumassa $N(\mu, \Sigma)$ noudattaa tiheysfunktiota [Therrien 1989]

$$b(O_t) = \frac{1}{(2\pi)^{K/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(O_t-\mu)^T\Sigma^{-1}(O_t-\mu)} \tag{6.1}$$



Kuva 6.4: *Markov-malleissa käytettyjen havaintojakauma-approksimaatioiden luokittelu kovarianssimatriisien perusteella.*

Σ on K-ulotteisen normaalijakauman komponenttien välinen kovarianssimatriisi

$$\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1K} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2K} \\ . & . & \cdots & . \\ . & . & \cdots & . \\ \sigma_{K1} & \sigma_{K2} & \cdots & \sigma_{KK} \end{bmatrix} \tag{6.2}$$

ja $\mu = (\mu_1, \mu_2, \cdots, \mu_K)^T$ K-ulotteisen normaalijakauman keskiarvovektori. Kovarianssimatriisin komponentti σ_{jk} on siis piirrevektorin komponenttien j ja k tulon odotusarvo.

$$\sigma_{jk} = E(o_j - \mu_j)(o_k - \mu_k) \tag{6.3}$$

Kaava 6.1 on tulkittavissa piirrevektorin ja normaalijakauman keskiarvovektorin väliseksi Mahalanobis-etäisyydeksi [Therrien 1989]

$$d_M(O_t, \mu) = (O_t - \mu)^T \Sigma^{-1} (O_t - \mu), \tag{6.4}$$

jota on painotettu sopivasti normeeratulla eksponenttifunktiolla

$$\frac{1}{\beta} e^{-\frac{1}{2} d(O_t, \mu)} \tag{6.5}$$

Kun todennäköisyysjakaumana käytetään M:n moniulotteisen normaalijakauman yhdistelmää on tilan i tiheysfunktio yksittäisten normaalijakaumien (kaava 6.1) painotettu summa [Liporace 1982].

$$b_i(O_t) = \sum_{m=1}^{M} c_{im} b_{im}(O_t) \tag{6.6}$$

Tilan i eri normaalijakaumien painokertoimet c_{im} ovat positiivisia reaalilukuja ja niiden summa on 1.

6.4.2 Yksinkertaistettu malli

Jos oletetaan havaintovektorin komponentit lähes riippumattomiksi, kovarianssimatriisia 6.2 voidaan approksimoida diagonaalimatriisilla

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ . & . & \cdots & . \\ . & . & \cdots & . \\ 0 & 0 & \cdots & \sigma_K \end{bmatrix} \tag{6.7}$$

Tällöin tiheysfunktio 6.1 saadaan yksinkertaisempaan muotoon.

$$b(O_t) = \frac{1}{(2\pi)^{K/2}(\prod_{k=1}^{K} \sigma_k)^{1/2}} e^{-\sum_{k=1}^{K} \frac{(o_{tk}-\mu_k)^2}{2\sigma_k^2}} \tag{6.8}$$

Kaavan 6.8 eksponentti on myös tulkittavissa eräänlaiseksi piirrevektorin ja normaalijakauman keskiarvovektorin väliseksi etäisyydeksi, joka poikkeaa tavallisesta euklidisesta etäisyydestä vain vektorikomponenttien erilaisten painotuksien suhteen.

$$d_D(O_t, \mu) = \sum_{k=1}^{K} \frac{(o_{tk} - \mu_k)^2}{\sigma_k^2} \tag{6.9}$$

Toisin sanoen tätä kaavaa käytettäessä havaintovektoreiden komponenteista toiset ovat luokittelun kannalta tärkeämpiä kuin toiset ja niissä olevia eroavuuksia painotetaan suuremmilla kertoimilla. Tämä etäisyysmitta korostaa nimenomaan komponenttien erilaisia tärkeysjärjestyksiä eri jakaumissa.

Jos havaintovektorin komponenttien varianssit oletetaan keskenään lähes yhtäsuuriksi tai jos komponentit on jo valmiiksi sopivasti skaalattu niiden suhteellisen merkityksen mukaan laskujen nopeuttamiseksi, korvataan komponenttien varianssit σ_k vakiolla σ ja kaava (6.8) redusoituu edelleen.

$$b(O_t) = \frac{1}{(2\pi\sigma)^{K/2}} e^{-\frac{\sum_{k=1}^{K}(o_{tk}-\mu_k)^2}{2\sigma^2}} = \frac{1}{\beta_1} e^{-\frac{1}{\beta_2}||O_t-\mu||^2} \tag{6.10}$$

Tämä on itseasiassa havaintovektorin euklidisen etäisyyden neliö jakauman keskikohdasta mitattuna $||O_t - \mu||^2$, jota on painotettu eksponenttifunktiolla. Eksponenttifunktion painokertoimet β_1 ja β_2 voivat olla jakaumakohtaisia parametreja, mutta jos painotus on kaikkialla samanlainen tai jakaumien välillä ei muuten enää tarvita erillistä painotusta, yleistetään ne koskemaan kaikkia jakaumia.

6.4.3 Yhteisen kovarianssimatriisien käyttö osajakaumien välillä

Jos kovarianssimatriisien Σ_{im} poikkeamat tilan i eri osajakaumien välillä m voidaan olettaa vähäisiksi, estimoitavien parametrien määrä on vähenee huomattavasti. Samalla kunkin tilan kovarianssiparametrien estimointiin on käytettävissä suurempi osa opetusmateriaalista ja saatavan estimaatin luotettavuus lisääntyy.

Tilaan liittyvästä kovarianssimatriisista voidaan tehdä kolme erityyppistä mallia, kuten erillistenkin kovarianssimatriisien tapauksessa: täysi ja diagonaalinen kovarianssimatriisi sekä yksikkömatriisi. Yksikkömatriisi tuntuu ehkä melko rajulta yleistykseltä, mutta toisaalta tilakohtaisien kovarianssimatriisien parametreja on usein melko vaikea luotettavasti estimoida, koska opetusmateriaalistakaan ei voida tarkkaan määrittää, mihin tilaan kukin piirrevektori kuuluu. On mahdollista, että epäluotettavasti estimoiduista kovariansseista on tunnistuksen kannalta enemmän haittaa kuin hyötyä, joten yksikkömatriisejakin on syytä kokeilla. Tällöin päästään lisäksi käyttämään laskennallisesti nopeaa euklidista etäisyysmittaa.

Eri tilojen välisten kovarianssimatriisien yhdistäminen on myös mahdollista, mutta koska piirrevektorien esikorostus (kepstrikertoimet on kerrottu korotetulla sinikäyrällä, kappale 3.3.3) on valittu yleensä luokittelussa tärkeimpiä kertoimia sopivimmin korostavaksi, kannattaa silloin valita yhteiseksi kovarianssimatriisiksi yksikkömatriisi.

6.4.4 Tiheysfunktioapproksimaatioiden vertailu

Vertailun tarkoituksena on selvittää, kuinka monen normaalijakauman lineaarikombinaatio tuottaisi parhaat tunnistustulokset ja kuinka paljon kovarianssimatriiseja voidaan yksinkertaistaa edellä kuvattujen kaavojen (6.1 – 6.10) mukaisesti. Koska parhaat tunnistustulokset antava approksimaatio riippuu luonnollisesti käytössä olevasta opetusmenetelmästä ja sen parametreista, sovelletaan vertailussa kappaleen 6.6 kokeissa valittua hyvää ja yksinkertaista opetusmenetelmää. Opetuksessa tiheysfunktion alkuarvot määrätään itseorganisoidun piirrekartan perusteella ja varsinainen parametrien estimointi Baum-Welch –algoritmilla. Taulukossa (6.4) on esitetty tulokset, kun normaalijakaumien keskikohdat on initialisoinnissa järjestetty neliönmuotoisiksi kartoiksi.

Taulukon (6.4) ensimmäisessä sarakkeessa on kovarianssimatriisin tyyppi, toisessa kartan koko ja kolmannessa estimoitavien tilakohtaisten tiheysfunktioiden kuvaamiseen tarvittava kovarianssi- ja keskiarvoparametrien määrä käytettäessä 21-dimensioisia piirrevektoreita. Tiloja kokeissa on määritelty 88 eli 22 nelitilaista mallia. Prosenttiluvut ovat yhden henkilön neljän eri puhekerran keskiarvoja.

Parhaan tunnistustuloksen tuottaa 25 jakauman yhdistelmä, jossa kovarianssimatriisi on jätetty yksikkömatriisiksi. Diagonaalista kovarianssimatriisia käytettäessä paras tulos saatiin 16 jakauman yhdistelmällä. Täydellisillä kovarianssimatriiseilla varustetuilla normaalijakaumilla jo 9 jakaumaa tuotti hyvän tuloksen, mutta todennäköisyyksien laskemiseksi käytetty ohjelma kävi tällöin suhteettoman raskaaksi verrattuna muihin tiheysfunktioapproksimaatioihin.

kovarianssimatriisi	kartan koko	parametrimäärä	oikein%	virhe%
erillinen,täysi	2x2	21x11x4 + 21x4 = 1008	90.2	18.4
erillinen,diag.	3x3	21x9 + 21x9 = 378	86.8	14.5
erillinen,diag.	4x4	21x16 + 21x16 = 672	85.7	15.1
erillinen,diag.	5x5	21x25 + 21x25 = 1050	87.0	13.8
täysi	1	21x11 + 21 = 252	88.3	24.0
yhteinen,täysi	2x2	21x11 + 21x4 = 315	94.1	13.3
yhteinen,täysi	3x3	21x11 + 21x9 = 420	95.5	7.6
yhteinen,diag.	3x3	21 + 21x9 = 210	96.0	9.1
yhteinen,diag.	4x4	21 + 21x16 = 357	96.2	7.1
yhteinen,diag.	5x5	21 + 21x25 = 546	95.1	7.1
yhteinen,diag.	6x6	21 + 21x36 = 777	94.2	7.4
yksikkömatriisi	3x3	21x9 = 189	96.0	6.3
yksikkömatriisi	4x4	21x16 = 336	96.2	6.0
yksikkömatriisi	5x5	21x25 = 525	96.2	5.6
yksikkömatriisi	6x6	21x36 = 756	96.1	5.7
yksikkömatriisi	7x7	21x49 = 1029	96.1	5.6

Taulukko 6.4: *Erilaisten tiheysfunktioapproksimaatioiden vertailu. Yhteinen kovarianssimatriisi tarkoittaa, että tilan tiheysfunktiota approksimoivat osajakaumat käyttävät samaa kovarianssimatriisia ja eroavat toisistaan keskikohtansa perusteella. Tunnistustarkkuudet ovat kaikki neljän kokeen keskiarvoja paitsi diagonaalisten kovarianssimatriisien kohdalla, jossa keskiarvot on laskettu kolmesta kokeesta.*

6.5 Vertailua vektorikvantisointiin

6.5.1 Koodikirjavektorit ja normaalijakaumat

Vektorikvantisoinnissa moniulotteisen havaintovektorin edustajaksi valitaan yleensä lähimpänä oleva koodikirjavektori. Vektoriavaruus tulee täten jaetuksi erillisiin viipaleisiin, jotka koostuvat alueista kunkin koodikirjavektorin ympärillä. Kun havaintovektori sattuu lähelle viipaleiden välistä rajapintaa, lähin koodikirjavektori ei välttämättä kuvaa havaintovektoria kovin hyvin. Avaruuden paloittaisuuden pehmentämiseksi ja kvantisointivirheen pienentämiseksi koodikirja mallitetaan tässä työssä joukoksi moniulotteisia normaalijakaumia, joiden jakaumat ovat limittäisiä. Tällöin siis avaruuden diskreetti jako erikokoisiin ja -muotoisiin paloihin korvautuu jatkuvalla jakaumalla eri koodikirjavektoreiden todennäköisyyksiä.

Normaalijakaumamallissa havaitun piirrevektorin todennäköisyys on laskettavissa kullekin koodikirjajakaumalle erikseen (kaava 6.1, 6.8 tai 6.10). Tavallisessa vektorikvantisoinnissa sen sijaan lähintä koodikirjavektoria vastaavan vektoriluokan todennäköisyydeksi asetetaan 1 ja muiden luokkien todennäköisyyksiksi 0. Vektoriluokka voi kuitenkin yhden symbolin sijasta vastata jotain diskreettiä symbolien todennäköisyysjakaumaa, mikäli luokan tunnukseksi on valittu enemmistöäänestyksen (kappale 5.3.1) voittaneen tunnuksen sijasta tunnusten äänijakauma [Laaksonen 1990]. Kun vektorikvantisointia on käytetty diskreettien HMM:ien kanssa [Torkkola et al. 1991a], jokaisella tilalla on oma erilainen ulostulosymbolien todennäköisyysjakauma, jonka symbolit vastaavat luokiksi ryhmiteltyjä koodikirjavektoreita.

6.5.2 Kvantisointivirhe ja havaintotodennäköisyys

Kun piirrevektori x korvataan lähimmällä koodikirjavektorilla m, syntyy vektorien välistä etäisyyttä vastaava kvantisointivirhe $\|x - m\|$. Tapahtumaa voidaan myös tarkastella toisesta suunnasta: prosessi yrittää kuvata jotain tiettyä koodikirjavektoria ja tuottaa piirrevektorin x. Mitä suurempi on x:n etäisyys koodikirjavektorista m sitä pienempi on todennäköisyys, että x:llä kuvattiin juuri m:ää. Todennäköisyys voidaan siis esittää kvantisointivirheen monotonisesti laskevana funktiona (esim. kaavan 6.10 jälkimmäinen muoto, kun β-parametrit ovat samoja kaikilla koodikirjan vektoreilla).

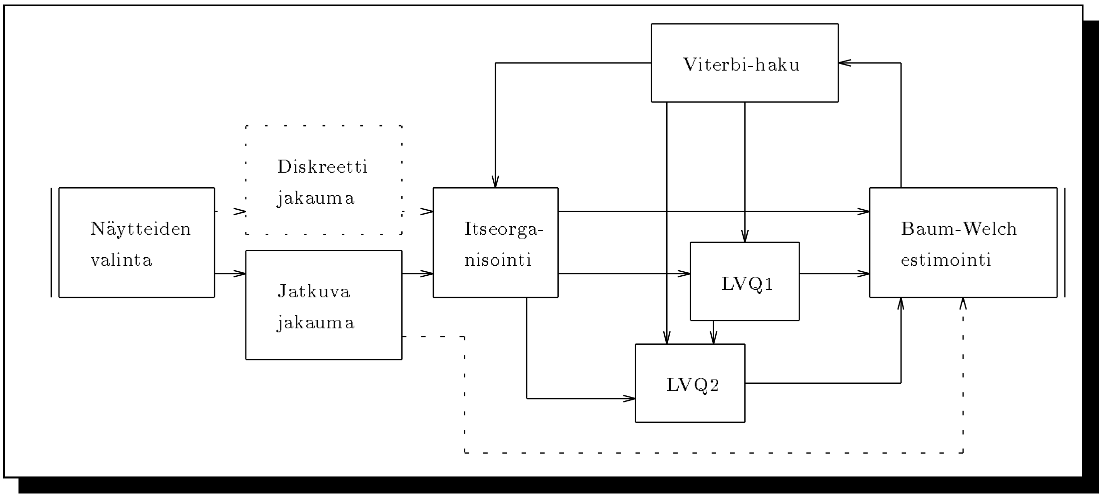
Haluttaessa laskea todennäköisyys, että havaitun piirrevektorin x on tuottanut tila i, jota kuvataan M koodikirjavektorilla, muodostetaan yksittäisten todennäköisyyksien painotettu summa (kaava 6.6). Painokerroin c_{im} vastaa todennäköisyyttä, jolla tilaa i edustaa koodikirjavektori m. Kaava 6.6 on itseasiassa eräs muoto Bayesin kokonaistodennäköisyyden kaavasta.

$$P(O_t = x|q_t = i) = \sum_{m=1}^{M} P(O_t = x|q_t = i, c_{im})P(c_{im}|q_t = i) \tag{6.11}$$

6.5.3 Koodikirjan osittaminen

Oppivassa vektorikvantisoinnissa [Kohonen 1990] käytetään yhtä suurta kvantisointikoodikirjaa kuvaamaan koko puheen piirrevektoriavaruutta, jonka eri foneemeihin liittyvät piirrevektorit jakavat luokkiin. Tätä kvantisointimallia voidaan käyttää myös kuvattaessa koodikirjavektoreita moniulotteisina normaalijakaumina [Huang et al. 1989a]. Tässä työssä kunkin tilan piirrevektoriavaruus, joka mallittaa foneemin tuottamisessa käytettävää ääniväylän muotoa, kuvataan erikseen pienellä kvantisointikoodikirjalla. Lähtökohta on siis sama kuin yleensä käytettäessä useiden normaalijakaumien yhdistelmiä jatkuvan jakauman HMM:ien ulostulojakaumina foneemien tunnistuksessa (esimerkiksi viitteissä [Rabiner 1989], [Nakagawa et al. 1990] ja [Kurimo 1991]) eli jokaisen Markov-mallin tilan tuottamien piirrevektoreiden todennäköisyysjakauma koostuu erilaisten normaalijakaumien yhdistelmästä. Kutakin tilaa vastaava tiheysfunktio (koodikirja) opetetaan siten, että se mahdollisimman hyvin vastaisi juuri kyseisen tilan piirrevektoreita. Testattava piirrevektori sopii näin suurimmalla todennäköisyydellä sen tilan tiheysfunktioon, josta vektori on peräisin.

6.6 Mallin parametrien opettaminen



Kuva 6.5: *Erilaisia vaihtoehtoja kätkettyjen Markov-mallien opetuksessa. Katkoviivoitettu osuus ei ole tässä työssä toteutettua.*

6.6.1 Siirtotodennäköisyyksien alkuarvot

Tilojen välisten siirtymien todennäköisyyksien alkuarvoksi foneemin mallissa kelpaa

$$a_{ij} = \begin{cases} N/T & \text{jos } j-i=1 \\ 1-N/T & \text{jos } j-i=0 \\ 0 & \text{muuten} \end{cases} \tag{6.12}$$

T on foneemin keskimääräinen pituus, N mallin tilojen määrä ja T/N siis keskimääräinen yhdessä tilassa vietetty aika. Kun kiellettyjen tilasiirtymien todennäköisyydet alustetaan nolliksi, kiellettyjen tilajonojen esiintymistodennäköisyys pysyy nollana.

Tilojen esiintyminen foneemien sisällä on hyvin foneemikohtaista; joissakin foneemeissa yksi tai kaksi tilaa voivat osoittaustua selvästi hallitseviksi, jolloin niissä tiloissa silmukkatodennäköisyys on suuri ja muissa pieni. Toisissa foneemeissa tilojen läpikäyntiajat vaihtelevat vain vähän tai ovat suhteessa koko foneemin kestoon, mutta saattaa käydä niinkin, että tilojen esiintyminen riippuu foneemin kestosta tai naapurifoneemeista. Nämä mahdollisuudet huomioonottaen edellä esitetty yleisluontoinen alustus (kaava 6.12) on sopiva.

6.6.2 Jakaumien itseorganisointi

Havaintojakaumien approksimaatioiden parametrien μ_{im}, Σ_{im} ja c_{im} iteratiivisen Baum-Welch –estimoinnin onnistumisen kannalta on välttämätöntä, että parametreilla on hyvät alkuarvot. Alkuarvojen merkitys on sitä tärkeämpi mitä useampia approksimoivia jakaumia käytetään yhtä tilaa kohti, sillä parametrien määrän lisääntyminen vaikeuttaa oleellisesti estimointia.

Jos tilan tuottamia piirrevektoreita approksimoidaan vain yhdellä moniulotteisella normaalijakaumalla, alkuarvot N-tilaisen mallin jakauman parametreille saadaan jakamalla kukin opetusjono N:ään yhtäpitkään osaan ja laskemalla kunkin osan havaintovektoreiden keskiarvot ja varianssit. Kun foneemin tilan tuottamien piirrevektorien jakaumaa approksimoidaan usean moniulotteisen normaalijakauman yhdistelmällä, ei hyvien alkuarvojen määrääminen jakaumille ole yhtä yksinkertaista. Mikäli jakaumat vain sijoitetaan jonnekin K-ulotteiseen piirreavaruuteen, ei Baum-Welch –estimointi kykene tehokkaasti sovittamaan useita approksimoivia jakaumia niin, että yhdistelmän yhteinen approksimointikyky olisi paras mahdollinen. Tähän sovitustyöhön kokeillaan algoritmia, joka ensin sijoittaa normaalijakaumien keskikohdat moniulotteiseen vektoriavaruuteen kuvaamaan mahdollisimman hyvin tuntemattoman jakauman tihentymiä ja sitten vasta estimoi jakaumien painokertoimet ja kovarianssit. Tässä jälkimmäisessä vaiheessa voidaan käyttää Baum-Welch –estimointia ja siirrellä tarvittaessa myös jakaumien keskikohtia.

Jakaumien optimaalinen sijoittelu on hankala optimointitehtävä, sillä havaintovektorijakaumat ovat usein monimutkaisia ja virheen minimoinnissa törmätään väistämättä lukuisiin paikallisiin minimeihin, koska opetukseen käytetään tyypillisesti suurta datamäärää. Lisävaikeutena on myös se, että varmuudella ei voida edes määrätä jokaisen opetukseen varatun havaintovektorin tuottanutta foneemia ja sen tilaa.

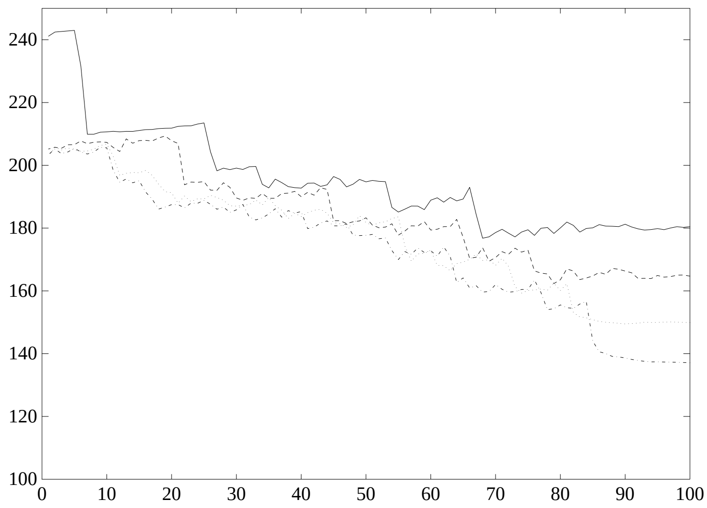
Tässä työssä on jakaumien sijoittelu tehty professori Kohosen kehittämän itseorganisoituvan piirrekartan [Kohonen 1990] avulla. Jakaumien keskikohtia vastaavat kartan yksiköt siroitellaan ensin satunnaisesti pitkin vektoriavaruutta ja sitten siirrellään yksiköitä satunnaisesti valittujen näytteiden perusteella vähitellen kohti opetukseen valittuja piirrevektoreita. Menetelmä esitettiin tarkemmin kappaleessa 5.2.

Opetusnäytteiden joukko pyritään valitsemaan mahdollisimman laajaksi ja mallitettavan foneemin tilan kaikkia erilaisia havaintovektoreita hyvin edustavaksi.

Itseorganisaatiossa yritetään säilyttää K-ulotteisen piirrevektorijakauman topologia mahdollisimman hyvin siirtämällä myös valittua näytevektoria lähinnä olevan karttayksikön naapureita hieman kohti näytevektoria.

Piirrevektorien monidimensioisuuden vuoksi itseorganisointia on hankala seurata ja siksi sopivien opetukseen liittyvien parametrien valinta ei ole yksinkertaista. Optimoimalla parametreja vuorotellen ei välttämättä päästä parhaaseen parametriyhdistelmään, koska kartan koko, opetuskierrosten määrä, opetettavan naapuruston koko ja opetusvakion suuruus ovat voimakkaasti sidoksissa toisiinsa. Itseorganisoimalla saadun jakauma-approksimaation hyvyyttä voidaan arvioida tutkimalla keskimääräisen kuvausvirheen suuruutta eli näytteen etäisyyttä lähimmästä kartan yksiköstä.

Kuva 6.6 esittää erään tilamallin kuvausvirheen kehitystä käytettäessä eri kokoisia karttoja. Tilamallin tuottamista piirrevektoreista on poimittu koko opetusaineistosta 450 näytettä (tai niin paljon kuin on, jos täyttä määrää ei löydy), joista kukin sisältää muutaman peräkkäisen vektorin. Yksi opetuskierros vastaa kunkin tilamallin opettamista kaikilla sitä vastaavilla havainnoilla. Sadan opetuskierroksen aikana opetusvakiota (kaava 5.7) pienennetään tasaisesti 0.2:sta 0.05:een ja opetettavien naapurien maksimietäisyyttä kartan leveydestä nollaan. Kuvasta on havaittavissa, että kuvausvirhe jää iteroinnin



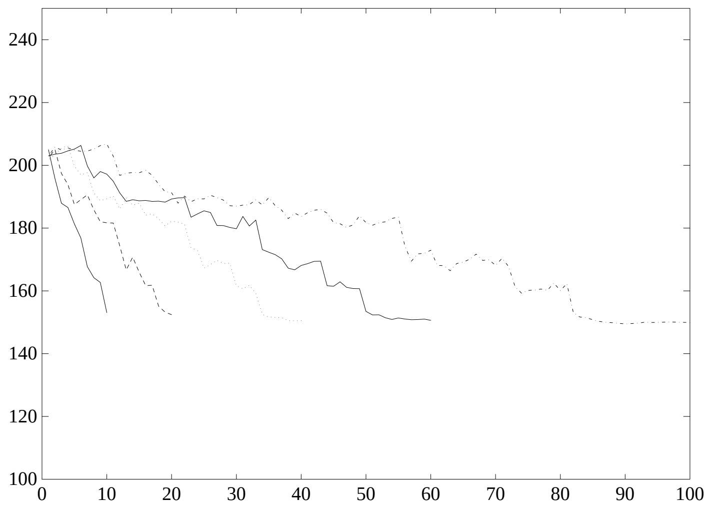
Kuva 6.6: *Esimerkkejä kuvausvirheen (pystyakseli) pienenemisestä itseorganisoinnissa. Yhtenäinen viiva, katkoviiva, pistejono ja pisteviiva kuvaavat erään tilamallin kuvausvirhettä 3x3, 4x4, 5x5 ja 6x6 kartalla.*

loppuvaiheessa sitä pienemmäksi mitä suurempaa karttaa käytetään.

Kuvassa 6.7 organisointi suoritetaan eri nopeuksilla niin, että parametrien alku- ja loppuarvot pysyvät samoina, mutta nopeammassa organisoinnissa parametriarvojen pienentäminen tapahtuu suuremmin askelin. Kartan koko on kaikissa tapauksissa sama 5x5 yksikköä. Nopeammassa organisoinnissa kuvausvirhe pienenee nopeammin, mutta jää kuitenkin samalle tasolle kuin hitaammassa organisoinnissa.

Yksittäisten opetustapahtumien merkitystä kuvaavan opetusvakion α (käsitelty yleisemmin kappaleessa 5.2.3) pienentämisen vaikutusta itseorganisoitumiseen on tutkittu kuvassa 6.8. Suuri opetusvakion alkuarvo johtaa kuvausvirheen nopeaan pienenemiseen opetuksen alkuvaiheessa, mutta kuvausvirheen lopullinen taso ei kuitenkaan merkittävästi muutu. Kun alkuarvo pidetään samana ja kasvatetaan loppuarvoa, opetusvakion arvot ovat keskimäärin suurempia kuin pienemmällä loppuarvolla, mikä näkyy iteroinnin keskivaiheessa pienempänä kuvausvirheenä. Mikäli loppuarvo jää liian suureksi, kartta ei saavuta stabiilia tilaa opetuksen loppuvaiheessa vaan saattaa jäädä värähtelemään.

Jakaumien itseorganisointi on kuitenkin vain yksi vaihe Markov-mallien opetuksessa ja itseorganisoinnilla saavutettu pieni kuvausvirhe vain osatavoite. Vaikka useilla eri parametrivalinnoilla saavutetaan lähes yhtä pieni kuvausvirhe, jakauma-approksimaatiot ovat kuitenkin erilaisia. Onhan eri tavoin opetettujen karttojen päätyminen samallaiseen ratkaisuun varsin epätodennäköistä. Puheentunnistuksessa tavoitteen kannalta luotettavaa vertailua voidaan tehdä vasta vertailemalla riippumattomasta aineistosta saatuja tunnistustuloksia, kun Markov-mallit on kokonaisuudessaan opettu. Itseorganisoinnin vaikutus tunnistustuloksiin on esitetty taulukossa 6.5. Tulokset ovat keskiarvoja neljästä eri opetus- ja testiaineistokombinaatiosta. Markov-malleille on itseorganisointivaiheen jälkeen opetet-



Kuva 6.7: *Esimerkkejä kuvausvirheen (pystyakseli) pienenemisestä itseorganisoinnissa, kun oppimisvakiota ja opetettavan naapuruston kokoa pienennetään eri nopeuksilla. Samoja alku- ja loppuarvoja on käytetty opetettaessa aineisto 10, 20, 40, 60 ja 100 kertaa.*

tu iteratiivisella Baum-Welch –algoritmilla tilojen transitiotodennäköisyydet ja jakaumien sopivat painokertoimet. Myös jakaumien huippujen paikkoja voidaan tarpeen vaatiessa muuttaa jatko-opetuksessa.

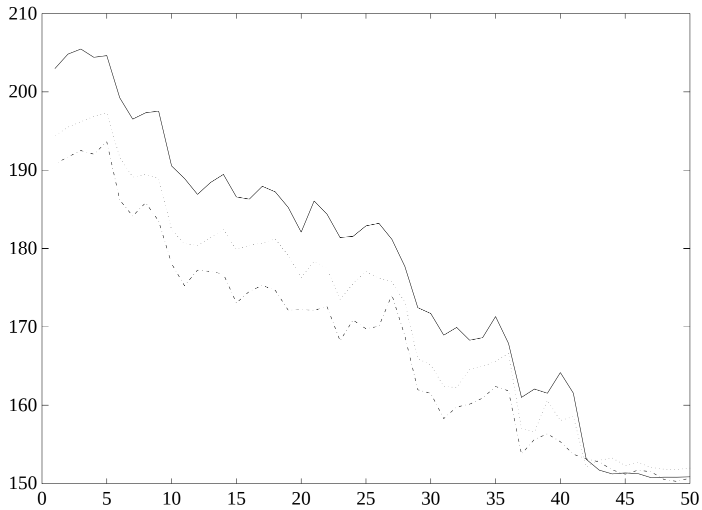
Tulosten perusteella opetuksen alkuvaiheessa erot eri tavoin itseorganisoitujen jakaumien antamissa tunnistustarkkuuksissa ovat pieniä, mutta kasvavat opetuksen määrää lisättäessä. Johtopäätös tuloksista on, että hitaammin itseorganisoitu jakauma tarjoaa ilmeisesti paremmat mahdollisuudet mallien tarkentamiseen. Erot ovat kuitenkin pieniä. 50 opetuskierrosta näyttää olevan riittävän hidas itseorganisointi parhaan mahdollisen alkuasetelman luomiseksi.

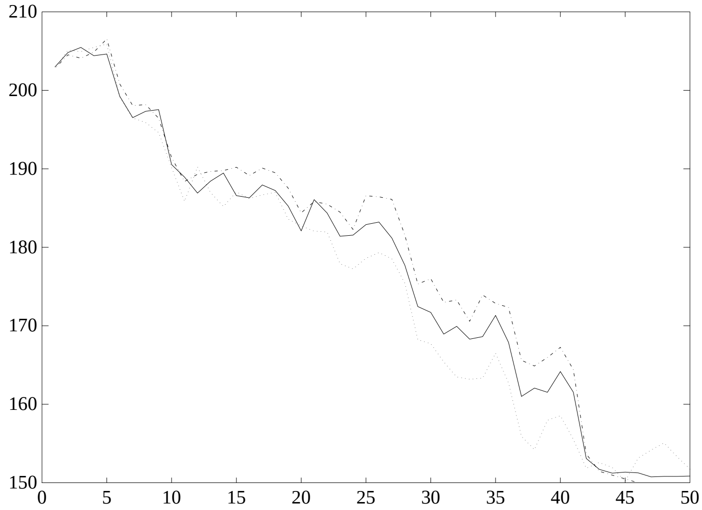
6.6.3 Erottelukyvyn parantaminen LVQ:n avulla

Kun itseorganisoituvien karttojen avulla on sijoiteltu jakaumien keskikohdat sopiviin paikkoihin, voidaan paikkoja vielä hieman muutella paremman tilojenvälisen erottelukyvyn saavuttamiseksi.

LVQ eli oppiva vektorikvantisointi (kappale 5.3.1) tähtää vektorien luokittelun parantamiseen niin, että eri luokkiin kuuluvat vektorit saataisiin mahdollisimman hyvin eroteltua toisistaan.

LVQ1 (kappale 5.3.2) on opetusmenetelmä, jossa opetusvektoria lähinnä olevaa karttayksikköä siirretään hieman poispäin opetusvektorista, mikäli karttayksikön edustama luokka on väärä, ja hieman lähemmäksi, mikäli luokka on oikea. Kaikki opetusvektorit käydään läpi ja opetetaan vastaavia karttayksiköitä. Opetusta jatketaan sovittu kierrosmäärä tai





Kuva 6.8: *Esimerkkejä kuvausvirheen (pystyakseli) pienenemisestä itseorganisoitaessa erilaisilla opetusvakion arvoilla. Itseorganisointi on tehty opettamalla 5x5 kartalle eräs tilamalli pienentäen α:aa tasaisesti arvosta 0.4 (-.), 0.3 (.) ja 0.2 (-) arvoon 0.05 (ylempi) ja arvosta 0.2 arvoon 0.1 (.), 0.05 (-) ja 0.01 (-.) (alempi kuva).*

	2 kertaa BW		5 kertaa BW		8 kertaa BW	
A-kierrosmäärä	oikein%	virhe%	oikein%	virhe%	oikein%	virhe%
0	95.5	8.0	96.0	6.1	96.1	5.9
5	95.9	6.4	96.1	6.1	96.1	5.9
10	95.9	6.4	95.9	6.1	96.1	5.8
50	95.8	6.3	96.1	5.8	96.2	5.6
100	95.9	6.2	96.0	6.0	96.2	5.7

Taulukko 6.5: *Tunnistustarkkuuden muuttuminen itseorganisointikierrosten (A) lisääntyessä. Yksi kierros tarkoittaa, että kaikki näytteet käydään kertaalleen läpi. Opetusvakiota on kullakin kierrosmäärävaihtoehdolla vähennetty tasaisesti 0.2:sta 0.05:een. Kartan koko on 5x5 ja naapuruston sädettä pienennetään tasaisesti 5:stä 0:aan. Itseorganisoinnin jälkeen Markov-mallit on opettu Baum-Welch –algoritmilla (BW) 2, 5 ja 8 kertaa.*

kunnes virheiden määrä on riittävän pieni.

LVQ2 (kappale 5.3.3) poikkeaa LVQ1:stä siten, että lähintä karttayksikköä opetetaan vain, kun sen edustama luokka on väärä. Tällöin haetaan myös lähinnä oleva oikeaa luokkaa edustava yksikkö ja sitä siirretään lähemmäksi opetusvektoria. Opetus voidaan rajoittaa vain sellaisiin tapauksiin, joissa opetusvektori sijaitsee lähes yhtä kaukana lähimmistä oikeaa ja väärää luokkaa edustavista yksiköistä.

Luokkien välisen erottelukyvyn parantaminen tuntuu aiheelliselta kätkettyjen Markov-mallien yhteydessä, sillä kun tilojen ulostulomalleja viritetään kuvaamaan mahdollisimman hyvin signaalin eri vaiheita, on odotettavissa, että monilla tilamalleilla voidaan kuvata hyvin myös muiden tilojen tuottamaa signaalia. Tällöin tilojen väliset erot jäävät pieniksi ja oikean tilan löytäminen vaikeutuu.

Foneemien tunnistuksen onnistumiselle on ratkaisevampaa oikean tilan ja sopivimman virheellisen tilan todennäköisyyksien välinen ero kuin itse todennäköisyyksien arvot.

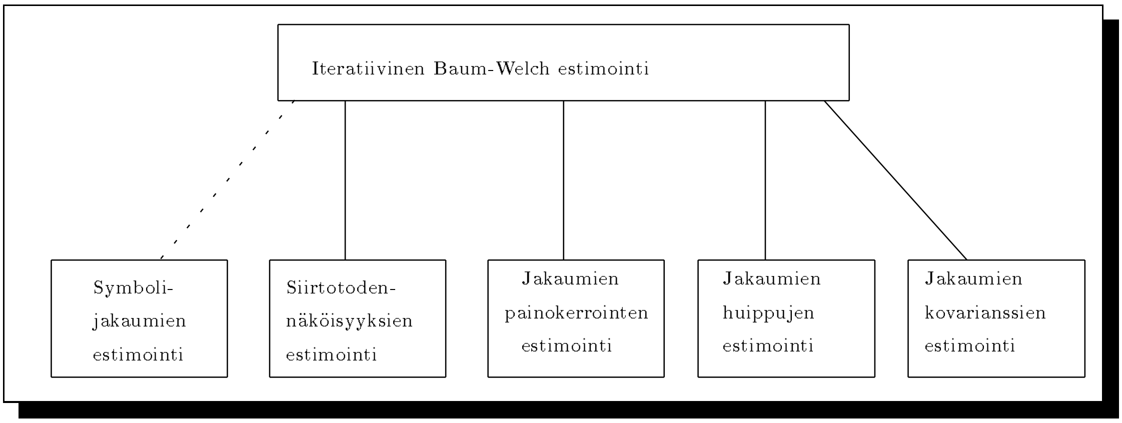
LVQ-opetusmenetelmiä testattiin hakemalla kaikkien tilojen koodikirjavektoreiden joukosta kutakin opetusvektoria lähimpänä oleva. Foneemimallia, jonka tilan tiheysfunktioapproksimaatioon tämän lähin vektori kuului, verrattiin opetusvektorin edustamaan foneemiin ja opetus tehtiin kaavojen 5.10 ja 5.11 mukaan. Yksi opetuskierros sisältää kaikkien opetukseen varattujen puheaineistojen piirrevektoreiden läpikäymisen. Lopuksi kokeiltiin vielä yhdistelmäopetusta, jossa ensin ajettiin kierros LVQ1:tä ja sen jälkeen kierros LVQ2:ta.

Tulokset (taulukko 6.6) osoittavat LVQ:n käytön johtavan pieniin parannuksiin keskimääräisessä tunnistustarkkuudessa. Parhaaksi osoittautui pienellä opetusvakion arvolla annettu LVQ2-opetus.

6.6.4 Parametrien uudelleenestimointi

Opetus	alku	loppu	oikein%	virhe%
			96.0	6.0
2 kertaa LVQ1	0.2	0.05	96.2	6.0
2 kertaa LVQ1	0.05	0.01	96.1	5.7
5 kertaa LVQ1	0.02	0.001	96.0	5.9
2 kertaa LVQ2	0.2	0.05	96.2	5.7
2 kertaa LVQ2	0.05	0.01	96.2	5.5
5 kertaa LVQ2	0.02	0.001	96.2	5.5
LVQ1 ja LVQ2	0.2	0.01	96.3	5.6

Taulukko 6.6: *LVQ-opetuksen avulla saavutettuja tunnistustuloksia. Ylimmällä rivillä tunnistustarkkuus ilman LVQ-opetusta. Yksi opetuskierros tarkoittaa kaikkien opetusvektorien läpikäymistä. Sarakkeissa "alku" ja "loppu" on opetusvakion alku- ja loppuarvo. Tilamallien jakauma-approksimaatiot on alustettu itseorganisoimalla ja LVQ-opetuksen jälkeen mallit opetetaan vielä Baum-Welch –algoritmilla.*



Kuva 6.9: *Iteratiivisen Baum-Welch –estimoinnin sovellukset. Katkoviivoitettu osuus liittyy vain diskreettien HMM:ien opetukseen.*

Tilojenvälisten siirtymien todennäköisyydet eli transitiotodennäköisyydet a_{ij} estimoidaan käyttäen kappaleessa 6.6.4 esitettyä iteratiivista Baum-Welch –algoritmia.

Koska tilojen tiheysfunktioden estimoinnissa käytetyt näytteet on määrätty hieman keinotekoisesti jakamalla kukin foneeminäyte tasamittaisiin osiin, voidaan parametrien estimaatteja parannella iteratiivisesti, kun edellisten estimaattien avulla jaetaan näytteet uudestaan tiloihin. Näinollen parametrit c_{im}, μ_{im} ja Σ_{im} uudelleenestimoidaan kuten tilojenväliset siirtotodennäköisyydetkin iteratiivisella Baum-Welch –algoritmilla.

Lasketaan ensin opetusmateriaalista $\gamma_t(i,m)$:t eli todennäköisyydet sille, että systeemi on hetkellä t tilassa i jakauman m mukaan ottaen huomioon havainnot ja mallien parametrien alkuarvot. Soveltamalla kaavaa 4.14 jokaiselle tilan i jakaumalle m erikseen saadaan

$$\gamma_t(i,m) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \frac{c_{im}b_{im}(O_t)}{b_i(O_t)} \tag{6.13}$$

	vain siirtotn:t		myös painokertoimet		myös jakaumahuiput	
Kierrosmäärä	oikein%	virhe%	oikein%	virhe%	oikein%	virhe%
0	95.1	7.3	95.1	7.3	95.1	7.3
1	95.4	7.4	95.7	7.1	95.8	6.5
2	95.6	7.3	95.7	6.8	95.8	6.3
4	95.6	7.4	95.8	6.8	95.9	6.0
6	95.6	7.4	95.8	6.8	96.2	5.7
8	95.6	7.4	95.7	6.8	96.2	5.6
10	95.5	7.5	95.8	6.7	96.2	5.7
12	95.5	7.5	95.8	6.7	96.2	5.7
14	95.5	7.5	95.8	6.7	96.1	5.7

Taulukko 6.7: *Opetuksen määrän vaikutus tunnistustuloksiin iteratiivisen Baum-Welch–algoritmin eri sovelluksissa. Vasemmalla tiheysfunktioapproksimaatioita ei muuteta lainkaan, keskellä uudelleenestimoidaan jakaumien painokertoimet ja oikealla uudelleenestimoidaan edellisten lisäksi myös jakaumien huiput. Tiheysfunktioapproksimaatiossa piirrekomponenttien väliset kovarianssimatriisit pidetään yksikkömatriiseina.*

Havaintojen $\gamma_t(i,m)$:llä painotetusta keskiarvosta saadaan μ_{im} ja Σ_{im}:lle suurimman uskottavuuden estimaatit [Rabiner 1989].

$$\hat{\mu}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m) O_t}{\sum_{t=1}^{T} \gamma_t(i,m)} \tag{6.14}$$

$$\hat{\Sigma}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m)(O_t - \mu_{im})(O_t - \mu_{im})^T}{\sum_{t=1}^{T} \gamma_t(i,m)} \tag{6.15}$$

Vastaavasti estimoidaan uudet jakaumien painokertoimet $\hat{c}_{im}$

$$\hat{c}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m)}{\sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t(i,m)} \tag{6.16}$$

Sijoittamalla estimaatit $\hat{\mu}_{im}$ ja $\hat{\Sigma}_{im}$ kaavaan 6.1 ja $\hat{c}_{im}$:t kaavaan 6.6 saadaan laskettua uudet arvot havaintotodennöisyyksille $b_{im}(O_t)$ ja $b_i(O_t)$. Näistä saadaan edelleen (kaavat 4.9, 4.11, 4.12 ja 6.13) uudet hetkelliset tilatodennäköisyydet eri jakaumien $m = 1, \cdots, M$ perusteella $\gamma_t(i,m)$, ja (kaavoista 4.15, 6.14, 6.15 ja 6.16) parannetut estimaatit parametreille. Iterointia toistetaan ennalta sovittu kierrosmäärä tai kunnes estimaattien muutos alittaa ennalta sovitun rajan.

Edellä kuvattu parametrien estimointi tähtää sellaisten parametrien saamiseen, jotka maksimoisivat opetukseen käytetyn jonon todennäköisyyden. Käytännössä estimointiin käytetään kuitenkin useita erilaisia opetusjonoja, ja halutaan, että niiden kaikkien todennäköisyydet maksimoituisivat. Estimointikaavoihin 4.15, 6.14, 6.15 ja 6.16 on siis lisättävä summaus yli kaikkien opetusjonojen (L kpl).

$$\hat{\mu}_{im}^L = \frac{\sum_{l=1}^{L} \sum_{t=1}^{T_l} \gamma_t^l(i,m) O_t^l}{\sum_{l=1}^{L} \sum_{t=1}^{T_l} \gamma_t^l(i,m)} \tag{6.17}$$

	ilman kovariansseja		kovarianssit	
Kierrosmäärä	oikein%	virhe%	oikein%	virhe%
0	85.6	17.8	85.6	17.8
1	90.1	14.6	93.8	14.8
2	90.6	13.5	95.2	9.0
4	90.6	13.8	95.2	8.8
6	90.6	14.0	95.2	8.3
8	90.7	14.1	95.3	7.8
10	90.6	14.1	95.2	7.7
12	90.7	14.0	95.1	7.6
14	90.7	13.9	95.0	7.6

Taulukko 6.8: *Baum-Welch –opetuksen määrän vaikutus tunnistustuloksiin, kun tiheysfunktioapproksimaatiossa on mukana diagonaaliset kovarianssimatriisit. Vasemmalla uudelleenestimoidaan vain jakaumien painokertoimet ja huiput ja oikealla myös kovarianssimatriisit.*

$$\hat{\Sigma}^L_{im} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\gamma^l_t(i,m)(O^l_t-\mu_{im})(O^l_t-\mu_{im})^T}{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\gamma^l_t(i,m)} \tag{6.18}$$

$$\hat{c}^L_{im} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\gamma^l_t(i,m)}{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\sum_{m=1}^{M}\gamma^l_t(i,m)} \tag{6.19}$$

$$\hat{a}^L_{ij} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\xi^l_t(i,j)}{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\gamma^l_t(i)} \tag{6.20}$$

Taulukoissa 6.7 ja 6.8 on tutkittu opetuskierrosten määrän vaikutusta tunnistustarkkuuteen. Ennen Baum-Welch –opetusta Markov-mallien tilojen siirtotodennäköisyydet on alustettu kaavan 6.12 mukaan ja jakaumien tiheysfunktiot approksimoitu itseorganisoimalla huippukohtien paikat. Jakauman huippuja vastaavien normaalijakaumien painokertoimien alkuarvoina on kutakin huippua lähinnä olevien opetusvektoreiden osuus koko opetusmateriaalista. Kovarianssimatriisien alkuarvoja laskettaessa käytetään niiden opetusvektorien komponenttien odotusarvoja, joille jakauman huippu on lähin huippu μ_{ic}.

$$\bar{\Sigma}^L_{im} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\delta(\mu_{im},\mu_{ic})(O^l_t-\mu_{im})(O^l_t-\mu_{im})^T}{\sum_{l=1}^{L}\sum_{t=1}^{T_l}\delta(\mu_{im},\mu_{ic})} \tag{6.21}$$

$$\delta(\mu_{im},\mu_{ic}) = \begin{cases} 1 & \text{jos } \mu_{im}=\mu_{ic} \\ 0 & \text{muuten} \end{cases} \tag{6.22}$$

Jos kussakin tilassa käytetään vain yhtä kovarianssimatriisia lasketaan sen alkuarvo komponenttien painottamattomana odotusarvona kutakin opetusvektoria lähimmästä huipusta μ_{ic}.

$$\bar{\Sigma}^L_{i} = \frac{\sum_{l=1}^{L}\sum_{t=1}^{T_l}(O^l_t-\mu_{ic})(O^l_t-\mu_{ic})^T}{\sum_{l=1}^{L}T_l} \tag{6.23}$$

Kun tiheysfunktiot on approksimoitu jakaumahuippujen avulla (taulukko 6.7), kannattaa normaalijakaumien painokerrointen ja tilojen siirtotodennäköisyyksien lisäksi antaa myös jakaumahuippujen siirtyä tarvittaessa muiden parametrien muuttuessa. Kovarianssimatriisien Baum-Welch –estimointi on välttämätöntä, mikäli kovarianssiparametreja halutaan käyttää tiheysfunktion approksimoinnissa (taulukko 6.8). Kahdeksan opetuskierrosta näyttää yleisesti olevan riittävä opetusmäärä.

Äärellisen opetusaineiston käytön johdosta on käytännössä hyvin mahdollista, että jonkin osajakauman painokerroin tai kovarianssiparametri lähenee opetuksessa nollaa, koska sitä vastaava harvinainen äänteenosa on opetusaineistossa heikosti edustettuna. Tämän välttämiseksi painokertoimille c_{im} ja kovarianssimatriisien Σ_{im} komponenteille on määrätty alaraja (tässä 10^{-4}), jota pienemmiksi niitä ei päästetä.

6.6.5 Foneemitransitioiden estimointi

	A	D	E	H	I	J	K	L	M	N	O	P	R	S	T	U	V	Y	[	\	>	<
A	.	.	.	1	7	1	3	6	3	13	.	1	4	5	7	3	4	.	.	.	.	34
D	27	.	38	.	8	.	.	.	.	.	17	.	.	.	.	3	.	1	2	.	.	.
E	1	.	.	1	6	.	5	12	2	24	.	.	7	10	15	1	1	.	.	.	.	6
H	13	14	19	.	8	9	1	.	5	.	3	.	.	.	13	4	.	4	1	.	.	.
I	4	3	4	1	.	1	4	5	4	15	2	.	2	20	14	.	1	.	1	.	.	10
J	38	.	17	.	.	.	.	.	.	.	32	.	.	.	.	1	.	.	9	.	.	.
K	13	.	10	.	13	.	.	.	.	.	16	.	.	16	3	10	.	2	8	1	.	.
L	20	.	14	.	21	.	2	.	8	.	5	.	.	.	4	10	.	1	8	1	.	.
M	21	.	14	.	20	.	.	.	.	.	5	4	.	.	.	14	.	2	13	.	.	2
N	6	.	8	.	4	.	2	.	.	.	4	.	.	2	10	.	.	.	2	1	.	54
O	1	3	.	4	16	2	5	7	2	19	.	5	8	10	5	.	2	.	.	.	.	3
P	17	.	18	.	19	.	.	1	.	.	7	.	9	.	.	16	.	1	8	.	.	.
R	11	.	6	2	26	10	9	.	1	.	8	.	.	.	9	4	2	.	4	.	.	.
S	13	.	14	.	19	.	3	.	.	.	4	.	.	.	25	3	.	1	6	.	.	4
T	23	.	18	.	10	.	1	.	.	.	5	.	.	.	.	12	.	3	12	3	.	7
U	1	2	1	3	5	1	11	6	2	11	9	.	3	18	8	.	4	.	.	.	.	6
V	45	.	9	.	15	.	.	.	.	.	10	.	.	.	.	4	.	.	13	.	.	.
Y	.	2	.	5	1	1	12	9	4	5	.	3	1	10	22	.	4	.	1	5	.	7
[	.	.	.	2	3	.	2	5	3	20	.	.	5	4	5	.	3	6	.	.	.	35
\	3	.	.	.	11	9	1	3	3	14	.	.	.	23	6	.	.	5	4	.	.	10
>	3	.	6	2	1	7	13	5	6	2	10	6	1	11	9	.	6	2	1	.	.	.
<	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

Taulukko 6.9: *Kukin rivi sisältää rivin ensimmäisen foneemin jälkeisen seuraavan foneemin todennäköisyysjakauman. Luvut ovat prosenttien kokonaisosia ja alle 1%:n todennäköisyydet on merkitty pisteellä. [on Ä, \ Ö, > sanan alku ja < loppu.*

Todennäköisyydet, joilla tiettyä foneemia seuraa joku toinen tietty foneemi, voidaan luotettavimmin määrätä hankkimalla riittävän laaja tekstiaineisto ja laskemalla sieltä kutakin foneemia kohden seuraavan foneemin frekvenssit. Laskemalla myös sanan ensimmäisen ja viimeisen foneemin esiintymisfrekvenssit voidaan määrätä siirtymistodennäköisyydet alkuhiljaisuudesta seuraavaan foneemiin ja viimeisestä foneemista loppuhiljaisuuteen. Tuloksena saadaan esimerkiksi taulukon 6.9 kaltaiset todennäköisyydet.

6.7 Todennäköisimmän tilajonon käyttö opetuksessa

6.7.1 Todennäköisin havainnot tuottanut tilajono

```
K>><><<>PP<<MAAAAAAAAAAAAAA[AAEEEIIJJJJJIJIIII><>>>>>>TKKKKOAAAAAAAAAH<<<
2332321210013322223313233300113211123131311101030303301033331000111312000
>>>>>>>>>AAAAAAAAAAAAAAAAAAAIIIIIIIIIIIIIIIIKKKKKKKKKKKAAAAAAAAAAAA<<<<
0000122230000122222222222222300111111111222222301122222222300000001112300 00

<<<<<<<<>>K<<<<<K<<K
01011111133121002222
<<<<<<<<<<<<<<<<<<<<
01111111111111222223
```

Taulukko 6.10: *Todennäköisin tilojen välinen polku. Kukin kirjain kertoo käytössä olevan foneemimallin ja välittömästi sen alla oleva numero mistä tämän mallin tilasta on kyse. Ylempi polku on valittu olettamalla tilat toisistaan riippumattomiksi. Alemmassa on käytetty kätkettyjä Markov-malleja.*

Jos foneemin sisäisten tilojen välinen polku voidaan valita vapaasti ja tilat q_t ovat toisistaan riippumattomia ja yhtä todennäköisiä, signaalista saatujen havaintojen perusteella todennäköisin tilajono on suoraan

$$\max_Q \prod_{t=1}^{T} b_{q_t}(O_t) \tag{6.24}$$

jossa $Q = q_1, q_2, \cdots, q_T$ on tilojen välinen polku. Lisäksi oletetaan havainnot riippumattomiksi toisistaan.

Puhesignaalista lasketut peräkkäiset piirrevektorit poikkeavat kuitenkin satunnaisten häiriöiden vuoksi toisistaan sen verran, että tilojen riippumattomuus johtaa edestakaisin tilojen välillä poukkoilevaan polkuun, mikä ei vastaa todellista tilannetta. Todennnäköisin polku tilojen välillä ei siis välttämättä ole sama kuin peräkkäin asetetut hetkellisesti todennäköisimmät tilat. Kun foneemin sisäiset tilat esiintyvät järjestyksessä ja tilojen väliset siirtymiset noudattavat tilakohtaisia todennäköisyysjakaumia, todennäköisin polku kuvaa realistisemmin foneemin sisäistä rakennetta. Yksinkertaisuuden vuoksi voidaan tehdä Markov-oletus, eli että siirtymistodennäköisyys seuraavaan tilaan riippuu vain edellisestä tilasta. Tällöin todennäköisin tilajono on

$$\max_Q \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(O_t), \tag{6.25}$$

joka voidaan ratkaista tehokkaasti Viterbi-algoritmilla, kuten kappaleessa 4.4 esitettiin.

Puhesignaalissa oleelliset muutokset tapahtuvat havaintoväliin nähden verrattaen hitaasti. Lisäksi 10 millisekunnin välein muodostetut piirrevektorit lasketaan signaali-ikkunasta, jonka pituus on 20 millisekuntia. Näin peräkkäiset havaintovektorit eivät ole täysin riippumattomia toisistaan. Peräkkäisten havaintojen todennäköisyyksien kertominen keskenään ei anna tarkkaa todennäköisyyttä koko jonolle, mutta approksimaatio on kuitenkin useimmiten riittävän hyvä.

6.7.2 Todennäköisimmän tilasekvenssin avulla tehty LVQ

LVQ-opetuskaava	kierrosmäärä	oikein%	virhe%
	0	96.2	5.6
1	1	96.1	5.8
1	2	96.1	5.7
2	1	96.2	5.7
2	2	96.2	5.6
2	3	96.2	5.6

Taulukko 6.11: *Tunnistustarkkuudet toistamalla opetusprosessia, johon sisältyy todennäköisimmän polun haku Viterbi-algoritmilla, kierros LVQ-opetusta ja Baum-Welch –opetus. Ensimmäisellä rivillä on lähtöarvo, joka on saavutettu aiemmin esitellyllä alkuarvojen estimointimenetelmällä ja Baum-Welch –opetuksella. LVQ-opetuksessa opetusvakiota pienennetään 0.002:sta 0.001:een.*

Viterbi-algoritmin avulla saatua todennäköisintä tilajonoa voidaan vielä käyttää opetuksessa hyödyksi. Tarkastetaan, onko puhuttua sanaa vastaavassa tilasekvenssissä jokainen tila optimaalinen myös vektorikvantisoinnin mukaan. Mikäli jokin väärän koodikirjan vektori tuottaa pienemmän kvantisointivirheen kuin oikean tilan paras vektori, opetetaan koodikirjavektoreita esimerkiksi LVQ2-opetussäännön mukaan. LVQ-opetuksen jälkeen opetetaan malleja uudestaan Baum-Welch –algoritmilla kappaleessa 6.6.4 valitulla parhaalla tavalla (kahdeksan kierrosta). Opetusprosessia on toistettu iteratiivisesti muutama kierros taulukossa 6.11.

Todennäköisimmältä tilasekvenssiltä ei kuitenkaan voida vaatia, että se antaisi myös jokaisella erillisellä ajanhetkellä optimaalisen vektorikvantisoinnin. Tämä todettiin jo edellisessä kappaleessa vertailemalla kaavoja 6.24 ja 6.25. Toisaalta tavoitteet ovat samansuuntaisia, sillä intuitiivisesti on helppo uskoa, että hetkellinen piirrevektori sopii hyvin todellisen tilan jakaumaan ja että parannettaessa jakaumien erottelukykyä oikea tila erottuisi selvemmin muista. Väärien korjausten mahdollisesti tuottamien haittojen vähentämiseksi LVQ:ta kannattaa käyttää melko varovasti pienillä opetuskertoimilla.

Kuten taulukosta 6.11 on havaittavissa todennäköisimmän tilasekvenssin käyttö LVQ-opetuksen pohjana ei tuottanut parannuksia saavutettuun keskimääräiseen tunnistustarkkuuteen.

6.7.3 Todennäköisimmän tilasekvenssin avulla tehty itseorganisointi

Kierrosmäärä	oikein%	virhe%
1	96.2	5.6
2	96.2	5.4
3	96.2	5.6
4	96.1	5.7

Taulukko 6.12: *Tunnistustarkkuudet toistamalla opetusprosessia, johon sisältyy opetusnäytteiden todennäköisimpien tilasekvenssien haku Viterbi-algoritmilla, tilojen piirrevektorijakaumien itseorganisointi ja Baum-Welch –opetus. Ensimmäisellä rivillä on lähtöarvo, jossa Viterbi-algoritmia ei vielä käytetä.*

Korvaamalla jakaumien itseorganisaatiovaiheessa tasamittaiset tilajonot todennäköisimmillä tilasekvensseillä valitaan kutakin tilaa edustavat foneeminäytteet uudelleen ja itseorganisoidaan kartat uudelleen.

Uusien karttojen avulla voidaan koko opetusprosessi tarvittaessa toistaa samanlaisena tai hieman opetusta keventäen. Koko opetusprosessin iteratiivinen toisto on myös mahdollista, tosin laskennallisesti raskasta.

Tunnistustulokset (taulukko 6.12) osoittavat, että jakauminen täydellinen uudistaminen tuottaa yhden kerran toteutettuna pienen parannuksen, mutta seuraavat kierrokset eivät enää tuota parempia tuloksia.

6.8 Mallin parantaminen

6.8.1 Tilojen transitiotodennäköisyyksien painottaminen

Monissa tapauksissa suurin osa tunnistusvirheistä seuraa joko ylimääräisistä foneemeista oikeiden foneemien välissä tai kadonneista foneemeista. Tämä voidaan todeta tarkastelemalla tunnistustuloksen ja oikeiden sanojen välistä tunnistusmatriisia (esimerkiksi taulukko 6.13).

Mikäli ylimääräiset foneemit muodostavat selvän enemmistön virheistä, Viterbi-algoritmi suosii ilmeisesti sellaisia tilajonoja, joissa tilat vaihtuvat liian helposti. Jos taas virheet aiheutuvat pääasiassa kadonneista foneemeista, on algoritmissa suosittu tilajonoja, joissa tilanvaihtoja tapahtuu liian harvoin.

Tilatransitioiden määrään Viterbi-algoritmissa voidaan vaikuttaa kasvattamalla tai vähentämällä transitioiden todennäköisyyksiä verrattuna silmukoiden todennäköisyyksiin. Muutoksia tehtäessä kannattaa kuitenkin olla varovainen ja käyttää niiden pohjana vain tunnistustulosten keskiarvoja, sillä ylimääräisten ja kadonneiden foneemien suhteet ovat osoittautuneet varsin tapauskohtaisiksi [Kurimo 1991]. Aiemmissa testeissä vertailukohteena käytettyyn keskiarvotulokseen (painokerroin 1.0) erilaisilla painotuksilla tulevat muutokset on esitetty taulukossa 6.14.

```
          U  O  A  [  \  Y  E  I  S  M  N  L  R  J  V  H  D KPT
     .  .  7  1  .  .  1  5  .  1  .  .  .  1  2  .  2  .  2
U    1 97  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
O    1  .115  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
A    1  .  .252  1  .  .  .  .  .  .  .  .  1  .  .  .  .  .
[    .  .  .  . 97  .  .  .  .  .  .  .  .  .  .  .  .  .  .
\    .  .  .  .  .  3  .  .  .  1  .  .  .  .  .  .  .  .  .
Y    .  .  .  .  .  . 31  .  .  .  .  .  .  .  .  .  .  .  .
E    1  .  .  .  2  .  .127  .  .  .  .  .  .  .  .  .  .  .
I    4  .  .  .  .  .  .  4216  .  .  .  .  .  2  .  .  .  .
S    .  .  .  .  .  .  .  .  1133  .  .  .  .  .  .  .  .  .
M    1  .  .  .  .  .  .  .  .  . 82  1  .  .  .  .  .  .  .
N    .  .  .  .  .  .  .  .  .  .  .144  .  .  .  .  .  .  .
L    1  .  .  .  .  1  .  .  .  .  .  . 96  .  1  .  .  .  .
R    .  .  .  .  .  .  .  1  .  .  .  1  . 54  .  .  .  .  .
J    .  .  .  .  .  .  .  .  1  .  .  .  .  . 38  .  .  .  .
V    .  .  .  .  .  .  .  .  .  .  .  .  .  .  . 52  .  .  2
H    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  . 34  .  .
D    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  . 15  .
KPT 23  .  .  .  .  .  .  .  1  1  .  .  .  .  .  .  .  .255

Kirjaimia tullut lis{{  22
Kirjaimia kadonnut      33
Kirjaimia vaihtunut     23
Kirjaimia oikein      1841

Tekstiss{ oli kaikkiaan kirjaimia 1897

Eli virheit{ yhteens{  4.11 %. Oikein 97.05 %
```

Taulukko 6.13: *Tunnistamisen onnistumista mittaavan ohjelman antama tulostus. Rivin alussa on oikea foneemi ja sen jälkeen lukumäärät, kuinka monta mitäkin foneemia tunnistettiin sen paikalle testiaineistossa. Ensimmäinen numerorivi on ylimääräisten ja ensimmäinen numerosarake kadonneiden foneemien jakauma. Tulostuksessa { on ä, [Ä ja \ Ö.*

Painokerroin	oikein%	virhe%
1.0	96.2	5.6
1.5	95.7	5.8
0.5	97.0	6.1

Taulukko 6.14: *Silmukkatodennäköisyyksien painottamisen vaikutus tunnistustarkkuuteen.*

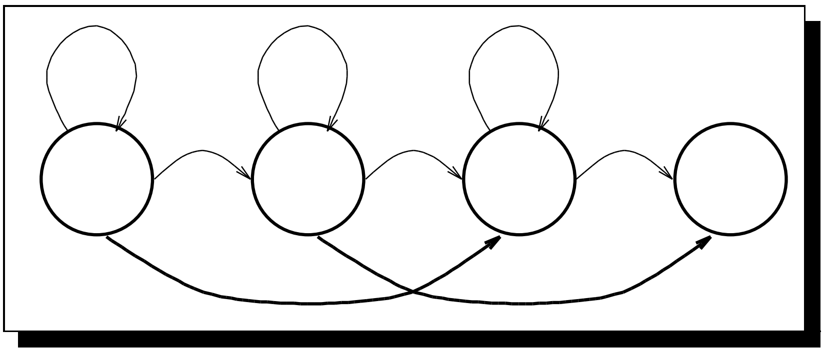
Painotus	oikein%	virhe%
	96.2	5.6
kaikki tilatransitiot	95.1	5.7
foneemitransitiot	95.9	5.3
havainnot	95.6	7.9

Taulukko 6.15: *Transitioiden ja havaintojen merkityksen painottaminen korottamalla todennäköisyydet toiseen potenssiin.*

Mikäli tunnistetut sanat ovat yleisesti oikeita sanoja pidempiä eli tilatransitioita on liikaa, voi syynä olla myös yksittäisten havaintojen tunnistustodennäköisyyksien yliarvostus transitiotodennäköisyyksiin verrattuna. Mitä enemmän painoa on hetkellisillä tunnistuksilla sitä vilkkaammin todennäköisin polku poukkoilee tilasta toiseen. Tällöin tulosta voidaan parantaa lisäämällä transitiotodennäköisyyserojen painoarvoa esimerkiksi neliöimällä. Vastaavaa menettelyä voidaan soveltaa havaintotodennäköisyyksiin jos tilatransitiot eivät ole riittävän suosittuja.

Taulukon 6.15 tunnistustulokset osoittavat, että keskiarvotulokseen verrattuna kohtuullisia tuloksia saadaan neliöimällä foneemien välisten transitioiden todennäköisyydet.

6.8.2 Foneemimallien joustavuuden lisääminen



Kuva 6.10: *Markov-malli, jossa yksittäinen välitila on ohitettavissa.*

Kun käytetään nelitilaisia ketjumalleja foneemien Markov-malleina (kuva 6.3 sivulla 50), on tavallista, että joidenkin tilojen silmukkatodennäköisyydet muuttuvat estimointiprosessissa pieniksi, jolloin viipyminen näissä tiloissa on hyvin epätodennäköistä. Tällöin on mahdollista, että siirtyminen suoraan seuraavaan tilaan olisikin todennäköisempää ja paremmin todellista tilaa kuvaavaa kuin kiertäminen harvinaisen tilan kautta.

Markov-malliin saadaan lisää joustavuutta sallimalla yhden tilan ohitukset kuvan 6.10 mukaisesti. Samalla voidaan paremmin kuvata sellaisia foneemeja, missä tilojen välinen polku voi vaihdella ympäristön tai sattuman vuoksi eri tilanteissa.

Lisättäessä mallin joustavuutta transitiotodennäköisyyksien estimointi puolestaan han-

Malli	oikein%	virhe%
ketju	96.2	5.6
ohitus	95.5	6.9

Taulukko 6.16: *Tilojen ohituksien sallimisen vaikutus tunnistustarkkuuteen.*

kaloituu. Parametreja on enemmän ja iteratiivisessa estimoinnissa käy vaihtoehtoisten reittien välillä usein niin, että pidemmän siirtymisen todennäköisyys laskee kohti nollaa. Estimointia voidaan helpottaa asettamalla pitkien transitioiden todennäköisyydeksi kiinteästi esimerkiksi puolet lyhyen transition todennäköisyydestä. Näin saadut tulokset (taulukko 6.16) osoittavat kuitenkin, että tunnistustarkkuus huononee.

6.8.3 Kiellettyjen trigrammien estäminen

Foneemien Markov-mallien välisillä siirtymistodennäköisyyksillä mallitetaan lähinnä foneemiparien välisiä suhteita. Luonnollinen mallin laajennus on käsitellä peräkkäisiä kolmen foneemin ryhmiä eli trigrammeja. Eri trigrammien esiintymistiheydet puheessa vaihtelevat suuresti; on helppo keksiä suomen kielessä mahdottomia (esimerkiksi /KKK/, /THP/ tai sanan päättyminen kaksoiskonsonanttiin) ja toisaalta hyvin tavallisia (esimerkiksi /SSA/ ja /LLA/) foneemikolmikoita.

Foneemikolmikoiden todennäköisyyksien mallittaminen vaatii huomattavasti lisää estimoitavia parametreja ja käyttö vertailtaessa mahdollisia tilajonoja on myös huomattavan työlästä eikä välttämättä teoreettisestikaan yhtä suoraviivaista toteuttaa kuin foneemiparien.

Yksinkertaisempi tapa, jolla trigrammeja voidaan kevyemmin käyttää tunnistusvirheiden poistamiseen on hakea laajasta tekstiaineistosta puuttuvat trigrammit. Näin hankitut trigrammit voidaan olettaa mahdottomiksi tekemättä kovin usein tunnistusvirheitä. Kun Viterbi-hakua toteutetaan luvussa neljä kuvatulla tavalla ja pidetään samalla kirjaa vertailtavien polkujen varrella käydyistä kahdesta viimeisestä foneemista, voidaan mahdottomia trigrammeja sisältävät polut hylätä välittömästi.

Trigrammit	oikein%	virhe%
ei	96.2	5.6
kyllä	95.8	5.3

Taulukko 6.17: *Kiellettyjen trigrammien poissulkemisen vaikutus tunnistustarkkuuteen.*

Ongelmia trigrammien käytössä aiheuttaa suomen kielessä yleiset yhdyssanat, joiden liittymäkohdissa voi esiintyä hyvin poikkeaviakin trigrammeja. Paras vain sallituja trigrammeja sisältävä sana ei myöskään ole välttämättä juuri oikea sana; trigrammien tarkastaminen viekin joskus vain kauemmas oikeasta sanasta. Esimerkiksi väärin tunnistuneen /P/:n takia yhden virheen sisältävä sana "IIENI" muuttui trigrammien tarkistuksen jälkeen sanaksi “IIMI”.

Tunnistuskeskiarvon (taulukko 6.17) perusteella trigrammien tarkistaminen vähensi virheitä vain 0.3 prosenttiyksikköä. Oikein tunnistuneiden merkkien määrän samanaikaisesta vähenemisestä voidaan päätellä, että osa trigrammien takia tehdyistä korjauksista johti väärään sanaan.

6.8.4 Kaksoisfoneemien tunnistaminen

Suomen kielelle tyypillisissä kaksoisfoneemeissa ei kirjoitetun kielen tavoin lausuttaessa muodosteta kahta samanlaista äännettä peräkkäin vaan ne sulautetaan yhdeksi erilaiseksi äänteeksi. Kaksoisfoneemien mallittamiseksi pitäisi siis oikeastaan laatia erilliset mallit, mutta koska toiset niistä ovat melko harvinaisia ja ne eroavat yksinkertaisista vastineistaan vaan kestonsa perusteella, ei erillisten mallien laatiminen kannata.

Raja-arvo	oikein%	virhe%
foneemikohtainen	96.2	5.6
150 ms	96.7	15.3
180 ms	96.6	7.6
200 ms	96.3	6.1
230 ms	95.4	6.4
250 ms	94.3	7.3

Taulukko 6.18: *Tunnistustulokset käytettäessä erilaisia kestorajoja kaksoisfoneemien tunnistamiseksi.*

Kaksoisfoneemit on helpointa erottaa tunnistetuista sanoista vasta lopuksi tutkimalla tunnistettujen foneemien kestoja ja jakamalla tiettyä rajaa pidemmät foneemit kaksoisfoneemeiksi. Tällä menetelmällä ei kuitenkaan päästä kovin hyviin tuloksiin, sillä foneemien ja kaksoisfoneemien välinen raja vaihtelee 150 – 250 millisekunnin välillä riippuen paljon siitä mikä äänne on kyseessä ja millaisessa sanassa se esiintyy.

Parempi menetelmä kaksoisfoneemien erottamiseen on laskea jokaiselle foneemille kestojakauma erikseen ja päätellä siitä foneemikohtaiset kestorajat.

Taulukossa 6.18 on vertailtu erilaisten foneemien kestorajojen vaikutusta tunnistustarkkuuteen. Foneemikohtaisilla rajoilla päästiin parhaaseen tulokseen ja sitä on käytettykin kaikissa muissa tämän työn kokeissa.

Vielä paremman tuloksen saamiseksi olisi jokaisen foneemin kestojakauma otettava huomioon jo Viterbi-haussa transitiotodennäköisyyksien ohella [Levinson 1983] , [Rabiner 1989]. Tämä kuitenkin lisää huomattavasti laskentatyötä, sillä jokaiselle vertailtavalle tilajonolle olisi laskettava erikseen siinä läpikäytävien foneemien kestojen todennäköisyydet. Myös vaatimukset opetusaineiston määrästä ja edustavuudesta kovenevat, koska jokaisesta foneemista tarvitaan oikeassa suhteessa erimittaisia näytteitä riittävästi, jotta kestojakauman parametrit saadaan uskottavasti estimoitua.

Luku 7

Yhteenveto

Suomenkielisen puheen tunnistamista varten tässä työssä kehitettiin foneemeille jatkuvan havaintojakauman Markov-mallit, joiden opettamisessa hyödynnettiin itseorganisoituvia piirrekarttoja, oppivaa vektorikvantisointia sekä iteratiivista Baum-Welch –estimointia. Foneemien Markov-malleista yhdistetään kätkettyjen Markov-mallien menetelmillä sanojen foneettiset kuvaukset.

Testeissä oli käytettävissä laboratoriossa aikaisemmin kerätyt näytteet kolmelta miespuoliselta puhujalta, yhteensä 311 sanaa käsittävä aineisto puhuttuna neljään kertaan. Tunnistustarkkuutena käytettiin aina keskiarvoa neljästä testistä, joissa tunnistettiin vuorotellen kukin puhekerta ja mallit opetettiin kolmella riippumattomalla puhekerralla.

Verrattuna laboratoriossa tällä hetkellä olevaan vanhaan järjestelmään, jossa tosiaikaisen toiminnan saavuttamiseksi oli käytetty pieniä karttoja, olivat tunnistustulokset tarkkuudeltaan parempia, sillä vastaavissa testeissä tunnistusvirheiden määrä laski puhujilla keskimäärin 11.3 prosentista 6.9 prosenttiin. Näissä testeissä soinnittomat klusiilit /k/, /p/ ja /t/ luettiin samaan luokkaan ja tunnistus tehtiin vain yhden koodikirjan (kepstripiirteet) perusteella. Tunnistusvirheitä automaattisesti korjaavaa jälkikäsittelyä (DEC, kappale 3.3.6) ei tehty. Tulokset eivät ole kuitenkaan suoraan vertailukelpoisia.

Foneemin Markov-malliksi valittiin nelitilainen ketju, jonka tilat mallittavat puhuttaessa tapahtuvista ääniväylän akustisten ominaisuuksien muutoksista saatavia havaintoja. Kullakin tilalla on seuraavien tilatransitioiden todennäköisyysjakauma ja syntyvästä signaalista laskettujen hetkellisten kepstrikertoimien muodostamien piirrevektorien jatkuva todennäköisyysjakauma.

Parhaimmat tunnistustulokset saatiin mallilla, jossa tilan piirrevektorien todennäköisyysjakaumaa approksimoitiin 25 moniulotteisen normaalijakauman lineaarikombinaatiolla. Tässä mallissa normaalijakaumien komponenttien kovarianssimatriisina käytettiin yksikkömatriisiapproksimaatiota, jolloin jakaumaa kuvaavat parametrit voitiin keskittää jakauman huippujen paikan ja koon kuvaamiseen.

Tunnistustuloksia vertailemalla tutkittiin myös parasta menetelmää mallin opettamiseksi. Sopivaksi osoittautui antaa ensin normaalijakaumien huippuja kuvaavien vektorien itseorganisoitua pienimmän kvantisointivirheen tuottamiseksi tilojen piirrevektorijakaumilla. Menetelmänä käytettiin itseorganisoituvia piirrekarttoja [Kohonen 1990], joilla säilytetään myös approksimoitavan jakauman topologia niin hyvin kuin mahdollista. Oppivilla vektorikvantisointimenetelmillä voidaan muokata jakaumaa vielä tilojen erotteluominaisuuksien parantamiseksi. Lopuksi estimoidaan iteratiivisesti normaalijakaumien painokertoimet ja tilojen transitiotodennäköisyydet. Muiden parametrien muuttuessa normaalijakaumien huippujen sijaintienkin annetaan hakeutua edelleen kohti optimaalisia pisteitä.

Tutkimuksissa kokeiltiin myös mallien avulla opetusdatasta laskettujen todennäköisimpien polkujen käyttöä mallien havaintojakaumien hienosäädössä LVQ-tyyppisen opetuksen kanssa. Samoin kokeiltiin jakaumien määräämistä kokonaan uudelleen laskemalla parannetut alkuarvot todennäköisimpien polkujen antaman tilajaon perusteella. Saavutetut parannukset tunnistustuloksissa olivat vähäpätöisiä verrattuna menetelmien aiheuttamaan laskentatyön lisäykseen.

Lopuksi tutkittiin vielä keinoja tunnistusvirheiden vähentämiseksi haettaessa mitattuun puhesignaaliin parhaiten sopivaa tilajonoa. Tutkitut menetelmät tähtäsivät tehtyjen havaintojen ja foneemiparien todennäköisyyksien sopivimpaan painotukseen, mallien joustavuuden lisäämiseen, kolmen foneemin ryhmien tarkasteluun ja kaksoisfoneemien erottelun parantamiseen. Merkittäviä parannuksia vertailutuloksiin ei kuitenkaan saavutettu. Tosin vertailutuloksiksi oli muiden kokeiden perusteella valittu parhaat saavutetut tulokset, joissa tunnistusvirheitä ei enää kovin paljoa ollut. Mikäli vertailutuloksena käytettäisiin jotain muuta kuin parhaita tuloksia, virheitä vähentävistä keinoista olisi varmasti enemmän hyötyä.

Kokeissa opetusmateriaalina käytetty kolme kertaa toistettu 311 sanan aineisto saattoi varsinkin harvinaisimpien foneemien osalta olla liian suppea; /d/:stä saatiin yhteensä 45 ja /ö/:stä vain 12 näytettä. Toisaalta harvinaisten foneemien mallit eivät ole tunnistuksen kannalta kovin tärkeitä, sillä ne eivät yleensä sekaannu yleisempiin foneemeihin, joilla on paremmat mallit, eivätkä niiden tunnistuksessa sattuvat virheet paljoa vaikuta kokonaistunnistustarkkuuteen.

Tilojen tuottamien tiheysfunktioiden approksimoinnissa olisi kiinnostavaa soveltaa tässä työssä kehitettyihin menetelmiin viitteessä [Huang et al. 1989a] kuvattuja puolijatkuvia HMM:iä(SCHMM), joissa kaikki tilat käyttävät samoja normaalijakaumia erilaisilla lineaarikombinaatioilla. Myös välimuodot, esimerkiksi sellainen, jossa saman foneemin tilat käyttäisivät samojen normaalijakaumien erilaisia yhdistelmiä, voisivat tuottaa mielenkiintoisia tuloksia.

Mielenkiintoisia aiheita jatkotutkimuksiin saattaisivat olla kolmen foneemin ryhmien eli trigrammien ja foneemien kestojakaumien tarkempi soveltaminen. Deltakepstrit (peräkkäisten piirrevektorien kepstrikertoimien erotukset) tai peräkkäisistä piirteistä yhdistetyt pidemmät piirrevektorit saattavat parantaa reunimmaisten tilojen malleja. Viitteissä [Bahl et al. 1986] ja [Bahl et al. 1988] esitetyt erilaiset opetuskriteerit, kuten havaintojen ja Markov-mallien keskinäisen informaation maksimointi (MMI) ja virhetunnistusten määrän minimointi, saattaisivat lisäksi tarjota uusia näkymiä.

Lähdeluettelo

[Averbuch 1987] A. Averbuch. Experiments with the TANGORA 20000 word speech recognizer. *Proceedings of ICASSP*, sivut 701–704, New York, 1987.

[Bahl et al. 1983] L. Bahl, F. Jelinek, R. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190, 1983.

[Bahl et al. 1986] L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. *Proceedings of ICASSP*, sivut 49–52, 1986.

[Bahl et al. 1988] L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer. A new algorithm for the estimation of hidden Markov model parameters. *Proceedings of ICASSP*, sivut 493–496, 1988.

[Baker 1989] Janet M. Baker. DragonDictate^{TM}-30K: Natural language speech recognition with 30,000 words. *Proceedings of the European Conference on Speech Communication and Technology (Eurospeech89)*, osa 2, sivut 161–163, Paris, France, September 1989.

[Davis et al. 1980] Stephen B. Davis, Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

[Huang et al. 1989a] X.D. Huang, M.A. Jack. Semi-continuous hidden markov models for speech signals. *Computer Speech and Language*, 3, 1989.

[Huang et al. 1989b] X.D. Huang, M.A. Jack. Unified techniques for vector quantization and hidden Markov modelling using semi-continuous models. *Proceedings of ICASSP*, osa 1, sivut 639–642, New York, 1989.

[Juang et al. 1990] Biing-Hwang Juang, L.R. Rabiner. The segmental K-means algorithm for estimating parameters of hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38:1639–1641, 1990.

[JUANG 1985] B.H. Juang. Maximum likelihood estimation for mixture multivariate stochastic observation of Markov chains. *AT&T Technical Journal*, 64:1235–1249, 1985.

[KANGAS 1988] Jari Kangas. Soinnittomien klusiilien erottelu Otaniemen puheentunnistusjärjestelmässä. Diplomityö, Teknillinen Korkeakoulu, 1988.

[KASKI 1990] Sami Kaski. Kepstrien käyttö puheentunnistuksessa. Erikoistyö TKK:n informaatiotekniikan laboratorioon, 1990.

[KOHONEN ET AL. 1984] Teuvo Kohonen, Kai Mäkisara, Tapio Saramäki. Phonotopic maps - insightful representation of phonological features for speech recognition. *Proceedings of 7th International Conference on Pattern Recognition*, sivut 182–185, Montreal, Canada, 1984.

[KOHONEN ET AL. 1992] Teuvo Kohonen, Jari Kangas, Jorma Laaksonen, Kari Torkkola. LVQPAK: A program package for the correct application of learning vector quantization algorithms. *Proceedings of the International Conference on on Artificial Neural Networks (to be published)*, Bristol, 1992.

[KOHONEN 1982] Teuvo Kohonen. Clustering, taxonomy, and topological maps of patterns. *Proceedings of the 6th International Conference on Pattern Recognition*, osa 1, sivut 114–128, Munich, Germany, 1982.

[KOHONEN 1986A] Teuvo Kohonen. Dynamically expanding context, with application to the correction of symbol strings in recognition of continuous speech. *Proceedings of the 8th International Conference on Pattern Recognition*, sivut 1148–1151, Paris, France, 1986.

[KOHONEN 1986B] Teuvo Kohonen. Learning vector quantization for pattern recognition. Raportti TKK-F-A601, Helsinki University of Technology, 1986.

[KOHONEN 1989] Teuvo Kohonen. *Self-Organization and Associative Memory.* Springer-Verlag, Berlin-Heidelberg-New York-Tokio, 3 edition, 1989.

[KOHONEN 1990] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, sivut 1464–1480, 1990.

[KURIMO 1990] Mikko Kurimo. Relaksaation käyttö kvasifoneemien tunnistuksessa. Erikoistyö TKK:n informaatiotekniikan laboratorioon, 1990.

[KURIMO 1991] Mikko Kurimo. Jatkuvan jakauman kätketyt Markov-mallit puheentunnistusjärjestelmässä. Erikoistyö TKK:n informaatiotekniikan laboratorioon, 1991.

[LAAKSONEN ET AL. 1988] Kaino Laaksonen, Anneli Lieko. *Suomen kielen äänne– ja muoto–oppi.* Oy Finn Lectura Ab, 1988.

[Laaksonen 1990] Jorma Laaksonen. Segmentointimenetelmä Otaniemen puheentunnistusjärjestelmään. Diplomityö, Teknillinen Korkeakoulu, 1990.

[Lee 1988] Kai-Fu Lee. Large-vocabulary speaker-independent continuous speech recognition: The SPHINX system. Raportti CS-88-148, Carnegie Mellon University, 1988.

[Levenshtein 1966] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Sov. Phys.-Dokl.*, 10(8):107–710, 1966.

[Levinson 1983] S.E. Levinson. Continuously variable duration hidden Markov models for automatic speech recognition. *Computer, Speech and Language*, 1(1):29–45, 1983.

[Linde et al. 1980] Y. Linde, A. Buzo, R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communication*, 28(1):84–95, 1980.

[Liporace 1982] Louis A. Liporace. Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Transactions on Information Theory*, 5:729–734, 1982.

[MacQueen 1967] J. MacQueen. Some methods for classification and analysis of multivariate observations. *Proceedings of Fifth Berkeley Symposium on Math. Statist. and Prob.*, sivut 281–297, 1967.

[Nakagawa et al. 1990] Seiichi Nakagawa, Yoshimitsu Hirata. Comparison among time-delay neural networks, LVQ2, discrete parameter HMM and continuous parameter HMM. *Proceedings of ICASSP*, osa 1, sivut 509–512, 1990.

[O'Shaughnessy 1987] Douglas O'Shaughnessy. *Speech Communication.* Addison-Wesley, 1987.

[Poritz 1988] Alan B. Poritz. Hidden Markov models: A guided tour. *Proceedings of ICASSP*, osa 1, sivut 7–13, 1988.

[Rabiner et al. 1983] L.R. Rabiner, S.E. Levinson, M.M. Sondhi. On the application of vector quantization and hidden Markov models to speaker-independent isolated word recognition. *Bell System Technical Journal*, 62:1075–1105, 1983.

[Rabiner 1989] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of ICASSP*, osa 1, sivut 267–295, 1989.

[Ritter et al. 1986] H.J. Ritter, K.J. Schulten. On the stationary state of Kohonen's self-organizing sensory mapping. *Biol. Cybern.*, 54(1):99–106, 1986.

[Ritter et al. 1988] H.J. Ritter, K.J. Schulten. Convergence properties of Kohonen's topology conserving maps: Fluctuations, stability, and dimension selection. *Biol. Cybern.*, 60(1):59–71, 1988.

[Ritter 1989] H.J. Ritter. Asymptotic level density for a class of vector quantization processes. Raportti TKK-F-A9, Helsinki University of Technology, 1989.

[Therrien 1989] Charles W. Therrien. *Decision, Estimation and Classification: An Introduction to Pattern Recognition and Related Topics.* John Wiley and Sons, 1989.

[Tohkura 1987] Yoh'ichi Tohkura. A weighted cepstral distant measure for speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(10):1414–1422, 1987.

[Torkkola et al. 1990] Kari Torkkola, Mikko Kokkonen. A comparison of two methods to transcribe speech into phonemes: A rule-based method vs. backpropagation. *Proceedings of 1990 International Conference on Spoken Language Processing (ICSLP-90)*, osa 1, sivut 673–676, Kobe,Japan, November 1990.

[Torkkola et al. 1991a] Kari Torkkola, Jari Kangas, Pekka Utela, Sami Kaski, Mikko Kokkonen, Mikko Kurimo, Teuvo Kohonen. Status report of the finnish phonetic typewriter project. *Proceedings of the International Conference on Artificial Neural Networks*, osa 1, sivut 771–776, Espoo,Finland, June 1991.

[Torkkola et al. 1991b] Kari Torkkola, Mikko Kokkonen, Mikko Kurimo, Pekka Utela. Improving short-time speech frame recognition results by using context. *Proceedings of the 2nd European Conference on Speech Technology and Communication*, osa 2, sivut 793–796, Milano,Italy, September 1991.

[Torkkola 1990] Kari Torkkola. A combination of neural network and low level AI techniques to transcribe speech into phonemes. *Proceedings of COGNITIVA-90*, sivut 637–644, Madrid,Spain, November 1990.

[Vonnegut 1989] Kurt Vonnegut. *Galapagos.* Kustannusosakeyhtiö Tammi, Helsinki, 1989.