UNIVERSITATEA TEHNICĂ CLUJ-NAPOCA

Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei

Margit Antal

CONTRIBUŢII LA RECUNOAŞTEREA VORBIRII ŞI A VORBITORULUI

Teză de doctorat

PREŞEDINTE: Prof.dr.ing. Dan Pitică -decan, Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei

MEMBRI:

1. Prof.dr.ing. Gavril Toderean -conducător ştiinţific, Universitatea Tehnică din Cluj-Napoca
2. Prof.dr.ing. Inge Gavăt -referent, Universitatea Politehnică din Bucureşti
3. Prof.dr.ing. Corneliu Toma -referent, Universitatea Politehnică din Timişoara
4. Prof.dr.ing. Corneliu Rusu -referent, Universitatea Tehnică din Cluj-Napoca

2006

Mulţumiri

Realizarea acestei lucrări a fost posibilă datorită suportului şi sprijinului a mai multor persoane.

Pentru posibilitatea cercetării realizate şi a scrierii acestei lucrări mulţumesc domnului profesor Dr. Gavril Toderean, care m-a sprijinit pe tot parcursul perioadei de doctorat. El a fost cel care a avut încredere în mine şi a făcut să-mi pară munca de cercetare o experienţă plăcută.

Doresc să-i mulţumesc soţului meu Győző Nemes pentru primele lecţii de prelucrarea semnalelor şi pentru suportul şi răbdarea de-a lungul anilor de studii de doctorat. Fără el nu aş fi putut realiza această lucrare.

Dintre cei care m-au sprijinit în special prin discuţii ştiinţifice doresc să-i mulţumesc membrilor grupului de cercetare de inteligenţă artificială de la universitatea Szeged, în special domnului profesor János Csirik, cercetătorilor András Kocsor şi László Tóth.

În final, dar nu în ultimul rând doresc să mulţumesc familiei pentru sprijinul uman, fără care n-aş fi avut posibilitatea de a elabora această lucrare.

Cuprins

Listă de figuri

Listă de tabele

Listă de algoritmi

Capitolul 1

Introducere

Această lucrare investighează modele statistice pentru două probleme, şi anume recunoaşterea vorbitorului şi recunoaşterea vorbirii. Modelele utilizate au fost tehnicile de grupare şi modelele Markov ascunse.

În cazul recunoaşterii vorbitorului obiectivul principal este studierea conţinutului modelelor de vorbitori din punct de vedere fonetic. Un al doilea obiectiv este utilizarea altor informaţii, decât cele acustice pentru recunoaşterea vorbitorului. În acest caz se doreşte investigarea categoriilor fonetice, ca informaţie de ordin superior.

În cazul recunoaşterii vorbirii obiectivul principal este investigarea modelelor mixturilor Gassiene pentru modelarea fonemelor. Deoarece foarte multe articole [19, 40, 52, 57, 59, 66, 80] subliniază efectul neglijabil al probabilităţilor de tranziţie în cazul modelelor Markov ascunse, obiectivul este de a studia comportamentul sistemelor de recunoaştere fără aceşti parametri. Un al doilea obiectiv este modificarea algoritmilor de decodare pentru integrarea modelării duratei fonemelor în acestea.

1.1 Prezentarea capitolelor

Capitolul 1 este introducerea în care se prezintă motivaţiile acestei teze, structura pe capitole şi contribuţiile originale.

Capitolul 2 este dedicat prezentării tuturor noţiunilor teoretice care au fost utilizate în această lucrare. Sunt prezentate noţiuni de fonetică, metodele convenţionale de extragerea caracteristicilor, tehnicile de grupare, modelele Markov ascunse şi tehnicile de decodare.

Capitolul 3 prezintă problema recunoaşterii vorbitorului precum şi tehnicile

principale de modelare a vorbitorului. Capitolul se limitează la prezentarea metodelor de recunoaştere independente de text. Ca element original am studiat distribuţia conţinutului fonetic în modele de vorbitori de tip mixturi gaussiene. Au fost studiate două aspecte, distribuţia fonemelor individuale precum şi distribuţia categoriilor fonetice. Un alt element original introdus în acest capitol este modelarea vorbitorilor utilizând datele specifice ale unei singure categorii fonetice, cum ar fi vocalele sau nazalele. Categoriile fonetice au fost ordonate din punctul de vedere al puterii lor de discriminare a vorbitorului. În sfârşit a fost propus un model original, care grupează datele acustice pe categorii fonetice. Capitolul prezintă pentru toate modelele propuse şi evaluarea acestora pe baze de date publice.

Capitolul 4 prezintă problema clasificării fonemelor, unitatea acustică utilizată pentru recunoaşterea vorbirii. Fonemele sunt modelate cu mixturi gaussiene. Capitolul prezintă rezultate experimentale pentru clasificarea fonemelor, grupurilor, respectiv categoriilor de foneme. Pe baza rezultatelor experimentale, care indică faptul, că fonemele sunt confundate cu alte foneme din aceeaşi categorie, se propune un nou tip de clasificator, clasificatorul cu două niveluri. Sunt prezentate rezultatele experimentale pentru acest nou tip de clasificator. În final sunt prezentate rezultatele pentru clasificarea cadru cu cadru, care se utilizează tot pentru evaluarea modelelor. Măsurătorile sunt prezentate pentru două baze de date publice, TIMIT şi OASIS Numbers.

Capitolul 5 este o continuare naturală a capitolului precedent, se trece de la problema clasificării la problema recunoaşterii fonemelor. Ca algoritm de decodare se utilizează algoritmul Viterbi. Deoarece modelele de foneme utilizate sunt modele independente de context şi se utilizează modele GMM, algoritmul Viterbi produce un număr mare de inserări. Din acest motiv se propune o modificare a acestui algoritm, introducând un parametru empiric pentru a putea controla erorile de tip inserare. Acest parametru empiric poate fi privit în mod indirect ca şi un model simplist pentru modelarea duratei fonemelor. Rezultatele noastre obţinute cu modele mai simple (cu un număr redus de parametri) se pot compara cu cele obţinute cu modele HMM. Rezultatele experimentale sunt prezentate atât pentru determinarea parametrului empiric, cât şi pentru recunoaşterea fonemelor.

Capitolul 6 investighează problema modelării duratei fonemelor. Sunt prezentate rezultate statistice obţinute pentru durata fonemelor pentru cele două baze de date şi sunt propuse o serie de modele simple pentru modelarea duratei.

Capitolul 7 prezintă problema recunoaşterii cuvintelor izolate şi sunt prezentate două modalităţi pentru rezolvarea acestei probleme. Prima modalitate, cea

a modelării cuvintelor doar cu mixturi gaussiene, abordare originală şi simplistă, dar care dă rezultate excepţionale pentru un număr redus de cuvinte suficient de diferite, cum ar fi cifrele. A doua modalitate este cea generală, modelele de cuvinte sunt formate din modele de foneme. În acest caz trebuie rezolvată organizarea vocabularului, pentru care am utilizat varianta arborescentă, structura de date trie. Capitolul se termină cu prezentarea rezultatelor experimentale.

Ultimul capitol al acestei teze, capitolul 8, prezintă concluziile generale şi direcţiile de dezvoltare.

1.2 Contribuţii originale

- Am studiat conţinutul fonetic al modelelor de vorbitori. Am studiat distribuţia categoriilor fonetice, respectiv a fonemelor în componentele acestor modele. Am arătat că aceste componente nu sunt pure din punct de vedere fonetic.

- Ordonarea categoriilor fonetice pe baza puterii lor de discriminare a vorbitorilor. Am arătăt că vocalele şi nazalele au puterea discriminativă cea mai mare şi că utilizând doar vocalele unui vorbitor, acesta poate fi identificat cu mare acurateţe.

- Construirea modelelor de vorbitori structurate fonetic. Am arătat că structurând datele unui vorbitor pe bază de categorii fonetice şi creând submodelele acestor categorii fonetice, modelul obţinut are performanţe similare cu modelele obişnuite.

- Am demostrat în mod experimental că modelele GMM au performanţe similare cu modelele HMM pentru problema clasificării fonemelor. Acesta demonstrează faptul că probabilităţile de tranziţie din cadrul modelelor HMM pot fi neglijate.

- Am popus un clasificator cu două niveluri pentru problema clasificării fonemelor, care este mult mai eficient din punctul de vedere al calculelor cu o acurateţe uşor scăzută faţă de clasificarea obişnuită.

- Am propus o modificare a algoritmului Viterbi, pentru a putea fi folosit şi pentru modele GMM. Modelele de foneme GMM cu algoritmul Viterbi modificat furnizează rezultate la fel de bune ca şi modelele HMM, dar cu reducerea semnificativă a calculelor.

- Am prezentat un studiu referitor la duratele fonemelor pentru două baze de date. Pe baza acestor studii am elaborat o nouă modificare a algoritmului Viterbi pentru a putea include şi modelarea duratei fonemelor.
- În final am realizat un sistem cu componentele software necesare pentru recunoaşterea vorbitorului şi a vorbirii.

Capitolul 2

Bazele teoretice

2.1 Introducere

În acest capitol prezentăm bazele teoretice necesare atât pentru recunoaşterea vorbitorului cât şi pentru recunoaşterea vorbirii. Vom începe cu descrierea extragerii caracteristicilor utilizată în ambele sisteme, specificând particularităţile fiecărui sistem. Continuăm cu prezentarea noţiunilor fonetice strict necesare pentru elaborarea acestei lucrări. Următoarele două secţiuni prezintă metode matematice ca, tehnica grupărilor, respectiv teoria modelelor Markov ascunse utilizate în implementarea celor două sisteme. Penultima secţiune prezintă trei noţiuni de bază ale sistemelor pentru recunoaşterea vorbirii continue: modelele de limbaj, măsurarea erorii, precum şi tehnicile de decodare. În final sunt prezentate cele două baze de date de sunet utilizate în experimente.

Scopul unui sistem de recunoaşterea vorbirii este extragerea conţinutului lexical din semnalul vocal. Acesta se realizează în mai mulţi paşi. Detalii cu privire la acest subiect se găsesc în [26, 40, 67, 79]. Componentele unui asemenea sistem sunt următoarele:

- *Extragerea caracteristicilor:* În această parte a sistemului din forma de undă a semnalului vocal se extrag vectorii acustici. Aceşti vectori trebuie să conţină informaţii pe baza cărora se pot distinge unităţile lingvistice cum ar fi fonemele. Aceste informaţii ar trebui să fie robuste, adică pentru aceeaşi unitate lingvistică să fie asemănătoare indiferent de vorbitor sau de condiţiile de înregistrare. Desigur, asemenea caracteristici nu sunt uşor de găsit, semnalul fiind influenţat atât de diferite surse de zgomot, cât şi de vorbitor.

- *Modelarea acustică:* Vectorii acustici se utilizează pentru a crea câte un model pentru fiecare unitate ligvistică (de obicei foneme). Acest pas se numeşte şi pasul de antrenare. Scopul antrenării este estimarea parametrilor modelelor astfel încât aceste modele să reprezinte cât mai adecvat datele de antrenare şi de a minimiza eroarea de recunoaştere în cazul datelor neprezentate sistemului în faza de antrenare. Există două abordări de bază pentru modelarea acustică: una este utilizarea modelelor Markov ascunse utilizând mixturi gaussiene pentru estimarea probabilităţilor de emisie a stărilor [68](HMM/GMM) şi cealaltă este utilizarea reţelelor neuronale în loc de mixturi gaussiene în modelul Markov ascuns [18](HMM/ANN).

- *Decodare:* În această fază se determină secvenţa de cuvinte cea mai probabilă pentru secvenţa de vectori acustici extraşi din semnalul vocal. Decodarea este de fapt o căutare într-un spaţiu acustic utilizând mai multe surse de cunoştinţe, cum ar fi probabilităţi ale secvenţelor de vectori acustici în modelele fonetice, constrângeri impuse de vocabular, probabilităţi de secvenţe de cuvinte, care se numeşte model limbaj. În cazul sistemelor HMM/GMM decodarea se realizează de obicei cu algoritmul Viterbi sau o formă modificată a acestuia.

Recunoaşterea vorbitorului este procesul în care se recunoaşte vorbitorul pe baza unui material vocal furnizat de către vorbitor. Scopul este extragerea acelor informaţii din semnalul vocal, care caracterizează vorbitorul şi nu conţinutul lingvistic al semnalului. Fiecare vorbitor este caracterizat prin rata vorbirii, prin modul de pronunţare a diferitelor unităţi fonetice etc. Aceste particularităţi caracteristice vorbitorului sunt dificil de extras. Componentele standard a unui sistem de recunoaşterea vorbitorului sunt:

- *Extragerea caracteristicilor:* Se extrag informaţii pe baza cărora vorbitorii pot fi identificaţi, discriminaţi faţă de ceilalţi vorbitori ai sistemului.

- *Calculul scorului:* În acest modul se verifică ipoteza că un semnal vocal este produs de un anumit vorbitor. În funcţie de tipul problemei (identificare sau verificare) se va folosi un model adiţional numit şi model fundal [16].

- *Decizie:* În cazul sistemelor de verificarea vorbitorului vorbitorul este acceptat sau respins, iar în cazul sistemelor de identificare se va determina identitatea vorbitorului pe baza similarităţilor faţă de vorbitorii cunoscuţi

de către sistem. Problema de verificare este de fapt o decizie binară, iar problema de identificare este o problemă de clasificare.

Dacă vorbitorul poate rosti orice text, când solicită identificarea sau verificarea, sistemul se numeşte independent de text, iar dacă conţinutul lingvistic ce trebuie reprodus este determinat în prealabil, sistemul se numeşte dependent de text. În cel de-al doilea caz se utilizează atât recunoaşterea vorbitorului cât şi recunoaşterea materialului vorbit. Următoarele articole furnizează detalii despre acest subiect [16, 21, 36].

2.2 Elemente de fonetică

2.2.1 Semnalul vocal

Semnalul vocal este compus dintr-o succesiune de sunete, al căror aranjament este guvernat de legi fonetice [75]. Unda acustică vocală este generată prin mişcări fiziologice voluntare ale următoarelor elemente anatomice: coarde vocale, tract vocal, limbă, cavitatea nazală, buze, văl palatin.

Aerul expulzat din plămâni trece prin trahee, iar apoi printre coardele vocale. La generarea sunetelor sonore aerul împins spre buze duce la deschiderea şi închiderea coardelor vocale într-un ritm dependent de presiunea aerului în trahee şi de ajustările fiziologice, ce au loc. Cu cât tensiunea în coarde este mai mare, cu atât este mai mare şi frecvenţa fundamentală a semnalului vocal. Deschiderea dintre coardele vocale se numeşte glotă.

Tractul vocal este un tub acustic neuniform, ce se întinde de la glotă la buze, forma sa variind în timp. Componentele anatomice, ce produc aceste variaţii, sunt buzele, maxilarele, limba şi vălul palatin. În timpul generării sunetelor nenazale vălul palatin separă tractul vocal de cavitatea nazală, care constituie un tub acustic adiţional pentru generarea şi transmiterea sunetelor nazale: aerul trece prin glotă, iar tensiunea în coardele vocale este astfel ajustată, încât acestea vor vibra într-o oscilaţie de relaxare, generând pulsuri cvasiperiodice de aer, care excită tractul vocal.

Sunetele nesonore sunt generate prin crearea unei constricţii pe tractul vocal şi forţarea aerului prin ea cu o viteză suficient de mare pentru a se genera un zgomot cu spectru larg, care va excita tractul vocal. Constricţia, împreună cu vibraţia coardelor vocale, produce fricativele sonore.

Sunetele plozive se obţin prin creşterea presiunii aerului în cavitatea bucală, urmată de eliberarea sa bruscă printre buze.

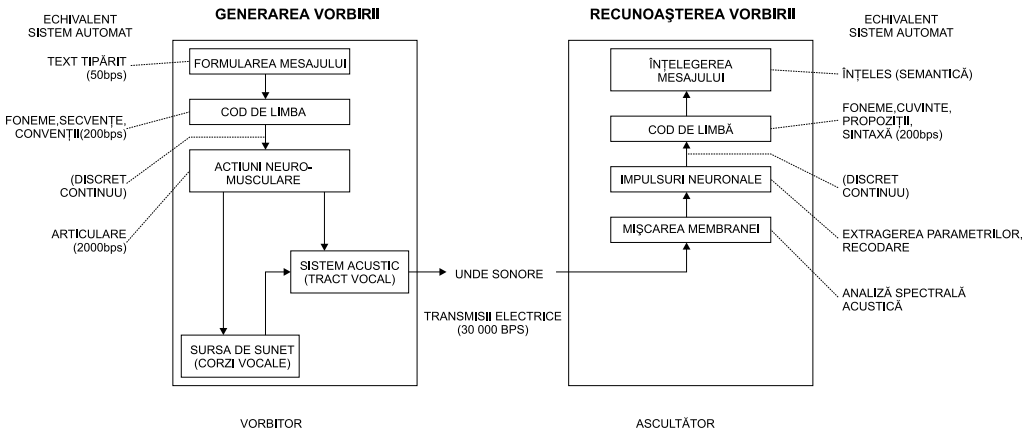


Figura 2.1: Schema producerii şi perceperii sunetului - varianta 1 (după [67])

2.2.2 Producere şi percepere

Figura 2.1 ne arată o diagramă schematică privind producerea şi perceperea vorbirii la fiinţe umane. Procesul de producere (generarea vorbirii) începe atunci, când vorbitorul formulează un mesaj (în minte - memorie) pe care doreşte să-l transmită cuiva care îl ascultă, prin vorbire. În cazul maşinii, acestui pas îi corespunde crearea textului, care conţine cuvintele mesajului. Pasul următor este conversia mesajului într-un cod al limbajului. Acestui pas îi corespunde conversia textului scris al mesajului într-o mulţime de foneme corspunzătoare sunetelor care alcătuiesc cuvintele. Odată ce a fost selectat limbajul, vorbitorul trebuie să execute o serie de comenzi neuromusculare pentru a pune în vibraţie coardele vocale şi pentru a forma tractul vocal pentru a produce secvenţa de sunete corespunzătoare, astfel obţinându-se rezultatul: semnalul vocal.

Odată ce semnalul a fost generat şi propagat către ascultător, începe procesul de percepere a semnalului. Primul pas în perceperea semnalului este la nivelul urechilor: partea interioară a acesteia, membrana, realizează o analiză spectrală a semnalului intrat. Un proces neuronal realizează conversia semnalului spectral obţinut la ieşirea membranei într-un semnal de activare al nervului auzului. Acest pas se mai numeşte şi pasul extragerii trăsăturilor. După acest pas activitatea neuronală de-a lungul nervului auzului, cu ajutorul creierului, este convertită într-un cod al limbajului, obţinându-se în final semnificaţia mesajului.

O altă reprezentare a procesului producere-percepere este dată de figura 2.2. Aici putem urmări stadiile procesului, precum şi viteza de propagare a semnalului între stadii. Viteza formulării mesajului este cel mai lent, aproximativ 50 bps

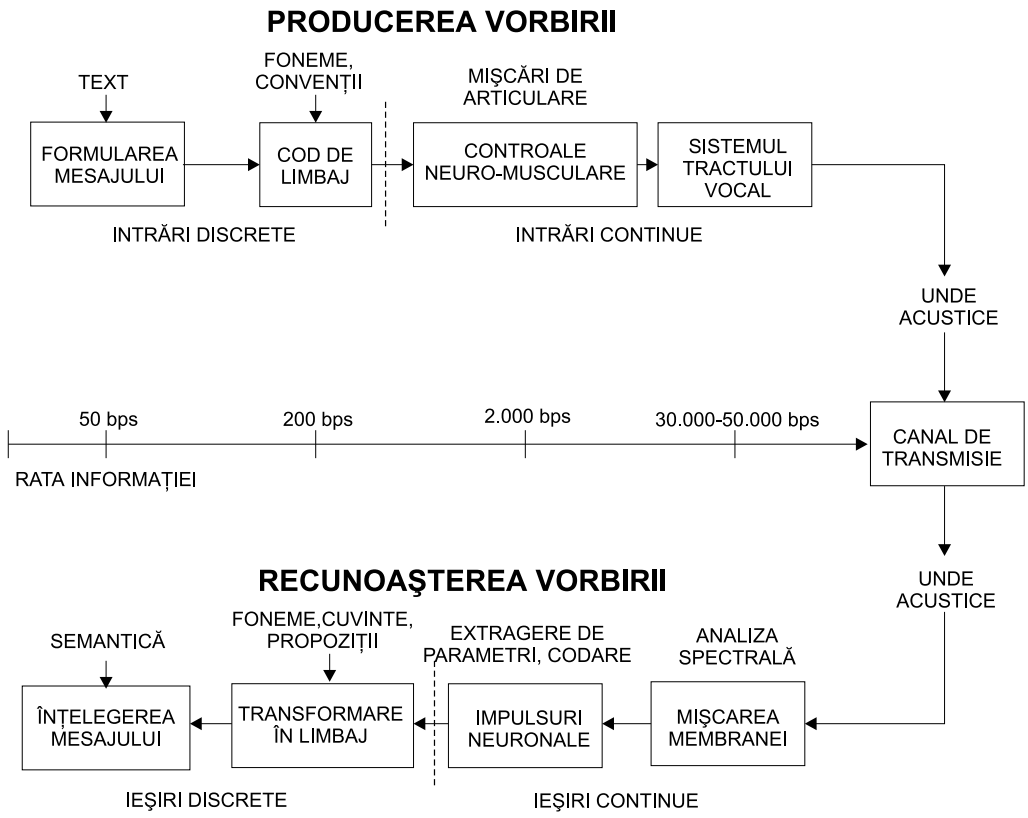


Figura 2.2: Schema producerii şi a perceperii sunetului - varianta 2 (după [67])

corespunzând cu 8 sunete pe secundă, unde fiecare sunet este unul dintre cele 50 de simboluri. După această codificare viteza creşte la 200 bps. În stadiul următor la conversia informaţiei în semnal viteza atinge rata de 2000 bps la nivelul neuromuscular şi 30000-50000 bps la nivelul semnalului acustic.

2.2.3 Transcriere fonetică

Procesul de translatare a unei rostiri într-o secvenţă de simboluri, numite foneme, se numeşte transcriere fonetică. Stabilirea unui alfabet fonetic a stârnit multe discuţii. În cazul limbii engleze utilizarea reprezentării ortografice cu ajutorul alfabetului roman nu este corespunzătoare. În anul 1888, un grup de foneticieni din Europa, au publicat alfabetul fonetic internaţional (**IPA** - *International Phonetic Alphabet*), pentru a facilita şi standardiza trancripţiile fonetice. Acest alfabet fonetic are suficiente simboluri pentru a putea descrie toate limbile lumii.

IPA este foarte comod pentru transcriere cu scris de mână, dar simbolurile nu pot fi scrise cu tastatura unui calculator. Din această cauză s-a dezvoltat un nou alfabet fonetic, alfabetul **ARPAbet**. Acest alfabet fonetic s-a dezvoltat sub auspiciul ARPA (Advanced Research Projects Agency) din SUA. Există două versiuni ale alfabetului ARPAbet, prima versiune utilizând simboluri de o literă şi o altă versiune utilizând doar secvenţe de litere. Tabelul 1 prezintă a doua versiune a alfabetului.

2.2.4 Clasificarea fonemelor

Există mai multe criterii, pe baza cărora fonemele pot fi clasificate. Fonemele pot fi grupate pe baza următoarelor proprietăţi:

- Forma de undă
- Caracteristica de frecvenţă
- Modul de articulare
- Locul de articulare
- Tipul excitaţiei
- Staţionaritate

Simbol	Exemplu	Simbol	Exemplu
iy	heed	v	vice
ih	hid	th	thing
ey	hayed	dh	then
eh	head	s	so
ae	had	z	zebra
aa	hod	sh	show
ao	hawed	zh	measure
ow	hoed	hh	help
uh	hood	m	mom
uw	who'd	n	noon
er	heard	nx	sing
ax	ago	l	love
ah	mud	el	cattle
ay	hide	em	some
aw	how'd	en	son
oy	boy	dx	batter
ix	roses	q	'glottal stop'
p	pea	w	want
b	bat	y	yard
t	tea	r	race
d	deep	ch	church
k	kick	jh	just
g	go	wh	when
f	five		

Tabelul 1: Alfabetul fonetic ARPAbet

Un fonem se numeşte staţionar sau continuu în funcţie de configuraţia tractului vocal pe durata pronunţării. Un fonem va fi necontinuu dacă este necesară schimbarea configuraţiei tractului vocal. Tabelul 2 prezintă acest tip de clasificare.

Continue		Necontinue	
Vocale	iy, ih,e y, eh, ae, er, ax, ah, uw, uh, ow, ao, aa	Vocale (diftongi)	ay, aw, oy
Semivocale	hh	Semivocale	r, l, w, y
Fricative	v, dh, z, zh, f, th, s, sh	Plozive	b, d, g, p, t, k
Africate	jh, ch		
Nazale	m, n, nx		

Tabelul 2: Clasificarea fonemelor limbii engleze (după [26])

Vocalele, fricativele, africatele şi nazalele aparţin categoriei de foneme continue, iar diftongii, semivocalele şi plozivele formează clasa fonemelor necontinue. Vocalele se află printre fonemele cu cele mai mari amplitudini. Durata vocalelor este variabilă, în mod tipic între 40 şi 400 milisecunde. În funcţie de poziţia limbii pe durata rostirii, fonemele se pot clasifica în vocale faţă (/iy/, /ih/, /ey/, /eh/, /ae/), centrale (/er/, /ax/, /ah/) şi spate (/uw/,/uh/,/ow/, /ao/,/aa/).

Formele de undă ale vocalelor ne arată şi structura cvaziperiodică ale acestora. Vocalele se disting de celelalte foneme prin locaţia formanţilor. De obicei primii trei formanţi sunt suficienţi. Pentru o voce bărbătească formanţii apar în jurul frecvenţelor 500, 1500, 2500, 3500 Hz. Primii doi formanţi F_1 şi F_2 sunt definiţi de forma tractului vocal, care este specific fiecărui vorbitor în parte. F_3 este specific doar câtorva sunete. Începând cu formantul F_4 locul acestor formanţi nu este influenţat de articulaţiile produse în timpul vorbirii.

Vocalele sunt sunete sonore, teoretic având forma tractului vocal constantă pe durata pronunţării. O definiţie rezonabilă pentru un diftong ar putea fi următoarea: o silabă vocalică conţinând două vocale. Totuşi un diftong nu este pronunţia consecutivă a două vocale, deoarece prima vocală se pronunţă mai lung decât cea de-a doua vocală. Diftongii universal acceptaţi ar fi următoarele: /ay/ - pie ; /aw/ - out; /oy/ - toy. Deci sunetele vocalice, care nu pot fi pronunţate fără mişcări articulatorii se numesc diftongi.

Semivocalele sunt similare cu vocalele. În limba engleză avem următoarele semivocale: /l/ - lawn; /r/ - run; /w/-wet; /y/-yam.

Pronunţarea consoanelor prezintă mult mai multe constrângeri, decât cea a vocalelor. Anumite consoane pot necesita mişcări articulatorii foarte precise. Sunetele, a căror pronunţie necesită trecerea fluxului de aer prin cavitatea orală

şi suspendarea totală a fluxului pentru o perioadă scurtă, se numesc consoane întrerupte sau plozive.

Fricativele sunt produse prin excitarea tractului vocal cu un flux de aer constant, care devine turbulent într-un anumit punct. Constrângerea în tractul vocal va rezulta o excitaţie nesonoră. Totuşi anumite fricative pot avea şi o componentă sonoră, caz în care obţinem excitaţie mixtă. Astfel fricativele, care nu au această componentă sonoră, se numesc fricative nesonore, iar cele care au excitaţie mixtă, se numesc fricative sonore. Fricativele sonore sunt următoarele: /f/ - free; /th/ - thick; /s/ - cease; /sh/ - mesh. Fricativele nesonore corespunzătoare celor sonore sunt următoarele: /v/ - vice; /dh/- then; /z/ - zephyr; /zh/ - measure.

În mod similar cu diftongii şi africatele se formează prin trecerea de la o plozivă spre o fricativă. În limba engleză sunt două africate: /ch/ - change; /jh/ -jam.

Africata nesonoră /ch/ se produce prin pronunţarea plozivei nesonore /t/, urmată de o tranziţie spre o fricativă nesonoră /sh/. Africata sonoră /jh/ se produce prin pronunţarea plozivei sonore /d/, urmată de fricativa sonoră /zh/.

Plozivele /b/, /d/, /g/, /p/, /t/, /k/ se produc prin acumularea unei presiuni undeva în tractul vocal şi eliberarea bruscă a acesteia printre buze. În cazul plozivelor nesonore /p/, /t/, /k/ eliberarea bruscă a presiunii este urmată de o fricaţie nesonoră. Această fricaţie nesonoră este urmată de o aspiraţie. Plozivele nesonore sunt caracterizate prin perioade mai lungi de fricaţie. Intervalul de timp până la eliberarea presiunii se numeşte intervalul de închidere.

Plozivele sonore sunt similare plozivelor nesonore, cu diferenţa că acestea din urmă includ şi vibraţia corzii vocale. Caracteristicile plozivelor variază în funcţie de poziţia lor în cuvânt sau propoziţie. Intervalul de închidere poate să lipsească, dacă apare la sfârşitul silabelor. Plozivele se modifică şi în cazul, când apar între vocale. Deoarece plozivele sunt foneme tranzitorii, proprietăţile lor sunt extrem de puternic influenţate de fonemele care le preced, respectiv le urmează. Tocmai din această cauză forma de undă nu este suficientă pentru distingerea lor. Spectrogramele ne oferă mai multe caracteristici, prin care putem distinge plozivele.

Consoanele nazale /m/, /n/, /nx/ sunt produse prin excitarea cu o undă glotală a cavităţii nazale deschise şi a cavităţii orale închise. Această categorie de foneme prezintă anumite similarităţi cu vocalele, dar sunt caracterizate de un nivel energetic mai scăzut datorită faptului că, cavitatea nazală nu poate emite sunetul la fel de puternic ca şi cavitatea orală. În funcţie de locul unde se produce închiderea cavităţii orale distingem 3 tipuri de nazale: /m/ - more (nazal frontal); /n/ - noon (nazal central); /nx/ - sing (nazal posterior)

Cu toate că nazalele sunt singurele care incorporează cavitatea nazală în scopul producerii structurii lor de frecvenţe rezonante, există şi alte foneme care devin nazalizate - fonemele care le preced sau urmează după nazale. În primul rând categoria vocalelor este cea mai afectată de acest fenomen. Nazalizarea produce foneme cu o lăţime de bandă mai largă a primului formant (F_1) [39]. Gradul în care nazalizarea afectează spectrul, depinde de modul de cuplare între cele două cavităţi.

2.3 Extragerea caracteristicilor

Din semnalul vocal putem extrage atât caracteristici acustice de nivel scăzut cum ar fi ampliitudinea spectrală cât şi caracteristici de nivel înalt cum ar fi accentul, dialectul, stilul de vorbire sau modul de pronunţare a unor cuvinte specifice. Caracteristicile de nivel înalt sunt utilizate de oameni pentru recunoaşterea vorbitorului în cauză. Extragerea caracteristicilor de nivel înalt sunt greu de cuantificat, de aceea majoritatea sistemelor pentru recunoaşterea vorbitorilor utilizează doar caracteristicile de nivel scăzut. În continuare prezentăm extragerea caracteristicilor care se utilizează atât în sisteme de recunoaşterea vorbirii cât şi în sisteme de recunoaşterea vorbitorilor.

Extragerea caracteristicilor se realizează în următorii paşi: preprocesare, analiză cepstrală şi postprocesare.

2.3.1 Preprocesare

Preprocesarea are ca scop amplificarea frecvenţelor înalte din spectru. Acesta se realizează prin utilizarea unui filtru de ordin I. (FIR), care are următoarea formă

$$s_p(n) = s(n) - as(n-1) \tag{2.1}$$

unde a se numeşte coeficient de preaccentuare şi are valori cuprinse între $0.9 \leq a \leq 1$. În sistemul nostru această valoarea utilizată este 0.97. Semnalul vocal nu este un semnal staţionar, dar pe perioade scurte de timp este cvazistaţionar. Ferestruirea semnalului are ca scop descompunerea semnalului pe bucăţi pe care acesta este aproape staţionar. Lungimea ferestrei este de obicei între 10 şi 30 de milisecunde. Pentru recunoaşterea vorbitorului am utilizat ferestre de 32 de milisecunde iar pentru recunoaşterea vorbirii ferestre de 20 de milisecunde. Aceste ferestre se aplică la fiecare 10 milisecunde până la sfârşitul semnalului. Rezultatul ferestruirii este obţinerea unei secvenţe de bucăţi de semnal numite şi cadre. Cea

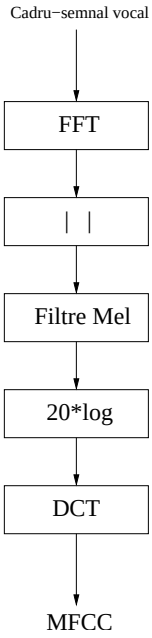


Figura 2.3: Reprezentarea modulară a extragerii coeficienţilor Mel cepstrali

mai utilizată funcţie de fereastră este fereastra de tip Hamming dată de ecuaţia următoare

$$w(n) = \begin{cases} 0.54 - 0.46\cos(\frac{2\pi n}{N-1}) & n = 0, 1, \ldots, N-1 \\ 0 & altfel \end{cases} \tag{2.2}$$

unde N este numărul eşantioanelor din cadru.

2.3.2 Analiză cepstrală

2.3.2.1 Coeficienţii Mel-cepstrali (MFCC)

Figura 2.3 ne arată paşii necesari pentru extragerea coeficienţilor Mel cepstrali.

După faza de preprocesare pentru fiecare cadru se aplică transformata Fourier rapidă (FFT) pe un număr de puncte N', putere a lui 2 şi $N' \geq N$. Din spectrul de putere obţinut se utilizează doar jumătate, deoarece spectrul este simetric. Spectrul astfel obţinut trebuie netezit pe de o parte fiindcă nu ne interesează fluctuaţile mici ale spectrului, pe de altă parte din considerente de reducere a dimensionalităţii. Pentru a realiza această netezire şi a obţine anvelopa spectrală, se aplică un banc de filtre asupra spectrului. Un banc de filtre este o serie de filtre trece-bandă cu care se multiplică spectrul, unul după altul, pentru a obţine câte

o valoare medie pentru fiecare bandă de frecvenţă. Un banc de filtre se defineşte prin forma filtrelor precum şi prin aşezarea acestora pe scara de frecvenţe. În practică aşezarea acestor filtre se face pe baza scării Mel, scară determinată a.î. să fie cât mai apropiat de modul de percepere ale frecvenţelor de către urechea umană. Localizarea frecvenţelor centrale este dată de ecuaţia

$$f_{MEL} = 1000 \cdot \frac{\log(1 + f_{LIN}/1000)}{\log 2} \tag{2.3}$$

În final se consideră logaritmul acestei anvelope spectrale şi se multiplică cu 20 pentru a obţine anvelopa spectrală în decibeli (dB).

Ultima transformare aplicată este transformata cosinus discretă (DCT).

$$c_n = \sum_{k=1}^{K} S_k \cdot \cos\left[n\left(k - \frac{1}{2}\right)\frac{\pi}{K}\right], \quad n = 0, 1, 2, \ldots, D,$$

unde K este numărul filtrelor din bancul de filtre, S_k sunt coeficienţii spectrali obţinuţi după aplicarea filtrelor, iar D este numărul coeficienţilor cepstrali calculaţi ($D \leq K$).

2.3.2.2 Coeficienţii cepstrali de predicţie liniară (LPCC)

Analiza prin predicţie liniară (LPC) este bazată pe un model liniar de producerea vorbirii. Este un model de autoregresie care este descrisă foarte detaliat în cărţile [26, 67].

Producerea vorbirii poate fi descrisă prin combinarea următoarelor patru elemente:

- sursa glotală - văzută ca un tren de impulsuri pentru sunete sonore sau ca un zgomot alb pentru sunete nesonore
- tractul vocal
- cavitatea nazală
- buzele

Fiecare dintre acestea poate fi reprezentat printr-un filtru, un filtru trece-jos pentru sursa glotală, un filtru autoregresiv pentru tractul vocal, un filtru ARMA (Auto Regressive Moving Average) pentru tractul nazal şi un filtru (MA -moving average) pentru buze. În final tot sistemul de producere a vorbirii poate fi privit ca

un filtru ARMA, iar modelele uzuale modelează vorbirea cu o variantă simplificată a filtrului ARMA, cu un filtru AR.

Principiul predicţiei liniare este estimarea parametrului unui filtru AR pe o porţiune ferestruită a unui semnal vocal. Fereastra este mutată periodic şi se recalculează parametrii filtrului pe următorul segment vocal. Pentru fiecare fereastră coeficienţii estimaţi se numesc coeficienţi de predicţie liniară (LPC). Coeficienţii cepstrali se calculează în mod direct din coeficienţii de predicţie liniară folosind următoarea formulă

$$\begin{cases} c_0 = \ln \sigma^2 \\ c_m = a_m + \sum_{k=1}^{m-1} \left(\frac{k}{m}\right) c_k a_{m-k}, \quad 1 \leq m \leq p \\ c_m = \sum_{k=1}^{m-1} \left(\frac{k}{m}\right) c_k a_{m-k}, \quad p < m \end{cases} \tag{2.4}$$

unde σ^2 este termenul de câştig din modelul LPC, a_m sunt coeficienţii de predicţie liniară iar p este ordinea analizei de predicţie liniară.

2.3.3 Postprocesare

În faza de postprocesare se poate scădea din fiecare vector cepstral vectorul medie a vectorilor cepstrali (CMS). Această fază se aplică obligatoriu în cazul când înregistrările au fost făcute de pe linie telefonică.

Pentru a încorpora şi variaţia coeficienţilor de-a lungul timpului, se calculează derivatele de ordin întâi şi doi. În mod clasic aceşti parametri se numesc coeficienţi Δ şi $\Delta\Delta$ şi se calculează cu ajutorul următoarei formule

$$\Delta C_m(t) = \frac{\sum_{k=-K}^{K} k C_m(t+k)}{\sum_{k=-K}^{K} k^2} \tag{2.5}$$

K este numărul de vectori vecini care se consideră, în aplicaţie am utilizat K=2.

2.4 Tehnici de grupare

2.4.1 Introducere

În multe aplicaţii de recunoaşterea formelor este foarte dificilă sau costisitoare, chiar imposibilă etichetarea datelor de antrenare cu eticheta categoriei corespunzătoare. Clasificarea nesupervizată se referă la situaţii în care obiectivul este

determinarea regiunilor de decizie folosind date neetichetate. Etichetele categoriilor şi alte informaţii legate de sursa datelor influenţează doar interpretarea rezultatelor grupării nu şi formarea acestora.

Clasificarea nesupervizată sau gruparea este o problemă foarte dificilă, deoarece datele pot forma grupe cu mărimi şi forme diferite. Mai mult, şi numărul grupelor depinde de rezoluţia, cu care vizualizăm datele. În literatură au fost propuse următoarele definiţii pentru noţiunea de grupă:

- formele care aparţin unei grupe sunt mai aproapiate de formele din aceeaşi grupă şi mai îndepărtate de formele aflate în alte grupe (*grupe omogene*)
- grupele sunt formate din noruri de puncte cu o densitate ridicată şi sunt separate de alte grupe prin puncte izolate sau zone având o densitate scăzută (*grupele sunt bine separate*)

Simplitatea, consistenţa şi naturaleţea cu care algoritmii de grupare organizează datele au condus la utilizarea acestor tehnici în aplicaţii ca mineritul datelor (data mining), regăsirea informaţiei, segmentarea imaginilor, compresia şi codarea semnalelor şi machine learning. O prezentare amănunţită a algoritmilor de grupare găsim în cărţile [29, 30, 35]. Majoritatea algoritmilor se bazează pe două tehnici de bază populare:

- grupare partiţională iterativă bazată pe eroarea pătratică (square error partitional algorithm)
- grupare ierarhică aglomerativă

Tehnicile ierarhice organizează datele în grupe suprapuse, care pot fi afişate în forma unei dendograme sau arbori. Cele partiţionale bazate pe eroarea pătratică încearcă să găsească acea partiţie, care minimizează dispersia intra-grupă şi maximizează dispersia inter-grupe. Algoritmul, care îşi propune găsirea soluţiei optime, trebuie să examineze toate partiţiile posibile, care se pot obţine prin gruparea a n vectori d-dimensionale în K grupe. Un asemenea algoritm ar avea o complexitate exponenţială, total inutilizabil în practică. Pentru evitarea complexităţii exponenţiale au fost propuse o serie de euristici care reduc timpul de căutare, negarantând însă optimalitatea soluţiei.

Tehnicile de grupare partiţionale se utilizează mai frecvent, decât cele ierarhice, de aceea vom începe prezentarea cu această tehnică de grupare. Studiile recente au adus la iveală următoarele concluzii:

- orice algoritm de grupare va produce un set de grupe ca şi rezultat, indiferent dacă în date există sau nu tendinţă de grupare
- nu există algoritmul „cel mai bun" de grupare; se recomandă testarea a mai multor algoritmi de grupare
- extragerea caracteristicilor dintr-o colecţie de date precum şi tehnicile de normalizare sunt la fel de importatante ca şi alegerea strategiei de grupare

Gruparea partiţională se poate formula în modul următor [41]:

Se dau n puncte într-un spaţiu d-dimensional. Să se determine gruparea acestor puncte în K grupe astfel, încât punctele aparţinând aceleiaşi grupe să fie mai apropiate de restul punctelor din aceeaşi grupă, decât de restul punctelor din afara grupei. Specificarea numărului de grupe este opţională, însă este necesară adoptarea unui criteriu de grupare. Criteriul de grupare poate fi un criteriu global sau unul local. Un criteriu global, cum ar fi eroarea minimă pătratică, reprezintă grupele cu câte un vector prototip şi clasifică punctele pe baza similarităţii cu aceste prototipuri. Un criteriu local formează grupele prin identificarea regiunilor de densitate mare sau prin adăugarea punctelor împreună cu cei k-vecini cei mai apropiaţi la aceaşi grupă.

Tabelul 2.1 prezintă algoritmii cei mai importanţi de grupare. La alegerea unui algoritm de grupare trebuie să considerăm următoarele aspecte:

1. Ce defineşte o grupă? Există un prototip reprezentativ pentru fiecare grupă?
2. Cum definim apartenenţa la o grupă? Ce metrică utilizăm?
3. Câte grupe există? Este necesară fixarea numărului de grupe?
4. Care este capacitatea de generalizare a modelului obţinut? Prin ce metodă se determină apartenenţa unui punct nou la grupele deja existente?

2.4.2 Algoritmul K-means

Se dau punctele $X = \{x_1, x_2, \ldots, x_n\}$, unde $x_i \in R^d$ formeză datele de intrare ale algoritmului de grupare. Obiectivul este să se găsească acea partiţie, care pentru un K fixat (număr de partiţii) minimizează eroarea pătratică. Notăm grupele cu $\{C_1, C_2, \ldots C_K\}$, fiecare având n_k puncte, a. î. fiecare x_i să aparţină unei singure grupe $\Sigma_{k=1}^{K} n_k = n$.

Algoritm	Descriere	Observaţii
K-means	Identifică grupele hipersferice; Grupele hiperelipsoidale se pot determina prin utilizarea distanţei Mahalanobis	K se fixează dinainte; se fixează şi centrele iniţiale ale grupelor
Fuzzy K-means	Similar cu K-means Fiecărui punct i se ataşează un grad de apartenenţă	Se fixează: K, centrele iniţiale ale grupelor şi funcţia de apartenenţă
Arborele de acoperire minim	Grupele se formează prin ştergerea muchiilor inconsistente	Se fixează definiţia muchiei inconsistente
Mutual Neighbourhood	Se calculează valoarea mutuală de vecinătate pentu fiecare pereche de puncte	Trebuie specificată adâncimea vecinătăţii
Single-Link	Un algoritm ierarhic de grupare	Detalii: 2.4.7
Complete-Link	Un algoritm ierarhic de grupare	Detalii: 2.4.7
Decompoziţie în mixturi	Se consideră că punctele sunt generate de K distribuţii	K se fixează, însă se pot defini criterii pentru estimarea acestui parametru

Tabela 2.1: Algoritmi de grupare (După [41])

Algoritm 1 Algoritmul K-means

Pas 1. Se selectează o partiţe iniţială cu K grupe
Pas 2. Se generează o nouă partiţie ataşând fiecare punct x_i grupei celei mai apropiate
Pas 3. Se recalculează centrele de greutate ale grupelor cu formula 2.6.
Pas 4. Se repetă Pas 2 şi Pas 3. până când se atinge valoarea optimală a funcţiei criteriu

Centrul de greutate al grupei C_k se defineşte prin

$$\mu^{(k)} = \left(\frac{1}{n_k} \sum_{i=1}^{n_k} x_i^{(k)} \right) \tag{2.6}$$

unde $x_i^{(k)}$ este al i-lea punct aparţinând grupei C_k. Definim eroarea pătratică pentru C_k în modul următor

$$e_k^2 = \sum_{i=1}^{n_k} d^2 \left(x_i^{(k)}, \mu^{(k)} \right) \tag{2.7}$$

unde $d\left(x_i^{(k)},\ \mu^{(k)}\right)$ este o funcţie distanţă. Această eroare se numeşte şi dispersia intra-grupă. Pentru toate grupele eroarea se defineşte prin

$$E_K^2 = \sum_{k=1}^{K} e_k^2. \tag{2.8}$$

Obiectivul grupării este găsirea acelei partiţii conţinând K grupe, pentru care 2.8 este minimă.

Algoritmul K-means se poate formula în modul arătat în (1).

Pentru oprirea algoritmului (1) se pot utiliza diverse criterii, dintre care amintim următoarele:

- Nu se modifică conţinutul grupelor (stabilizare)
- S-a depăşit numărul maxim de iteraţii
- Modificarea erorii totale este sub o valoare de prag $1 - \frac{E_K^{(n+1)}}{E_K^{(n)}} < T$

Funcţia de distanţă utilizată în formula 2.7 influenţează puternic forma grupelor. Ca şi funcţie de distanţă se utilizează cel mai des distanţa Euclideană şi distanţa Euclideană ponderată definite cu formulele 2.9, respectiv 2.10. Dacă în

formula 2.10 se utilizează $W = \Sigma^{-1}$, unde Σ este matricea de covarianţă, vorbim despre distanţă Mahalanobis. Utilizând distanţa euclideană se obţin grupe hipersferice, iar cu distanţa Mahalanobis aceste grupe vor fi hiperelipsoidale.

$$d^2(x,y) = (x-y)^T(x-y), \quad x,\ y \in R^d \tag{2.9}$$

$$d^2(x,y) = (x-y)^T W\ (x-y), \quad x,\ y \in R^d \tag{2.10}$$

Algoritmul K-means formulat în (alg.1) este un algoritm foarte eficient d.p.v. computaţional şi produce rezultate surprinzător de bune dacă grupele sunt compacte, cu forme hipersferice şi bine separate. Utilizând distanţa Mahalanobis algoritmul este capabil să detecteze chiar şi grupele de forme hiperelipsoidale. Au fost multe încercări în direcţia îmbunătăţirii performanţelor algoritmului K-means. Incorporarea unei funcţii criteriu fuzzy a rezultat în algoritmul fuzzy K-means. O altă metodă frecvent utilizată este execuţia de mai multe ori a algoritmului şi reţinerea acelei soluţii pentru care funcţia criteriu este optimă [4, 45].

2.4.3 Algoritmul Fuzzy K-means

În fiecare iteraţie a algoritmului K-means clasic fiecare punct este ataşat la o singură grupă ($arg\ min_{i=1,K} d(x,\mu_i)$). Putem relaxa această condiţie definind un grad de apartenenţă (apartenenţă fuzzy) la o grupă pentru fiecare punct. La origine aceste grade de apartenenţă corespund probabilităţilor $\hat{P}(C_i|x_j,\hat{\Theta})$, unde $\hat{\Theta}$ este vectorul de parametri pentru funcţia de apartenenţă. Funcţia criteriu care trebuie optimizată devine

$$J_{fuz} = \sum_{i=1}^{K}\sum_{j=1}^{n}\left[\hat{P}(C_i|x_j,\hat{\Theta})\right]^b d^2(x_j,\mu_i) \tag{2.11}$$

unde b este un parametru liber care se utilizează pentru gradul de amestecare al grupelor. Dacă $b = 0$, ecuaţia 2.11 este echivalentă cu 2.8. Dacă $b > 1$, criteriul permite ca fiecare punct să aparţină mai multor grupe. Suma probabilităţilor de apartenenţă al fiecărui punct la diferite grupe este 1

$$\sum_{i=1}^{K}\hat{P}(C_i|x_j) = 1, \quad j = 1,\ldots,n, \tag{2.12}$$

Algoritm 2 Algoritmul Fuzzy K-means

Pas 1. Se iniţializează K, b, μ_1, $\mu_2, \ldots, \mu_K$, $P(C_i|x_j)$, $\hat{i} = 1, \ldots, K$; $j = 1, \ldots, n$; Se normalizează $\hat{P}(C_i|x_j)$ cu ecuaţia 2.12
Pas 2. Se recalculează μ_i cu ecuaţia 2.14
Pas 3. Se recalculează $P(\hat{C_i}|x_j)$ cu ecuaţia 2.15
Pas 4. Se repetă Pas 2 şi Pas 3 până când se stabilizează valorile μ_i şi $\hat{P}(C_i|x_j)$

unde din considerente de simplitate nu am notat dependenţa de $\hat{\Theta}$. Notând cu $\hat{P}_j$ probabilitatea apriori $\hat{P}(C_j)$, soluţia optimă a ecuaţiei 2.11 se obţine prin

$$\partial J_{fuz}/\partial \mu_i = 0 \quad \partial J_{fuz}/\partial \hat{P}_j = 0 \tag{2.13}$$

Acesta ne conduce la soluţiile

$$\mu_j = \frac{\sum_{j=1}^{n} \left[\hat{P}(C_i|x_j)\right]^b x_j}{\sum_{j=1}^{n} \left[\hat{P}(C_i|x_j)\right]^b} \tag{2.14}$$

şi

$$\hat{P}(C_i|x_j) = \frac{(1/d_{ij})^{1/(b-1)}}{\sum_{r=1}^{K}(1/d_{rj})^{1/(b-1)}}, \qquad d_{ij} = d^2(x_j, \mu_i) \tag{2.15}$$

În general J_{fuz} este minim când centrele grupelor μ_j sunt aproape de punctele cu un grad mare de apartenenţă la grupa j. Algoritmul fuzzy K-means este dat în (Alg. 2).

În mod clar algoritmul clasic K-means formează un caz special al algoritmului fuzzy K-means, caz în care gradul de apartenenţă se defineşte în modul următor

$$P(C_i|x_j) = \begin{cases} 1 & daca\ d(x_j, \mu_i) < d(x_j, \mu_{i'})\ \forall\, i' \neq i \\ 0 & altfel \end{cases} \tag{2.16}$$

Încorporarea probabilităţilor conduce câteodată la îmbunătăţirea convergenţei faţă de varianta clasică K-means.

2.4.4 Cuantizare vectorială

Este interesant de notat că, concepte aparent diferite folosite pentru grupări partiţionale conduc în final la acelaşi algoritm. Se poate verifica uşor că algoritmul generalizat Lloyd pentru cuantizare vectorială dezvoltat de către inginerii din domeniul comunicaţiilor este echivalent cu algoritmul K-means.

Obiectivul unui cuantizor vectorial [65] (Vector Quantisation - **VQ**) este repre-

zentarea unui set de puncte $X \subseteq R^d$ printr-un set $Y \subseteq R^d$, $|Y| = K$. Elementele mulţimii Y se numesc vectori de cod sau cuvinte de cod, iar mulţimea Y se numeşte dicţionar de vectori de cod. Un cuantizor vectorial se poate reprezenta printr-o funcţie $q: X \rightarrow Y$. Cunoaşterea lui q ne permite să obţinem o partiţionare S a mulţimii de puncte X în K partiţii $(S_i, i = 1 \ldots K)$ sau celule.

$$S_i = \{x \in X \ : \ q(x) = y_i\}, \qquad i = 1, \ldots, K. \tag{2.17}$$

Eroarea de cuantizare este valoarea definită de $d(x, q(x))$, unde d este o funcţie de distanţă. Pentru d putem considera eroarea pătratică

$$d(x, y) = \sum_{i=1}^{d} (x_i - y_i)^2 \tag{2.18}$$

În acest caz eroare pătratică medie se defineşte prin

$$MSE = D(\{Y, S\}) = \frac{1}{K} \sum_{p=1}^{K} d(x_p, q(x_p)) = \frac{1}{K} \sum_{i=1}^{K} D_i \tag{2.19}$$

unde D_i este distorsiunea totală pentru celula i.

$$D_i = \sum_{n: x_n \in S_i} d(x_n, y_i).$$

Obiectivul cuantizării este determinarea unui dicţionar de vectori de cod pentru care formula 2.19 are valoare minimă. Acest lucru este practic echivalent cu minimizarea ecuaţiei 2.8 în cazul algoritmului K-means. În cazul algoritmului VQ, la intrarea algoritmului trebuie furnizat numărul vectorilor de cod K. Versiunea supervizată a algoritmului VQ se numeşte LVQ [49, 79] (Learning Vector Quantisation).

După ce am prezentat ideea de bază a algoritmilor de grupare, precum şi cea a cuantizării, putem trage concluzia că acestea sunt concepte similare. Singura deosebire ar fi că, un algoritm de grupare are ca şi scop identificarea grupărilor într-o mulţime de date multidimensionale, iar algoritmul de cuantizare nu are ca şi scop principal această identificare, ci reprezentarea mulţimii de date printr-o mulţime restrânsă de date. Practic ambii algoritmi ne pot conduce la acelaşi rezultat în cazul când minimizează aceeaşi funcţie criteriu. Diferenţa dintre algoritmul de grupare şi VQ este arătată pe figura 2.4. Imaginea din colţul stânga-sus reprezintă mulţimea originală de date. Următoare imagine, din colţul dreapta-sus, ne

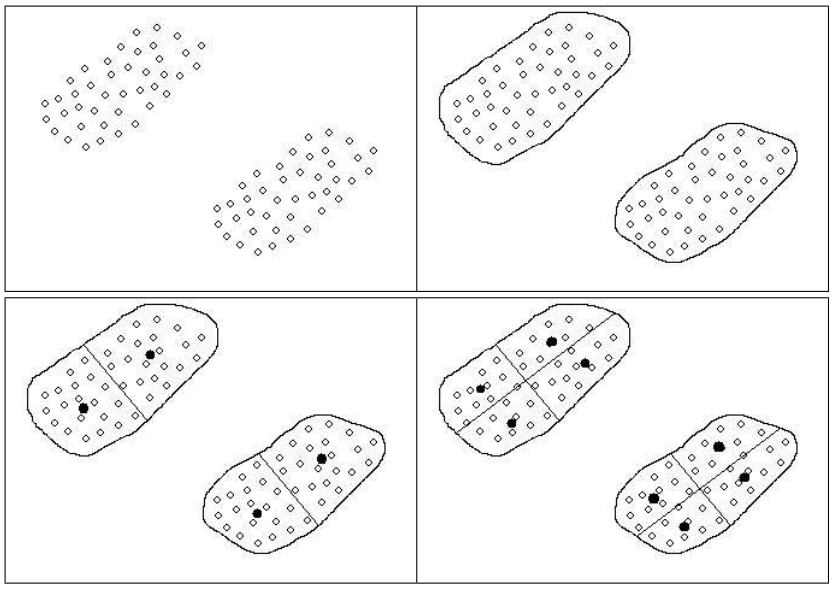


Figura 2.4: Algoritm de grupare şi VQ (după [65])

arată cele două grupe găsite de un algoritm de grupare. A treia şi a patra imagine ne arată rezultatul unui algoritm VQ cu 4 vectori de cod respectiv cu 8 vectori de cod.

2.4.5 Algoritmul LVQ

LVQ este un algoritm adaptiv de clasificare bazat pe datele de antrenare şi informaţii despre clasele acestora [42, 49]. Cu toate că aparţine algoritmilor de antrenare supervizate, LVQ utilizează tehnici nesupervizate pentru obţinerea centroizilor grupelor. Arhitectura reţelelor neuronale LVQ este foarte asemănătoare cu cea a reţelelor neuronale competitive, cu excepţia faptului că, fiecare ieşire a reţelei este asociată cu o clasă. Figura 2.5 prezintă un exemplu, unde dimensiunea vectorului de intrare este 2, iar spaţiul vectorilor de intrare este împărţit în şase grupe făcând parte din două clase. Primele două grupe aparţin clasei 1, iar celelalte patru clasei 2. Algoritmul LVQ are doi paşi. În primul pas se utilizează un algoritm nesupervizat de grupare pentru localizarea centroizilor, fără a utiliza informaţiile de clasă. În pasul următor, utilizând informaţiile de clasă se îmbunătăţesc aceste centre de grupe. Pentru pasul întâi se poate utiliza orice algoritm de grupare nesupervizată. După ce s-au obţinut grupele, acestea vor fi etichetate cu etichetele claselor. O asemenea etichetare se poate obţine prin metoda votării, adică o grupă va fi etichetată cu clasa k, dacă majoritatea punctelor din acea

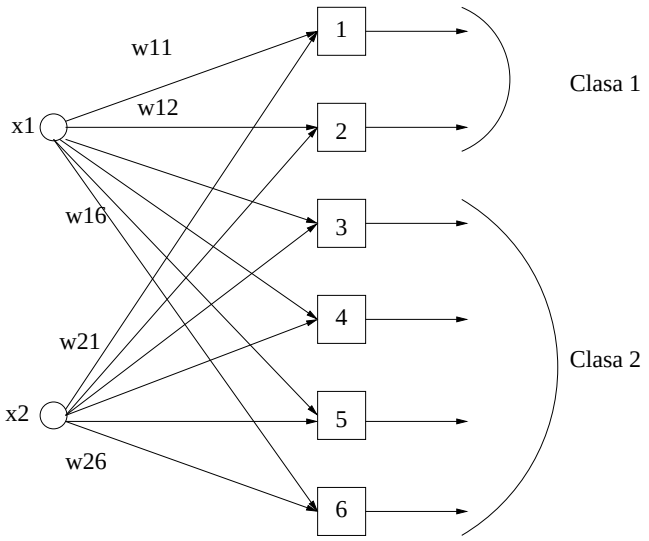


Figura 2.5: Reţea neuronală LVQ

Algoritm 3 Algoritmul LVQ-1

Pas 1. Se iniţializează centroizii cu un algoritm de grupare
Pas 2. Se etichetează grupele cu informaţiile de clasă
Pas 3. Se selectează un punct x în mod aleator şi se determină centroidul cel mai apropiat $k = \arg\min_{i=1,M} d(x, \mu_i)$, M reprezintă numărul centroizilor
Pas 4. Dacă x şi μ_i aparţin aceleiaşi clase, se modifică μ conform formulei

$$\Delta\mu_k = \eta(x - \mu_k)$$

altfel conform formulei

$$\Delta\mu_k = -\eta(x - \mu_k)$$

η este rata de învăţare, o constantă pozitivă mică care descreşte la fiecare iteraţie.

grupă aparţin clasei k. În pasul 2 centroizii sunt îmbunătăţiţi prin următoarea metodă:

- Pentru orice punct de intrare x se determină centroidul cel mai apropiat μ.
- Dacă x şi μ aparţin aceleaşi clase, centroidul μ se apropie la punctul x, altfel se depărtează.

Algoritmul $LVQ-1$ este descris în (alg. 3). $LVQ-2$ şi $LVQ-3$ sunt două îmbunătăţiri ale algoritmului, ambele încearcă să utilizeze datele de antrenare într-un mod mai eficient [49].

2.4.6 Mixturi gaussiene

Mixturile finite reprezintă un model probabilistic puternic, des utilizat în domeniul recunoaşteri formelor [41]. Mixturile sunt capabile să modeleze situaţii în care fiecare formă este generată de către o sursă, aceasta din urmă făcând parte dintr-o mulţime de surse. Mixturile pot fi privite ca o clasă de modele capabile de reprezentarea unor densităţi probabilistice complexe.

Considerăm K surse, fiecare dintre acestea fiind caracterizată de o funcţie de probabilitate (măsură sau densitate) $p_m(x|\Theta)$, parametrizată prin Θ_m, $m = 1, \ldots, K$. La generarea unei forme prima dată se alege o sursă cu probabilitatea $\{w_1,\ w_2, \ldots, w_K\}$ din cele K surse şi o observaţie generată de sursa respectivă cu probabilitatea $p_m(x|\Theta_m)$. Variabila aleatoare definită de aceast mecanism de generare este caracterizată de următoarea funcţie de probabilitate

$$p(x|\Theta_{(K)}) = \sum_{m=1}^{K} w_m p_m(x|\Theta_m) \tag{2.20}$$

unde fiecare $p_m(y|\Theta_m)$ se numeşte o componentă iar

$$\Theta_{(K)} = \{\Theta_1,\ \Theta_2, \ldots, \Theta_K,\ w_1,\ w_2, \ldots, w_K\}. \tag{2.21}$$

De obicei se presupune că, toate componentele au forme funcţionale identice, de exemplu sunt gaussiene. Potrivirea unui model cu mixturi la un set de observaţii $X = \{x_1,\ x_2, \ldots, x_n\}$ înseamnă estimarea acelor parametri, care descriu cel mai potrivit datele. Cu toate că se pot construi mixturi din mai multe tipuri de componente, în majoritatea literaturii de specialitate sunt descrise mixturile gaussiene (GMM-Gaussian Mixture Model).

În continuare prezentăm mai detaliat metoda mixturilor gaussiene. Această metodă este utilizată atât pentru modelarea vorbitorilor în cazul sistemelor de recunoaşterea vorbitorilor [16, 36, 55, 70] cât şi pentru estimarea densităţilor asociate stărilor unui model Markov ascuns.

Cele mai bune rezultate în verificarea vorbitorului prin metode dependente de text s-au obţinut cu modelul GMM. Metoda s-a dovedit a fi metoda statistică cea mai robustă în acest sens. În acest caz distribuţia vectorilor spectrali extraşi dintr-un semnal vocal specific unui vorbitor este modelată printr-o mixtură gaussiană. Funcţia de probabilitate a unei mixturi gaussiene putem defini prin

$$p(x|\Theta_{(K)}) = \sum_{i=1}^{K} w_i b_i(x|\Theta_i) \tag{2.22}$$

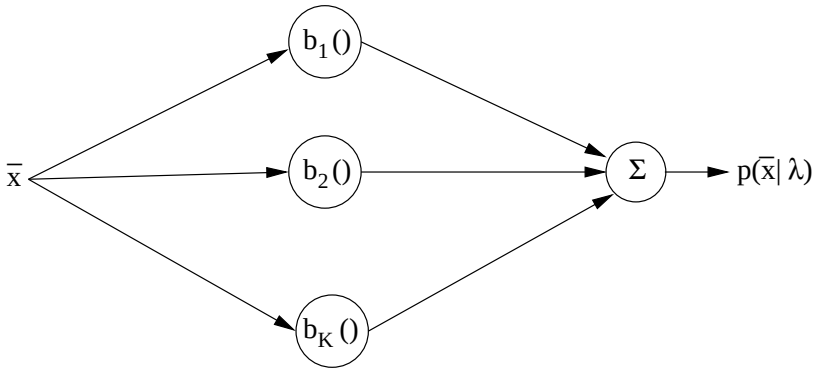


Figura 2.6: Calculul funcţiei de probabilitate

unde fiecare termen b_i este o funcţie gaussiană de d variabile $\left(x \in R^d\right)$ cu parametri $\Theta_i = (\mu_i, \Sigma_i)$ de forma:

$$b_i(x) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}}\, e^{-\frac{1}{2}(x-\mu_i)^T \sum_i^{-1}(x-\mu_i)} \tag{2.23}$$

unde μ_i şi Σ_i reprezintă valoarea medie, respectiv matricea de covarianţă a densităţii de ordin i. Calculul funcţiei de probabilitate a mixturii este prezentată de figura 2.6. Ponderile satisfac constrângerile $\sum_{i=1}^{K} w_i = 1$, care garantează că 2.25 este o funcţie de densitate adevărată. Matricea de covarianţă $\sum_i$ poate fi completă sau diagonală. Pentru a reduce numărul parametrilor, se poate utiliza o singură matrice de covarianţă comună pentru toate componentele densităţii, numită şi matrice de covarianţă globală (care la rândul ei poate fi şi diagonală). Aceste constrângeri se utilizează pentru a limita numărul parametrilor liberi care trebuie estimaţi din datele de antrenare. Astfel o mixtură gaussiană este parametrizată prin vectori de medie, matrici de covarianţă şi ponderile mixturii:

$$\lambda = \{w_i,\ \mu_i,\ \Sigma_i\} \qquad i = 1, \ldots, K. \tag{2.24}$$

Există două avantaje majore ale aplicării metodei GMM în aplicaţii de modelare a vorbirii. Primul avantaj constă în posibilitatea de a reprezenta prin densităţile componentelor o mulţime de clase acustice. Aceste clase acustice vor reprezenta diferite configuraţii de tract vocal care sunt utile atât în caracterizarea identităţii vorbitorilor cât şi pentru caracterizarea unităţilor de vorbire. Clasa acustică de ordin i va fi reprezentată prin media μ_i şi matricea de covarianţă Σ_i.

Al doilea avantaj al utilizării metodei este observaţia empirică că, o combinaţie

liniară de funcţii de bază gaussiene poate reprezenta o clasă largă de distribuţii. Una dintre caracteristicile cele mai puternice ale metodei este capabilitatea acesteia de a forma aproximări netede pentru densităţi arbitrare.

Estimarea parametrilor modelului Există multe tehnici pentru estimarea parametrilor modelului. Cea mai populară metodă rămâne estimarea conform criteriului probabilităţii maxime (Maximum Likelihood-ML). Dându-se datele de antrenare, obiectivul este de a estima parametrii modelului 2.24, adică găsirea parametrilor care maximizează probabilitatea datelor de antrenare. Probabilitatea unei secvenţe de n vectori de antrenare $X = \{x_{1,}, x_2, \ldots, x_n\}$ se poate calcula cu formula 2.25 în cazul în care datele de antrenare sunt observaţii independente.

$$p(X|\lambda) = \prod_{i=1}^{n} p(x_i|\lambda) \tag{2.25}$$

Din considerente computaţionale, în loc de a calcula $p(X|\lambda)$, se calculează logaritmul acestuia

$$\log P(X|\lambda) = \sum_{i=1}^{n} \log\ p(x_i|\lambda). \tag{2.26}$$

Utilizând metoda multiplicatorilor Lagrange, funcţia criteriu care trebuie maximizată devine

$$J = \sum_{j=1}^{n} \log p(x_j|\lambda) - \mu(\sum_{i=1}^{K} w_i - 1),$$

unde μ este multiplicatorul Lagrange. Calculând derivatele parţiale $\frac{\partial J}{\partial w_i}$, $\frac{\partial J}{\partial \mu_i}$, $\frac{\partial J}{\partial \Sigma_i}$ şi egalându-le cu zero, obţinem următoarele formule:

$$w_i = \frac{1}{n} \sum_{j=1}^{n} q_i(x_j) \tag{2.27}$$

$$\mu_i = \frac{\sum_{j=1}^{n} q_i(x_j) x_j}{\sum_{j=1}^{n} q_i(x_j)} \tag{2.28}$$

$$\Sigma_i = \frac{\sum_{j=1}^{n} q_i(x_j)(x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{n} q_i(x_j)}, \tag{2.29}$$

unde

$$q_i(x) = \frac{w_i b_i(x)}{\sum_{j=1}^{K} w_j b_j(x)} \tag{2.30}$$

este probabilitatea aposteriori a componentei de ordin i.

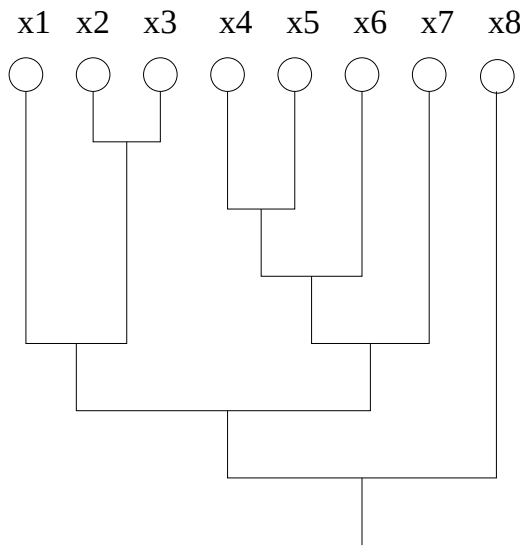


Figura 2.7: Dendogramă reprezentând gruparea datelor

2.4.7 Grupare ierarhică

În algoritmii de grupare prezentaţi înainte, grupele create de algoritmi au fost netede, în sensul că algoritmii erau incapabili să detecteze cazurile în care anumite grupe erau formate din alte subgrupe. Totuşi există mulţimi de date în care datele au o structură ierarhică, grupele fiind formate din alte subgrupe.

O ierarhie de clase se poate construi în două moduri [30]:

- aglomerativ - se porneşte cu clase formate dintr-un singur obiect şi se unesc clasele apropiate până se obţine numărul de clase dorite
- diviziv - iniţial obiectele formează o singură clasă, iar clasele unui nivel se obţin prin divizarea claselor nivelului precedent

Să considerăm mulţimea de puncte $X = \{x_1,\ x_2, \ldots, x_n\}$ şi o partiţionare a acestor puncte în K grupe. Se porneşte cu n clase, fiecare conţinând exact un punct $D_i = \{x_i\},\ i = 1, \ldots n$. La fiecare pas se aleg acele două clase care sunt cele mai asemănătoare şi se unesc. Algoritmul se repetă până când se obţine numărul de clase dorite. Acest algoritm se poate reprezenta în mod grafic cu ajutorul unui arbore, care se numeşte dendogramă şi ne arată modul de grupare a punctelor (vezi figura 2.7). Această metodă de grupare este dat de algoritmul (alg. 4).

Algoritmul (alg. 4) se termină când numărul specificat de clase este atins. Pentru a măsura asemănarea între două clase se pot utiliza următoarele funcţii

Algoritm 4 Algoritmul de grupare ierarhică aglomerativă

Pas 1. Se iniţializează K-numărul claselor; Fie $K^{'} = n \quad D_i = \{x_i\},\ i = 1, \ldots, n$.
Pas 2. Se caută grupele cele mai asemănătoare D_i şi D_j şi se fuzionează, $K^{'} = K^{'} - 1$
Pas 3. Se repetă Pas 2 până când $K^{'} = K$

de distanţă:

$$d_{min}(D_i, D_j) = \min_{x \in D_i, x^{'} \in D_j} d(x, x^{'}) \tag{2.31}$$

$$d_{max}(D_i, D_j) = \max_{x \in D_i, x^{'} \in D_j} d(x, x^{'}) \tag{2.32}$$

$$d_{avg}(D_i, D_j) = \frac{1}{n_i n_j} \sum_{x \in D_i} \sum_{x^{'} \in D_j} d(x, x^{'}) \tag{2.33}$$

$$d_{mean}(D_i, D_j) = d(\mu_i, \mu_j) \tag{2.34}$$

Dacă se utilizează funcţia de distanţă definită prin 2.31, algoritmul se numeşte algoritmul de grupare pe criteriul vecinului celui mai apropiat. Dacă algoritmul se termină, când distanţa dintre clasele cele mai apropiate depăşeşte o valoare de prag, se numeşte algoritmul legăturii unice (Single Linkage Algorithm). Privim punctele, care trebuie grupate, ca nodurile unui graf. Acestea, împreună cu arcele care leagă nodurile din aceeaşi grupă D_i, formează un drum. Dacă $d_{min}(.,.)$ este utilizat pentru calculul distanţei între submulţimi, nodurile cele mai apropiate vor determina submulţimile cele mai apropiate. Reuniunea grupelor D_i cu D_j corespunde adăugării unui arc la perechea de noduri corespunzătoare celor mai apropiate submulţimi. Deoarece aceste arce care leagă grupele întotdeauna leagă două grupe distincte, graful rezultat nu conţine cicluri, adică formează un arbore. Dacă algoritmul se continuă până când toate grupele vor fi legate, arborele astfel obţinut se numeşte arborele minim de acoperire.

Dacă se utilizează funcţia $d_{max}(.,.)$ definită de ecuaţia 2.32, algoritmul se numeşte grupare cu vecinii cei mai îndepărtaţi (Farthest-Neighbor Clustering). Dacă algoritmul se termină, când distanţa dintre clasele cele mai apropiate depăşeşte o valoare de prag, se numeşte algoritmul cu legături complete (Complete Linkage Algorithm). Acest algoritm produce un graf în care toate nodurile dintr-o grupă sunt legate prin arce. Exprimându-ne în terminologia grafurilor, fiecare grupă formează un subgraf complet. Distanţa dintre grupe este determinată de perechea de noduri cele mai distante ale celor două grupe. La reuniunea a două grupe, se

Algoritm 5 Algoritmul de cuantizare vectorială binară

Pas 1. Se formează un dicţionar de vectori de cod având un singur vector de cod, centroidul tuturor punctelor
Pas 2. Se dublează numărul vectorilor de cod din dicţionar conform ecuaţiilor 2.35, 2.36
Pas 3. Se utilizează algoritmul K-means clasic pentru a obţine un dicţionar de vectori de cod optim conform noii împărţiri
Pas 4. Se repetă paşii 2 şi 3 până când se obţine dicţionarul de vectori de cod cu dimensiunea dorită.

vor adăuga arce între toate perechile de noduri ale celor două grupe.

Un algoritm ierarhic diviziv este algoritmul de cuantizare vectorială binară [26, 38, 67]. Acest algoritm este cunoscut şi sub numele de algoritmul Linde, Buzo, Gray (LBG). Acest algoritm se utilizează pentru crearea unui dicţionar de coduri, format din $K = 2^M$ vectori de cod. Se porneşte cu o singură grupă şi se împart iterativ grupele în câte două grupe. Divizarea grupelor se realizează în mod uniform conform următoarelor ecuaţii:

$$\mu_i^+ = (1+\epsilon)\mu_i \tag{2.35}$$

$$\mu_i^- = (1-\epsilon)\mu_i \tag{2.36}$$

Valoarea ϵ se alege între 0.01 şi 0.05. Pentru determinarea noilor centroizi de după împărţire, se utilizează algoritmul K-means clasic (alg. 1). O versiune mai eficientă aplică algoritmul K-means pentru fiecare grupă separat şi se reţine şi structura de arbore pentru o căutare eficientă în cadrul dicţionarului de vectori de cod.

Algoritmul 5. este un algorim eficient din punct de vedere computaţional, care produce întotdeauna acelaşi rezultat, fiind iniţializat în mod determinist. Algoritmul produce întotdeauna un dicţionar de vectori de cod având dimensiunea $K = 2^M$.

2.5 Modele Markov ascunse

Începând cu anii '80 cercetătorii din domeniul recunoaşterii vorbirii au început să aplice metode stohastice pentru modelarea vorbirii. Expresia stohastic se utilizează pentru a indica faptul că aceste modele încearcă să caracterizeze variabilitatea prezentă în pronunţarea sunetelor. Un model stohastic este modelul Markov ascuns (Hidden Markov Model - HMM), care stă la baza mai multor sis-

teme comerciale de recunoaşterea vorbirii. Aceste modele au fost introduse în domeniul recunoaşterii vorbirii în mod independent de către Baker la universitatea Carnegie-Mellon şi Jelinek la IBM în anul 1975. Vorbirea a fost modelată ca un proces stohastic dublu, în care rezultatele primului proces stohastic sunt ascunse, doar cel de-al doilea proces este observabil. Cele două procese trebuiau caracterizate prin prisma celui de-al doilea, cel care era observabil. Algoritmul, care permitea identificarea parametrilor modelului, este astăzi cunoscut sub numele de Expectation Maximisation (**EM**) şi a fost prima dată publicat de către Dempster în 1977 [27]. Baum şi colegii săi au elaborat algoritmul pentru estimarea parametrilor modelului HMM [13], care poate fi considerat un caz special al algoritmului **EM** dat de către Dempster. Acest algoritm poartă numele de algoritmul de reestimare **Baum-Welch**. Întârzierea aplicării acestor algoritmi pentru recunoaşterea vorbirii se datorează faptului că rezultatele lui Dempster precum şi rezultatele obţinute de către Baum şi colegii săi au fost publicate în reviste care sunt citite de prea puţini ingineri. Descrierea amănunţită a acestor modele o găsim în [14, 26, 29, 38, 40, 43, 44, 67, 68].

2.5.1 Elemente ale unui model Markov ascuns cu observaţii discrete

Un model Markov ascuns cu observaţii discrete este caracterizat de următoarele elemente:

N -numărul de stări ascunse ale modelului, care în anumite aplicaţii sunt asociate cu semnificaţii fizice; vom eticheta stările individuale cu $\{1, 2, ...N\}$, şi vom nota starea la momentul t cu q_t.

M -numărul de simboluri distincte, adică dimensiunea alfabetului discret generat de model; vom nota simbolurile individuale astfel: $\{v_1, v_2, ...v_M\}$.

$A = \{a_{ij}\}$−matricea probabilităţilor de tranziţie între stări, unde:

$$a_{ij} = P[q_{t+1} = j|q_t = i], \;\; 1 \leq i, j \leq N \tag{2.37}$$

În cazul special, când se poate trece din orice stare în orice altă stare într-un singur pas (cazul modelului ergodic) $a_{ij} > 0 \;\forall\, i, j$

$B = \{b_j(k)\}$−matricea probabilităţii de emisie a simbolurilor, unde:

$$b_j(k) = P[o_t = v_k|q_t = j], \;\; 1 \leq k \leq M, \tag{2.38}$$

defineşte distribuţia simbolurilor în starea j, cu $1 \leq j \leq N$.

$\pi = \{\pi_i\}$−distribuţia stării iniţiale, unde:

$$\pi_i = P[q_1 = i], \;\; 1 \leq i \leq N \tag{2.39}$$

Se poate vedea din definiţiile de mai sus că, o specificaţie completă a modelului Markov ascuns λ presupune o stabilire a parametrilor N şi M, precum şi a celor trei seturi de probabilităţi A, B, π. Un model HMM va fi notat în modul următor:

$$\lambda = (A,\ B,\ \pi) \tag{2.40}$$

2.5.2 Cele trei probleme de bază ale modelelor Markov ascunse

Pentru ca modelul Markov ascuns prezentat în secţiunea anterioară să fie utilizabil în aplicaţii reale, trebuie rezolvate trei probleme de bază. Aceste trei probleme sunt următoarele:

1. **Problema evaluării:** Dându-se secvenţa de observaţii $O = (o_1, o_2, ...o_T)$ şi un model $\lambda = (A,\ B,\ \pi)$, să se calculeze în mod eficient probabilitatea secvenţei de observaţii, condiţionată de model, $P(O|\lambda)$.

2. **Problema determinării secvenţei de stări optimale:** Dându-se secvenţa de observaţii $O = (o_1, o_2, ...o_T)$, şi un model $\lambda = (A, B, \pi)$, să se determine acea secvenţă de stări $q = (q_1, q_2, ...q_T)$, care a generat secvenţa de observaţii cu probabilitatea maximă.

3. **Problema antrenării:** Dându-se secvenţa de observaţii $O = (o_1, o_2, ...o_T)$, şi un model $\lambda = (A, B, \pi)$, să se reestimeze parametrii modelului $\lambda = (A, B, \pi)$ astfel încât $P(O|\lambda)$ să fie maxim.

Problema 1 poate fi privită ca o problemă de evaluare a probabilităţii de potrivire între un model dat şi o secvenţă de observaţii. În cazul problemelor de recunoaştere avem un set de modele şi o secvenţă de observaţii. Pentru a decide care model a generat secvenţa de observaţii, se va evalua probabilitatea secvenţei de observaţii în fiecare model, şi se va alege acel model, care a furnizat probabilitatea maximă.

Problema 2 este o problemă în care se încearcă descoperirea procesului stohastic al modelului prin determinarea succesiunii corecte de stări. De fapt, este foarte greu de găsit această secvenţă de stări aşa-numită corectă. Pentru a rezolva această problemă cât mai bine posibil, se foloseşte un criteriu de optima-

litate. Unul din algoritmii cei mai cunoscuţi pentru rezolvarea acestei probleme este algoritmul Viterbi.

Problema 3 este o problemă în care se doreşte optimizarea parametrilor modelului pentru a descrie cât mai bine o secvenţă dată de observaţii. Secvenţa de observaţii utilizată la ajustarea parametrilor modelului se numeşte secvenţă de antrenare. Problema antrenării este foarte importantă pentru cele mai multe aplicaţii ale modelelor Markov ascunse, deoarece permite adaptarea optimă a parametrilor modelului la datele de antrenare, prin reestimarea acestora. Cel mai cunoscut algoritm de reestimare a parametrilor este algoritmul Baum-Welch.

2.5.2.1 Soluţii pentru Problema 1

Dorim să calculăm probabilitatea secvenţei de observaţii, $O = (o_1, o_2, ...o_T)$ în modelul $\lambda = (A, B, \pi)$. Metoda cea mai simplă este să enumerăm toate secvenţele posibile de lungime T. Obţinem N^T secvenţe de lungime T. Considerăm secvenţa fixată:

$$q = (q_1,\ q_2, ...,\ q_T) \tag{2.41}$$

unde q_1 este starea iniţială. Probabilitatea secvenţei de observaţii O este:

$$P(O|q, \lambda) = \prod_{t=1}^{T} P(o_t|q_t, \lambda) \tag{2.42}$$

Am presupus că observaţiile sunt independente, deci:

$$P(O|q, \lambda) = b_{q_1}(o_1) \cdot b_{q_2}(o_2)...b_{q_T}(o_T) \tag{2.43}$$

Probabilitatea secvenţei q este dată de formula:

$$P(q|\lambda) = \pi_{q_1} \cdot a_{q_1 q_2} \cdot a_{q_2 q_3}...a_{q_{T-1} q_T} \tag{2.44}$$

Probabilitatea ca secvenţa q să rezulte secvenţa de observaţii O este:

$$P(O, q|\lambda) = P(O|q, \lambda) \cdot P(q|\lambda) \tag{2.45}$$

Probabilitatea secvenţei de observaţii O în modelul dat este:

$$\begin{aligned} P(O|\lambda) &= \sum_{toate\ q} P(O|q, \lambda)P(q|\lambda) = \\ &= \sum_{q_1, q_2, ...q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2)...a_{q_{T-1} q_T} b_{q_T}(o_T) \end{aligned} \tag{2.46}$$

Algoritm 6 Algoritmul forward

Pas 1. Iniţializare

$$\alpha_1(i) = \pi_i b_i(o_1),\ 1 \leq i \leq N. \tag{2.48}$$

Pas 2. Inducţie

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}),\ 1 \leq t \leq T-1,\ 1 \leq j \leq N \tag{2.49}$$

Pas 3. Terminare

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{2.50}$$

Interpretarea acestei formule este următoarea:

- Iniţial ne aflăm în starea q_1 cu probabilitatea π_{q_1} şi se generează simbolul o_1 cu probabilitatea $b_{q_1}(o_1)$.
- Se trece în starea q_2 cu probabilitatea de tranziţie $a_{q_1 q_2}$ şi se generează simbolul o_2 cu probabilitatea $b_{q_2}(o_2)$.
- Acest proces se continuă până când se face inclusiv tranziţia din starea q_{T-1} în starea q_T cu probabilitatea $a_{q_{T-1} q_T}$ generând simbolul o_T cu probabilitatea $b_{q_T}(o_T)$.

Calculul acestei probabilităţi presupune efectuarea de aproximativ $2T \cdot N^T$ de operaţii. Din cauza ineficienţei acestui algoritm se utilizează alţi algoritmi, cum ar fi algoritmul înainte (**forward procedure**).

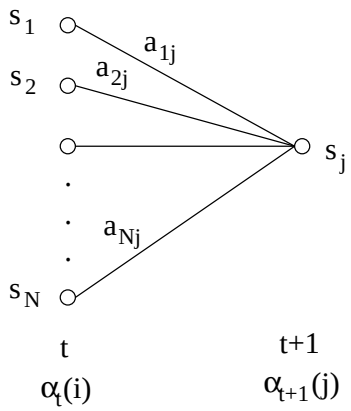
Algoritmul înainte (The Forward Procedure)

Considerăm variabila

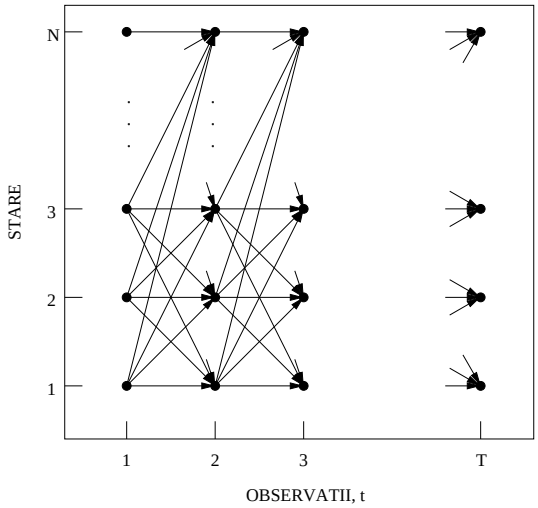
$$\alpha_t(i) = P(o_1 o_2 ... o_t, q_t = i|\lambda) \tag{2.47}$$

care este probabilitatea, ca până în momentul t să se obţină subsecvenţa de observaţii $o_1 o_2 ... o_t$, iar sistemul să fie în starea i la momentul t. Această probabilitate se poate calcula inductiv cu algoritmul 6.

Primul pas iniţializează probabilităţile de a fi în starea i şi de a avea observaţia o_1. Pasul inductiv este ilustrat în figura 2.8(a). Figura ne arată cum se poate ajunge din starea i, $1 \leq i \leq N$, la momentul t, în starea j la momentul $t+1$.



(a)



(b)

Figura 2.8: a) Secvenţe de operaţii necesare în vederea calculării variabilei $\alpha_{t+1}(j)$. b) Implementarea calculării lui $\alpha_t(i)$

Deoarece $\alpha_t(i)$ este probabilitatea secvenţei de observaţii $o_1o_2...o_t$ la momentul t, şi cu condiţia ca sistemul să fie în starea i, produsul $\alpha_t(i)a_{ij}$ va fi probabilitatea ca sistemul în momentul $t+1$ să fie în starea j. Însumând acest produs pentru toate stările posibile i, $1 \leq i \leq N$ rezultă probabilitatea de a fi în momentul $t+1$ în starea j. După acest pas se vede că $\alpha_{t+1}(j)$ se obţine prin înmulţirea valorii obţinute cu $b_j(o_{t+1})$. Calculele din formula 2.49 sunt repetate pentru toate stările j, $1 \leq j \leq N$, pentru un t fixat. După aceea se iterează pentru fiecare $t = 1, 2, ..., T-1$. În pasul 3 obţinem $P(O|\lambda)$ ca suma variabilelor $\alpha_T(i)$.

Dacă examinăm complexitatea calculului $\alpha_t(j)$, $1 \leq t \leq T$, $1 \leq j \leq N$, observăm că necesită aproximativ N^2T calcule. Mai precis sunt necesare $N(N+1)(T-1)+N$ multiplicări şi $N(N-1)(T-1)$ adunări. Comparând cu calculele directe, obţinem raportul:

$$\frac{C_{forma-directa}}{C_{forward}} = \frac{2TN^T}{N^2T} = 2N^{T-2} \tag{2.51}$$

Procedura forward este bazată pe structura de latice din figura 2.8(b).

Algoritmul înapoi (The Backward Procedure)

În mod similar cu procedura înainte, definim variabila $\beta_t(i)$ definit prin:

$$\beta_t(i) = P(o_{t+1}o_{t+2}...o_T|q_t = i, \lambda) \tag{2.52}$$

care este probabilitatea ca în momentul t sistemul să fie în starea i şi începând cu acest moment să se obţină secvenţa de observaţii $o_{t+1}o_{t+2}...o_T$. Algoritmul descris este dat de (alg. 7).

Pasul 1 defineşte în mod arbitrar, ca $\beta_T(i)$ să fie 1 pentru orice i. Pasul 2 este ilustrat şi în figura 2.9.

Pentru a calcula $\beta_t(i)$ pentru t şi i fixat, trebuie să însumăm produsele obţinute din următoarele probabilităţi:

- probabilitatea de tranziţie din starea i în starea j (a_{ij})
- probabilitatea de a obţine observaţia o_{t+1} la momentul $t+1$, sistemul fiind în starea j ($b_j(o_{t+1})$)
- probabilitatea, ca începând cu momentul $t+1$ să se obţină secvenţa de observaţii $o_{t+2}o_{t+3}...o_T$. ($\beta_{t+1}(j)$)

Algoritm 7 Algoritmul backward

Pas 1. Iniţializare

$$\beta_T(i) = 1,\ 1 \leq i \leq N \tag{2.53}$$

Pas 2. Inducţie

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j),\ t = T-1, T-2, ..., 1,\ 1 \leq i \leq N \tag{2.54}$$

Pas 3. Terminare

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i \beta_1(i) b_i(o_1) \tag{2.55}$$

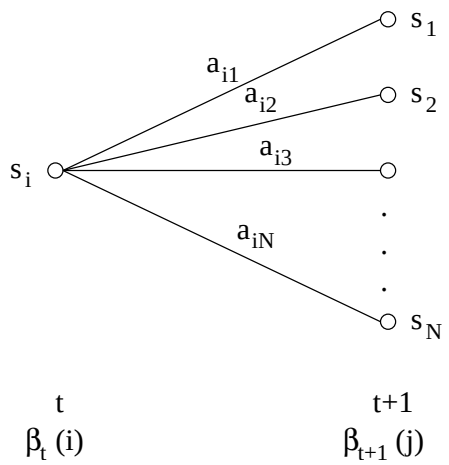


Figura 2.9: Operaţii necesare pentru calculul variabilei $\beta_t(i)$

Repetând calculele pentru $1 \leq t \leq T$ şi $1 \leq i \leq N$ algorimul va fi de complexitate $O(N^2T)$.

2.5.2.2 Soluţie pentru Problema 2

Pentru prima problemă am avut soluţii exacte. În cazul problemei 2 va fi mai dificil să dăm o soluţie exactă din cauza definiţiei de secvenţă optimă de stări, deoarece există mai multe criterii de optimalitate. Un criteriu de optimalitate ar fi să alegem cea mai probabilă stare la fiecare moment t. Acest criteriu de optimalitate maximizează numărul corect de stări. Pentru a implementa această soluţie, definim variabila de probabilitate

$$\gamma_t(i) = P(q_t = i|O, \lambda) \tag{2.56}$$

care este probabilitatea de a fi în starea i la momentul t şi producând secvenţa de observaţii O în modelul λ. Putem exprima $\gamma_t(i)$ în modul următor:

$$\begin{aligned} \gamma_t(i) &= P(q_t = i|O, \lambda) \\ &= \frac{P(O, q_t = i|\lambda)}{P(O|\lambda)} \\ &= \frac{P(O, q_t = i|\lambda)}{\sum_{i=1}^{N} P(O, q_t = i|\lambda)} \end{aligned} \tag{2.57}$$

Deoarece $P(O, q_t = i|\lambda)$ este egal cu $\alpha_t(i)\beta_t(i)$, obţinem:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{2.58}$$

Utilizând $\gamma_t(i)$ putem obţine formula pentru starea cea mai probabilă la momentul t:

$$q_t^* = \arg \min_{1 \leq i \leq N} [\gamma_t(i)],\ 1 \leq t \leq T. \tag{2.59}$$

Ecuaţia 2.59 maximizează numărul stărilor corecte selectând starea cea mai probabilă pentru fiecare moment t. Totuşi, secvenţa astfel obţinută poate fi şi una imposibilă, de exemplu dacă avem probabilităţi de tranziţie cu valoarea 0 ($a_{ij} = 0$) pentru o anumită stare i. O posibilă soluţie pentru problema de mai sus este să modificăm criteriul de optimalitate astfel, încât să căutăm perechi de stări corecte sau triplete de stări corecte. Deşi acest criteriu poate fi aplicabil pentru anumite aplicaţii, criteriul cel mai des folosit este de a găsi o singură secvenţă de stări care maximizează $P(q|O, \lambda)$, care este echivalent cu maximizarea $P(q, O|\lambda)$. O tehnică

Algoritm 8 Algoritmul lui Viterbi

Pas 1. Iniţializare

$$\delta_1(i) = \pi_i b_i(o_1),\ 1 \leq i \leq N \tag{2.62}$$

$$\psi_1(i) = 0 \tag{2.63}$$

Pas 2. Inducţie

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t),\ 2 \leq t \leq T,\ 1 \leq j \leq N \tag{2.64}$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}],\ 2 \leq t \leq T,\ 1 \leq j \leq N \tag{2.65}$$

Pas 3. Terminare

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \tag{2.66}$$

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \tag{2.67}$$

Pas 4. Reconstituire drum

$$q_t^* = \psi_{t+1}(q_{t+1}^*),\ t = T-1, T-2, ..., 1 \tag{2.68}$$

formală pentru găsirea acestei secvenţe există, se bazează pe metoda programării dinamice şi se numeşte algoritmul Viterbi.

Algoritmul Viterbi

Pentru a găsi secvenţa de stări cea mai potrivită, $q = (q_1 q_2 ... q_T)$, pentru secvenţa de observaţii $O = (o_1 o_2 ... o_T)$, trebuie să definim cantitatea

$$\delta_t(i) = \max_{q_1 q_2 ... q_{t-1}} P[q_1 q_2 ... q_{t-1}, q_t = i, o_1 o_2 ... o_t | \lambda] \tag{2.60}$$

$\delta_t(i)$ este probabilitatea maximă de-a lungul unui singur drum, la momentul t, definită pentru primele t observaţii, terminată în starea i. Prin inducţie obţinem

$$\delta_{t+1}(j) = [\max_i\ \delta_t(i) a_{ij}] \dot{b}_j(o_{t+1}) \tag{2.61}$$

Pentru a găsi această secvenţă de stări, trebuie să păstrăm argumentul care maximizează ecuaţia 2.61 pentru fiecare t şi j. Pentru acesta vom utiliza tabloul $\psi_t(j)$. Algoritmul complet este dat în (alg. 8).

Trebuie să menţionăm că algoritmul Viterbi este asemănător cu algoritmul forward (cu excepţia pasului 4). Diferenţa majoră este maximizarea din ecuaţia

Algoritm 9 Algoritmul Viterbi modificat

Pas 0. Preprocesare

$$\widetilde{\pi}_i = \log(\pi_i),\ 1 \leq i \leq N$$

$$\widetilde{b}_i(o_t) = \log[b_i(o_i)],\ 1 \leq i \leq N,\ 1 \leq t \leq T$$

$$\widetilde{a}_{ij} = \log(a_{ij})\ 1 \leq i,\ j \leq N$$

Pas 1. Iniţializare

$$\widetilde{\delta}_1(i) = \log(\delta_1(i)) = \widetilde{\pi}_i + \widetilde{b}_i(o_1),\ 1 \leq i \leq N$$

$$\psi_1(i) = 0,\ 1 \leq i \leq N$$

Pas 2. Recurenţă

$$\widetilde{\delta}_t(j) = \log(\delta_t(j)) = \max_{1 \leq i \leq N} \left[\widetilde{\delta}_{t-1}(i) + \widetilde{a}_{ij}\right] + \widetilde{b}_j(o_t)$$

$$\psi_t(j) = \arg\max_{1 \leq i \leq N} \left[\widetilde{\delta}_{t-1}(i) + \widetilde{a}_{ij}\right], \quad 2 \leq t \leq T,\ 1 \leq j \leq N$$

Pas 3. Terminare

$$\tilde{P}^* = \max_{1 \leq i \leq N} [\tilde{\delta}_T(i)]$$

$$q_t^* = \arg\max_{1 \leq i \leq N} [\tilde{\delta}_T(i)]$$

Pas 4. Reconstituire drum

$$q_t^* = \psi_{t+1}(q_{t+1}^*),\ t = T-1, T-2, ..., 1$$

2.64, care în cazul algoritmului forward este o sumă.

O alternativă pentru implementarea algoritmului Viterbi

Logaritmând parametrii modelului, algoritmul Viterbi prezentat în secţiunea precedentă poate fi implementat fără utilizarea operaţiei de înmulţire (alg. 9).

2.5.2.3 Soluţie pentru Problema 3

Estimarea parametrilor modelului $\lambda = (A,\ B,\ \pi)$ pentru a satisface un criteriu de optimalitate este cea mai sofisticată problemă. Nu există metode analitice cunoscute pentru a rezolva problema pentru seturi de parametri care maximizează probabilitatea secvenţei de observaţii. Putem dealtfel alege $\lambda = (A,\ B,\ \pi)$ astfel, încât probabilitatea $P(O|\lambda)$ să aibă un maxim local folosind o procedură iterativă,

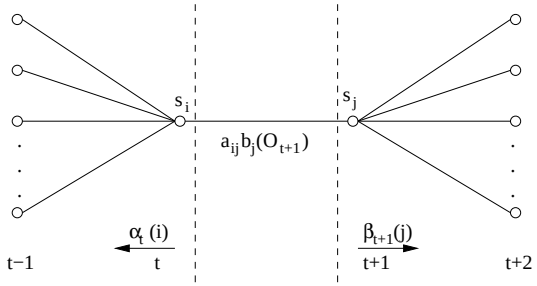


Figura 2.10: Operaţii necesare pentru calculul probabilităţii evenimentului, ca sistemul să se afle în momentul t în starea i, respectiv în momentul $t+1$ în starea j

de exemplu Baum-Welch (caz particular al algoritmului Expectation Maximization - EM) sau folosind tehnici de tip gradient. În această secţiune discutăm o procedură iterativă bazată în principal pe lucrarea clasică a lui Baum şi ai altor colegi pentru alegerea parametrilor modelului.

Definim $\xi(i,j)$, ca fiind probabilitatea, ca sistemul la momentul t să fie în starea i şi la momentul $t+1$ în starea j, dată fiind secvenţa de observaţii O şi modelul λ, ca

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda) \tag{2.69}$$

Drumurile care satisfac condiţiile cerute de ecuaţia 2.69 sunt reprezentate în fig. 2.10.

Din definiţiile variabilelor directe şi inverse putem scrie $\xi_t(i,j)$ sub forma

$$\begin{aligned} \xi_t(i,j) &= \frac{P(q_t = i, q_{t+1} = j, O|\lambda)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \end{aligned} \tag{2.70}$$

Am definit mai înainte $\gamma_t(i)$ ca şi probabilitatea, ca sistemul la momentul t să se afle în starea i, fiind dat întregul set de observaţii precum şi modelul. Astfel putem exprima $\gamma_t(i)$ prin $\xi_t(i,j)$ prin formula:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{2.71}$$

Dacă însumăm $\gamma_t(i)$ după t, obţinem o valoare $\sum_{t=1}^{T-1} \gamma_t(i)$ care poate fi interpretată ca frecvenţa trecerii prin starea i, sau echivalent, frecvenţa tranziţiilor făcute din starea i. În mod similar însumarea lui $\xi_t(i,j)$ după t, $\sum_{t=1}^{T-1} \xi_t(i,j)$, poate fi interpretată ca fiind numărul de tranziţii făcute din starea i în starea j.

Utilizând formulele de mai sus şi conceptul de numărare a evenimentelor, putem propune o metodă pentru reestimarea parametrilor modelului Markov ascuns.

$\overline{\pi_j}$ = frecvenţa cu care sistemul este în starea i la momentul $(t = 1) = \gamma_1(i)$

$\overline{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$, fiind raportul dintre numărul de tranziţii din starea i în starea j şi numărul de tranziţii din starea i.

$\overline{b_j}(k) = \frac{\sum_{t=1,o_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$, fiind raportul dintre frecvenţa evenimentului ca sistemul să fie în starea j şi să facă observaţia v_k şi frecvenţa ca sistemul să fie în starea j.

Dacă notăm modelul iniţial cu $\lambda = (A, B, \pi)$, iar modelul reestimat cu $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\pi})$, atunci s-a demonstrat de către echipa lui Baum că, ori modelul iniţial λ defineşte un punct critic al funcţiei de probabilitate, caz în care $\overline{\lambda} = \lambda$; ori $\overline{\lambda}$ este mai probabil ca λ în sensul $P(O|\overline{\lambda}) > P(O|\lambda)$; caz în care secvenţa de observaţii va fi generată cu o probabilitate mai mare de către modelul $\overline{\lambda}$.

Continuând procedura de mai sus, înlocuind la fiecare pas modelul λ cu $\overline{\lambda}$ şi repetând paşii de estimare obţinem în final estimata ML (Maximum Likelihood) al modelului Markov ascuns. Trebuie menţionat însă că prin algoritmul forward-backward obţinem doar un maxim local, iar în problemele reale funcţiile de probabilitate sunt foarte complexe cu mai multe maxime locale.

2.5.3 Topologii de modele Markov ascunse

O modalitate de clasificare a modelelor Markov ascunse este aceea, în care se ţine cont de structura matricei de tranziţie A. Până acum am discutat doar despre cazul special al modelelor ergodice, în care din fiecare stare se poate ajunge în orice altă stare (eventual chiar în aceeaşi stare), într-un număr finit de tranziţii. Un astfel de model este ilustrat în figura 2.11(a), pentru cazul particular cu 4 stări.

S-a constatat însă, că există situaţii în care proprietăţile anumitor semnale se modelează mai bine cu altfel de modele decât cel ergodic. Un astfel de model este ilustrat în figura 2.11(b). Acest model se numeşte model Bakis sau model stânga-dreapta (left-right model); numele se datorează faptului că stările se desfăşoară de la stânga la dreapta. Este evident că modelele Markov ascunse de tip stânga-dreapta pot modela cu uşurinţă semnale ale căror proprietăţi variază în timp,

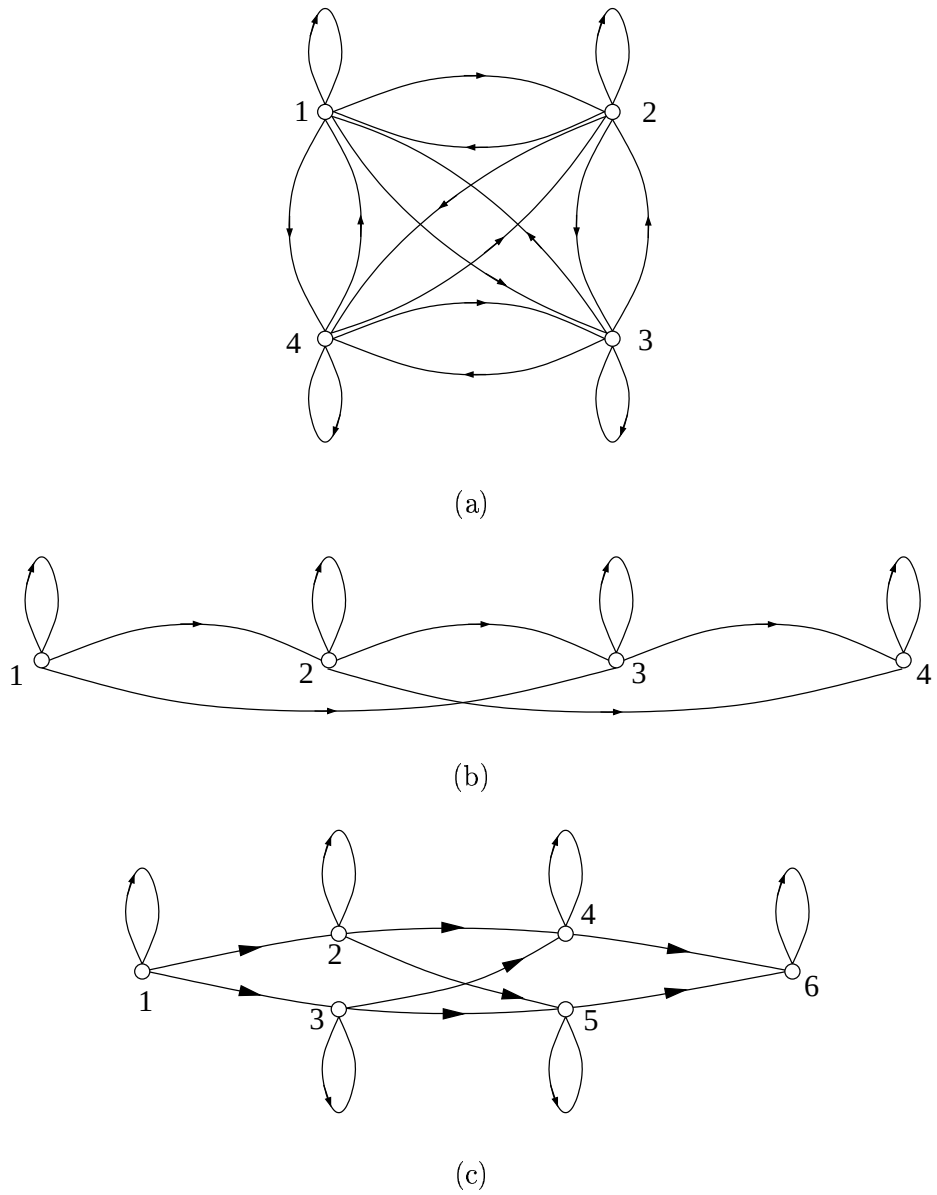


Figura 2.11: Trei tipuri diferite de MMA

ca de exemplu semnalul vocal. Proprietatea modelelor Markov ascunse de tip stânga-dreapta este aceea, că elementele matricei de tranziţie respectă condiţia:

$$a_{ij} = 0, \forall j < i, \tag{2.72}$$

ceea ce înseamnă că nu sunt permise tranziţii în stări cu indici mai mici decât cei ai stărilor curente. Mai mult, probabilităţile stărilor iniţiale au proprietatea:

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases} \tag{2.73}$$

deoarece secvenţa de stări trebuie să înceapă cu starea 1 şi să se termine în starea N.

Figura 2.11(c) ne arată încrucişarea a două modele stânga-dreapta. Prin punerea în paralel a două structuri de tip Bakis se poate obţine modelul din figură.

2.5.4 Tipuri de modele Markov ascunse

Cele mai utilizate modele Markov ascunse sunt:

- modele discrete
- modele continue
- modele semicontinue

2.5.4.1 Modele continue

În secţiunile precedente am prezentat modelele Markov ascunse de tip discret. În acest caz observaţiile apartin unei mulţimi finite. Dacă observaţiile aparţin unei mulţimi continue, trebuie adaptată descrierea modelului discret.

O observaţie în cazul modelului discret este o valoare discretă $o_t \in \{v_1,\ v_2, \ldots, v_M\}$, iar în cazul continuu $o_t \in R^d$, un vector d-dimensional.

Probabilităţile de emisie a observaţiilor în stările modelului se descrie cu $b_j(k) = P[o_t = v_k | q_t = j]$, $1 \leq k \leq M, \quad \forall\, 1 \leq j \leq N$ în cazul discret, iar în cazul continuu va fi modificat la $b_j(x) = \sum_{k=1}^{M} w_{jk} N(x,\ \mu_{jk},\ \Sigma_{jk}) = \sum_{k=1}^{M} w_{jk} b_{jk}(x)$, unde $N(x,\ \mu_{jk},\ \Sigma_{jk})$ sau $b_{jk}(x)$ este funcţia de densitate gaussiană cu valoare medie μ_{jk} şi matricea de covarianţă Σ_{jk}. M este numărul componentelor mixturii, iar w_{jk}, $k = 1, \ldots M$ sunt ponderile componentelor mixturii, care satisfac

constrângerea

$$\sum_{k=1}^{M} w_{jk} = 1.$$

Ecuaţiile de reestimare devin

$$\overline{w_{jk}} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{2.74}$$

$$\overline{\mu_{jk}} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot o_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2.75}$$

$$\overline{\Sigma_{jk}} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(o_t - \mu_{jk})(o_t - \mu_{jk})^T}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2.76}$$

unde

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}\right] \left[\frac{w_{jk} N(o_t, \mu_{jk}, \Sigma_{jk})}{\sum_{m=1}^{M} w_{jm} N(o_t, \mu_{jm}, \Sigma_{jm})}\right] \tag{2.77}$$

Interpretarea formulelor este simplă, şi anume: w_{jk} se estimează prin raportul a două valori medii, de câte ori este sistemul în starea j şi de câte ori generează o observaţie folosind componenta k a mixturii corespunzătoare stării j. Interpretări similare se pot da şi pentru μ_{jk} şi Σ_{jk}.

2.5.4.2 Modele semicontinue

Modelele semicontinue reprezintă un compromis între eficienţă şi aplicabilitate. Modelele continue au prea mulţi parametri, care prezintă probleme atât din punctul de vedere al estimării adecvate cât şi din punct de vedere computaţional, calculele fiind mult prea lente. În cazul modelelor semicontinue în loc să avem mixturi gaussiene pentru fiecare stare în parte, definim un singur set de mixturi gaussiene, cu un număr adecvat de componente, care vor fi utilizate în mod comun de toate stările. Numărul componentelor se alege astfel încât mixturile gaussiene să fie capabile pentru a reprezenta întregul spaţiu acustic.

În cazul modelelor HMM discrete se utilizează un dicţionar de vectori de cod $\{o_1, o_2, \ldots, o_M\}$ pentru discretizarea vectorilor acustici, adică pentru a cuantiza vectorii acustici în observaţii discrete. Distribuţia vectorilor acustici caracteristici

stărilor se va defini ca

$$b_j(x) = \sum_{k=1}^{M} b_j(k) f(x|o_k) = \sum_{k=1}^{M} b_j(k) N(x, \mu_k, \Sigma_k) \tag{2.78}$$

unde o_k este vectorul cod cu indicele k, $b_j(k)$ este probabilitatea de generare a observaţiei k în starea j (cazul modelelor discrete), sau ponderea în cazul modelelor continue, iar $N(x, \mu_k, \Sigma_k)$ sunt componentele mixturii gaussiene partajate de către toate stările. Dacă privim prin prisma modelelor HMM discrete, dicţionarul de vectori de cod va fi generalizat, fiecare vector cod va fi reprezentat de către o valoare medie şi o matrice de covarianţă. Cuantizarea cu acest dicţionar de vectori de cod generalizat produce valoarea $f(x|o_k)$ pentru fiecare vector de cod o_k.

Deoarece în cazul modelelor semicontinue M este de obicei o valoare mare, ecuaţia 2.78 se poate simplifica utilizând cele mai reprezentative L valori pentru $f(x|o_k)$ pentru fiecare x fără a afecta performanţa modelului.

$$b_j(x) = \sum_{k \in \eta(x)} f(x|o_k) b_j(k) \tag{2.79}$$

Deoarece $\eta(x)$ este mai mic decât M, ecuaţia 2.79 reduce în mod semnificativ cantitatea de calcule.

2.5.5 Detalii de implementare

În secţiunile precedente am studiat teoria generală a modelelor Markov ascunse şi câteva variante ale modelului. În această secţiune vom studia detalii practice legate de implementarea modelelor, cum ar fi scalarea, secvenţe de antrenare multiple şi estimarea parametrilor iniţiali. Pentru o parte dintre aceste detalii putem da soluţii analitice exacte, pentru restul doar soluţii empirice bazate pe experienţă.

2.5.5.1 Scalarea

Pentru a înţelege necesitatea scalării în procesul de reestimare a parametrilor modelului considerăm definiţia variabilei $\alpha_t(i)$ dată de ecuaţia 2.47. Se poate vedea că, $\alpha_t(i)$ este o sumă cu termenii având forma

$$\left(\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(o_s) \right)$$

cu $q_t = i$ şi b fiind o probabilitate discretă definită prin ecuaţia 2.38. Deoarece elementele matricilor A şi B sunt probabilităţi cu valori subunitare (de obicei foarte mici), se poate observa că, dacă t creşte, termenii $\alpha_t(i)$ tind către 0. Pentru valori mari ale parametrului t (peste 100) valorile $\alpha_t(i)$ depăşesc şi precizia dublă cu care lucrează calculatoarele. Astfel singura modalitate pentru efectuarea calculelor este încorporarea unei scalări la fiecare reestimare a parametrilor.

Procedura de scalare cea mai simplă înmulţeşte valorile $\alpha_t(i)$ cu un coeficient, care este independent de i pentru a păstra valorile scalate $\alpha_t(i)$ în domeniul de precizie al calculatorului pentru fiecare t, $1 \leq t \leq T$. O scalare similară se va utiliza şi în calculul coeficienţilor $\beta_t(i)$.

Pentru a înţelege mai bine procedura de scalare, considerăm formula de reestimare a coeficienţilor de tranziţie a_{ij}.

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \tag{2.80}$$

Notăm cu $\hat{\alpha}_t(i)$ valoarea scalată a lui $\alpha_t(i)$, iar cu $\widehat{\hat{\alpha}}_i(t)$ varianta locală a lui $\alpha_i(t)$, înainte de scalare.

Pentru $t = 1$ calculăm $\alpha_1(i)$ conform ecuaţiei 2.48 şi iniţializăm $\widehat{\hat{\alpha}}_1(i) = \alpha_1(i)$ cu $c_1 = \frac{1}{\sum_{i=1}^{N} \alpha_1(i)}$ şi $\hat{\alpha}_1(i) = c_1 \alpha_1(i)$. Pentru fiecare t, $2 \leq t \leq T$, calculăm $\widehat{\hat{\alpha}}_1(i)$ conform formulei 2.49 şi obţinem

$$\widehat{\hat{\alpha}}_t(i) = \sum_{j=1}^{N} \hat{\alpha}_{t-1}(j) a_{ji} b_i(o_t). \tag{2.81}$$

Calculăm coeficientul de scalare

$$c_t = \frac{1}{\sum_{i=1}^{N} \widehat{\hat{\alpha}}_t(i)} \tag{2.82}$$

având

$$\hat{\alpha}_t(i) = c_t \widehat{\hat{\alpha}}_t(i) \tag{2.83}$$

Din ecuaţiile 2.81-2.83 obţinem

$$\hat{\alpha}_t(i) = \frac{\sum_{j=1}^{N} \hat{\alpha}_{t-1}(j) a_{ji} b_i(o_t)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \hat{\alpha}_{t-1}(j) a_{ji} b_i(o_t)} \tag{2.84}$$

Prin inducţie putem scrie $\hat{\alpha}_{t-1}(j)$ ca

$$\hat{\alpha}_{t-1}(j) = \left(\prod_{\tau=1}^{t-1} c_\tau\right) \alpha_{t-1}(j) \tag{2.85}$$

Din care rezultă că $\hat{\alpha}_t(i)$ poate fi scrisă ca

$$\hat{\alpha}_t(i) = \frac{\sum_{j=1}^{N} \alpha_{t-1}(j) \left(\prod_{r=1}^{t-1} c_\tau\right) a_{ji} b_i(o_t)}{\sum_{i=1}^{N} \sum_{i=1}^{N} \alpha_{t-1}(j) \left(\prod_{r=1}^{t-1} c_r\right) a_{ji} b_i(o_t)} = \frac{\alpha_t(i)}{\sum_{i=1}^{N} \alpha_t(i)} \tag{2.86}$$

Similar calculăm şi $\beta_t(i)$ utilizând aceiaşi coeficienţi de scalare ca şi pentru şirul α. Astfel obţinem

$$\hat{\beta}_t(i) = c_t \beta_t(i). \tag{2.87}$$

Ecuaţia 2.80 devine

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} \hat{\alpha}_t(i) a_{ij} b_j(o_{t+1}) \hat{\beta}_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \hat{\alpha}_t(i) a_{ij} b_j(o_{t+1}) \hat{\beta}_{t+1}(j)} \tag{2.88}$$

dar fiecare $\hat{\alpha}_t(i)$ poate fi scris ca

$$\hat{\alpha}_t(i) = \left[\prod_{s=1}^{t} c_s\right] \alpha_t(i) = C_t \alpha_t(i) \tag{2.89}$$

şi fiecare $\hat{\beta}_{t+1}(j)$ poate fi scris ca

$$\hat{\beta}_{t+1}(j) = \left[\prod_{s=t+1}^{T} c_s\right] \beta_{t+1}(j) = D_{t+1} \beta_{t+1}(j). \tag{2.90}$$

Astfel ecuaţia 2.80 devine

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1} C_t \alpha_t(i) a_{ij} b_j(o_{t+1}) D_{t+1} \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} C_t \alpha_t(i) a_{ij} b_j(o_{t+1}) D_{t+1} \beta_{t+1}(j)} \tag{2.91}$$

În final termenul $C_t D_{t+1}$ poate fi scris în forma

$$C_t D_{t+1} = \prod_{s=1}^{t} c_s \prod_{s=t+1}^{T} c_s = \prod_{s=1}^{T} c_s = C_T \tag{2.92}$$

care este independent de t.

Este clar că, procedura de reestimare aplicată pentru a_{ij} se poate utiliza atât

pentru coeficienţii B, cât şi pentru π. Singura schimbare datorată scalării se produce în procedura de calcul al probabilităţii $P(O|\lambda)$. Nu putem însuma termenii $\hat{\alpha}_T(i)$, deoarece acestea sunt deja scalate, dar putem utiliza următoarea proprietate

$$\prod_{t=1}^{T} c_t \sum_{i=1}^{N} \alpha_T(i) = C_T \sum_{i=1}^{N} \alpha_T(i) = 1 \tag{2.93}$$

Astfel obţinem

$$\prod_{t=1}^{T} c_t P(O|\lambda) = 1 \tag{2.94}$$

sau

$$\log[P(O|\lambda] = \sum_{t=1}^{T} \log c_t. \tag{2.95}$$

Astfel se poate calcula logaritmul lui P, care oricum ar fi mult prea mic ca să-l putem repreprezenta exact într-un calculator.

Menţionăm că, în cazul utilizării algoritmului Viterbi pentru determinarea secvenţei de stări cu probabilitate maximă (ML), nu este necesară scalare, dacă utilizăm varianta cu logaritmi a algoritmului.

2.5.5.2 Secvenţe de observaţii multiple

În secţiunea 2.5.3 am discutat despre modelul Left-Right unde stările pot fi vizitate într-o ordine crescătoare. Acest model impune o serie de constrângeri pentru elementele matricii de tranziţie A şi pentru vectorul probabilităţilor iniţiale π. Marele neajuns al acestor modele este că o singură secvenţă de observaţii nu ajunge pentru antrenare. Astfel pentru a putea utiliza aceste modele la estimări reale, trebuie utilizate mai multe secvenţe de observaţii la antrenare.

Modificarea procesului de reestimare este simplă şi se face în modul următor. Notăm mulţimea secvenţelor de observaţii

$$O = [O^{(1)}, O^{(2)}, \ldots, O^{(k)}] \tag{2.96}$$

unde $O^{(k)} = (o_1^{(k)}, o_2^{(k)}, \ldots, o_{T_k}^{(k)})$ este secvenţa de observaţii cu numărul de ordine k. Presupunem că fiecare secvenţă de observaţii este independentă de restul secvenţelor de observaţii şi obiectivul nostru este ajustarea parametrilor modelului λ astfel, încât să maximizăm $P(O|\lambda)$.

$$P(O|\lambda) = \prod_{k=1}^{K} P(O^{(k)}|\lambda) = \prod_{k=1}^{K} P_k.$$

Deoarece formulele de reestimare se bazează pe frecvenţele producerii anumitor evenimente, formulele de reestimare pentru secvenţe de observaţii multiple se modifică prin adăugarea acestor frecvenţe. Astfel formulele modificate pentru $\overline{a}_{ij}$ şi $\overline{b_j}(l)$ vor fi

$$\overline{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) a_{ij} b_j(o_{t+1}^{(k)}) \beta_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{2.97}$$

şi

$$\overline{b}_j(l) = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i) \beta_t^k(i)} \tag{2.98}$$

iar π nu necesită reestimare deoarece $\pi_1 = 1$ şi $\pi_i = 0,\ i \neq 1$.

Scalarea corespunzătoare ecuaţiilor 2.97 şi 2.98 este evidentă, deoarece orice secvenţă de observaţii are factorul de scalare propriu. Ideea cheie este eliminarea acestui factor din fiecare termen înainte de însumare. Aceasta se poate realiza prin scrierea ecuaţiei de reestimare cu termenii variabilelor scalate,

$$\overline{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) a_{ij} b_j(o_{t+1}^k) \hat{\beta}_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)} \tag{2.99}$$

Astfel, pentru fiecare secvenţă $O^{(k)}$ aceiaşi factori de scalare vor apare în fiecare termen al sumei pentru fiecare t, care apar şi în termenul P_k şi astfel se simplifică exact. Astfel, utilizând valorile scalate atât pentru termenii α, cât şi pentru termenii β, ne conduc la o valoare nescalată $\overline{a}_{ij}$. Un rezultat similar se obţine şi pentru termenul $\overline{b}_j(l)$.

2.5.5.3 Estimarea parametrilor iniţiali

În teorie ecuaţiile de reestimare ale parametrilor unui MMA trebuie să ne conducă la nişte valori ai parametrilor, care corespund maximului local al funcţiei de probabilitate. O întrebare cheie este, cum să alegem valorile iniţiale ale parametrilor, încât maximul local să fie egal sau cât mai aproape de maximul global al funcţiei de probabilitate.

Pentru această întrebare nu există un răspuns evident şi simplu. Experienţa însă a demonstrat că valori iniţiale aleatoare şi uniform distribuite ale parametrilor π şi A sunt adecvate pentru cele mai multe cazuri. Cu toate acestea, tot experienţa ne-a demonstrat că, pentru parametrul B sunt necesare estimări iniţiale adecvate. Există mai multe căi pentru obţinerea acestor estimări iniţiale şi anume:

- segmentarea manuală a secvenţei de observaţii în stări, şi medierea obser-

vaţiilor pentru firecare stare în parte

- segmentarea Maximum Likelihood şi mediere
- segmentare K-means segmentală cu grupare

2.6 Recunoaşterea vorbirii

2.6.1 Introducere

Fie $X = \{x_1,\ x_2, \ldots, x_t\}$ o secvenţă de vectori acustici corespunzătoare unei rostiri. Problema recunoaşterii înseamnă determinarea acelei secvenţe de cuvinte $\hat{W}$ care satisface

$$\hat{W} = \max_W P(W|X) = \max_W P(X|W) \cdot P(W) \tag{2.100}$$

unde $W = w_1, w_2, \ldots, w_n, \quad w_i \in V$ este o secvenţă de n cuvinte aparţinând unui vocabular V fixat. În formula 2.100 $P(W)$ reprezintă probabilitatea secvenţei de cuvinte W şi este furnizată de componenta de model de limbaj al unui sistem pentru recunoaşterea vorbirii.

Modelarea limbajului este o problemă care apare nu doar în domeniul recunoaşterii vorbirii ci şi în domeniul prelucrării limbajului natural. Există o serie de cărţi, care tratează această problemă foarte detaliat. O parte din cărţi sunt din domeniul recunoaşterii vorbirii [14, 40, 43, 79], altele din domeniul procesării limbajului natural (Natural Language Processing-NLP) [58], iar altele pot fi privite ca şi cărţi interdisciplinare, deoarece tratează ambele domenii [44].

În cazul recunoaşterii vorbirii, cuvintele vor însemna forma cuvintelor pronunţate. Dacă acelaşi cuvânt are două pronunţii diferite, atunci se consideră două cuvinte. Utilizând regula Bayes $P(W)$ se poate descompune în modul următor

$$P(W) = \prod_{i=1}^{n} P(w_i|w_1 w_2 \ldots w_{i-1}) \tag{2.101}$$

unde $P(w_i|w_1 w_2 \ldots w_{i-1})$ este probabilitatea ca w_i să fie pronunţat după ce $w_1,\ w_2, \ldots, w_{i-1}$ au fost pronunţate. Secvenţa $w_1,\ w_2, \ldots, w_{i-1}$ se numeşte istoric şi se notează cu h_i. Rolul modelului de limbaj este să furnizeze sistemului de recunoaştere estimări corespunzătoare ale probabilităţilor $P(w_i|w_1 w_2 \ldots w_{i-1})$, ceea ce înseamnă practic predicţia cuvântului ce urmează după secvenţa de cuvinte $w_1 w_2 \ldots w_{i-1}$. Schema unui sistem, care utilizează regula lui Bayes pentru decizie este prezentată în figura 2.12.

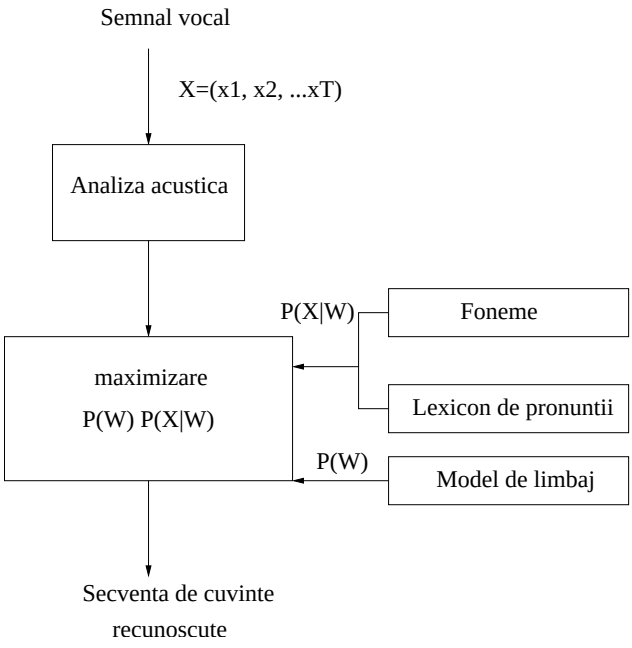


Figura 2.12: Regula lui Bayes pentru un sistem de recunoaşterea vorbirii

Relativ la acest sistem putem nota următoarele observaţii:

- Modelul de limbaj $P(W)$ este independent de observaţiile acustice şi are rolul de a ghida sistemul în concatenarea cuvintelor limbajului pentru a forma propoziţii.

- Modelul acustico-fonetic, $P(X|W)$, este probabilitatea condiţionată de a observa vectorii acustici X, când vorbitorul pronunţă secvenţa de cuvinte W. Atât aceste probabilităţi, cât şi probabilităţile corespunzătoare modelului de limbaj sunt estimate în faza de antrenare a sistemului de recunoaştere. În cazul unui sistem de recunoaştere având un vocabular mare, unitatea de recunoaştere este o parte a cuvântului, de obicei fonemul, silaba sau demisilaba. Modelele cuvintelor se obţin prin concatenarea acestor modele conform transcripţiilor fonetice ale cuvintelor.

- Decizia, care are ca şi rezultat secvenţa de cuvinte pronunţate, trebuie să fie luată cu ajutorul unei proceduri de optimizare, care combină informaţii din diferite surse de cunoaştere: modelul de limbaj, modelele acustico-fonetice ale fonemelor şi lexicul de pronunţii sau dicţionar. Procedura de optimizare este de fapt o căutare în spaţiul stărilor definite de aceste surse de cunoaştere.

2.6.2 Modele de limbaj

Există mai multe metode pentru modelarea limbajului, însă nu toate s-au dovedit viabile. O primă metodă pentru reprezentarea limbajului ar fi cu ajutorul unui automat finit (Finite State Automaton-FSA). Automatele finite acceptă limbajele regulare, care sunt descrise de gramatici regulare [26, 40, 44]. Un prim dezavantaj ar fi că, gramaticile regulare nu sunt suficient de puternice pentru generarea limbajelor naturale. O a doua metodă mai puternică pentru modelarea limbajelor este oferită tot de teoria limbajelor formale prin gramaticile independente de context, care sunt capabile să genereze limbaje mai puternice. În practică se utilizează varianta stohastică a acestor gramatici. Cărţile [40, 44] tratează detaliat această metodă de reprezentare a limbajului. O metodă simplă, dar utilizată în practică este cea de a reprezenta spaţiul limbajului prin perechi de cuvinte corecte. Această metodă se mai numeşte şi model $n - gram$ şi se intergrează cu uşurinţă în căutarea Viterbi. Modelul limbaj se utilizează de către componenta de căutare, care de obicei este o căutare de la stânga-dreapta. Notând cu $h_i = (w_1,\ w_2, \ldots w_{i-1})$ istoricul cuvântului w_i, ecuaţia 2.101 devine

$$P(W) = \prod_{i=1}^{n} P(w_i|h_i). \tag{2.102}$$

De obicei se utilizează clase de echivalenţe pentru h_i. Notând cu $\phi(h_i)$ clasa de echivalenţă, căreia aparţine h_i, formula 2.102 se scrie

$$P(W) = \prod_{i=1}^{n} P(w_i|\phi(h_i)). \tag{2.103}$$

Modelele n-gram utilizează cele $n-1$ cuvinte ca şi istoricul cuvântului w_i, adică $\phi(h_i) = \{w_{i-1},\ w_{i-2}, \ldots, w_{i-(n-1)}\}$. Pentru cazul modelelor trigram în calcule se utilizează formula

$$f(w_3|w_1w_2) = \frac{c(w_1w_2w_3)}{c(w_1w_2)} \tag{2.104}$$

unde $c()$ reprezintă frecvenţa de apariţie a unei anumite secvenţe de cuvinte, iar $f(|)$ reprezintă frecvenţa relativă. Din nefericire formula 2.104 nu este una foarte adecvată, deoarece foarte multe triplete $w_1w_2w_3$ nu apar nici în cazul textelor de antrenare de dimensiuni uriaşe. În anii 1970 cercetătorii de la IBM au făcut un experiment pe un corpus format din descrierea unor inovaţii având un vocabular de 1000 de cuvinte. Acest text a fost împărţit într-un set de antrenare şi într-

unul de testare format din 300000 repectiv 150000 cuvinte. În mod surprinzător 23% din trigramele din setul de testare nu au apărut deloc în setul de antrenare. Utilizarea unui model de limbaj definit cu formula 2.104 ar fi cauzat o eroare cel puţin de 23%.

De aceea este necesară netezirea frecvenţelor relative. Acest lucru se poate realiza simplu prin interpolarea frecvenţelor relative trigram, bigram şi unigram dată de următoarea formulă

$$P(w_3|w_1w_2) = \lambda_3 f(w_3|w_1w_2) + \lambda_2 f(w_3|w_2) + \lambda_1 f(w_3)$$

unde $\lambda_1 + \lambda_2 + \lambda_3 = 1, \quad \lambda_i \geq 0$.

2.6.3 Măsurarea erorii în sisteme de recunoaştere a vorbirii

Evaluarea performanţei unui sistem de recunoaşterea vorbirii este o problemă importantă. Eroarea de recunoaştere a cuvintelor este măsura cea mai folosită. Datele disponibile sunt împărţite în prealabil în date de antrenare şi date de test. Sistemul se antrenează folosind doar datele de antrenare, iar la testare sistemului i se prezintă datele de test, pe baza cărora se calculează eroarea.

Pentru măsurarea erorii de recunoaştere a cuvintelor se consideră trei tipuri de erori:

- *substituire (SUBST):* sistemul recunoaşte un cuvânt incorect în locul unuia corect
- *ştergere (DEL):* un cuvânt corect a fost omis din propoziţia recunoscută
- *inserare (INS):* un cuvânt care nu există în semnalul acustic a fost adăugat în propoziţia recunoscută

Rata de eroare pe cuvinte se defineşte ca fiind

$$WER = \frac{SUBST + DEL + INS}{N} \tag{2.105}$$

unde N este numărul de cuvinte din propoziţia prezentată sistemului pentru recunoaştere. Această problemă de aliniere este una clasică şi este cunoscută sub numele de cel mai lung subşir comun [23]. Problema se rezolvă uşor cu metoda programării dinamice, algoritmul se bazează pe completarea unui tabel folosind următoarea recurenţă:

Algoritm 10 Algoritmul WER

Pas 1. Iniţializare $d[0,0] = 0 \quad b[0,0] = 0$
Pas 2. Recurenţă
  Pentru fiecare $i = 1 \ldots n$
    Pentru fiecare $j = 1 \ldots m$
      Se calculează $d[i,j]$ şi $b[i,j]$ conform ecuaţiilor (2.106) şi (2.107)
Pas 3. $WER = \frac{d[n,m]}{n} \cdot 100\%$, Reconstituire drum optim conform cu 2.108

$$d[i,j] = \min \begin{cases} d[i-1,j]+1 & (stergere) \\ d[i-1,j-1] & (potrivire) \\ d[i-1,j-1]+1 & (substituire) \\ d[i,j-1]+1 & (inserare) \end{cases} \tag{2.106}$$

unde $d[i,j]$ reprezintă eroarea minimă în alinierea subşirurilor $A = (a_1, a_2, \ldots, a_i)$ şi $B = (b_1, b_2, \ldots b_j)$. Eroarea totală acumulată pentru şirurile $A = (a_1, a_2, \ldots, a_n)$ şi $B = (b_1, b_2, \ldots, b_m)$ va fi $d[n,m]$. Dacă dorim să calculăm separat numărul de inserări, ştergeri şi substituiri, va trebui să utilizăm încă un tabel ajutător pentru a reţine tipul erorii la fiecare $d[i,j]$. Astfel, dacă notăm

$$b[i,j] = \begin{cases} 1 & stergere \\ 2 & inserare \\ 3 & potrivire \\ 4 & substituire \end{cases} \tag{2.107}$$

vom putea determina exact frecvenţele celor trei tipuri de erori. Pentru reconstruirea drumului optim format din stările $(s_1, s_2, \ldots, 0)$ se utilizează formulele

$$s_1 = B[n,m]$$

$$s_t = \begin{cases} B[i-1,\ j] & daca \quad s_{t-1} = 1 \\ B[i,\ j-1] & daca \quad s_{t-1} = 2 \\ B[i-1,j-1] & daca \quad s_{t-1} = 3\ sau\ 4 \end{cases} \quad pentru\ t = 2, \ldots s_t = 0 \tag{2.108}$$

Algoritmul este definit în (alg. 10).

2.6.4 Tehnici de căutare

Există două strategii de bază pentru căutarea propoziţiei celei mai probabile care satisface atât constrângerile acustice cât şi cele lingvistice: strategia modulară şi

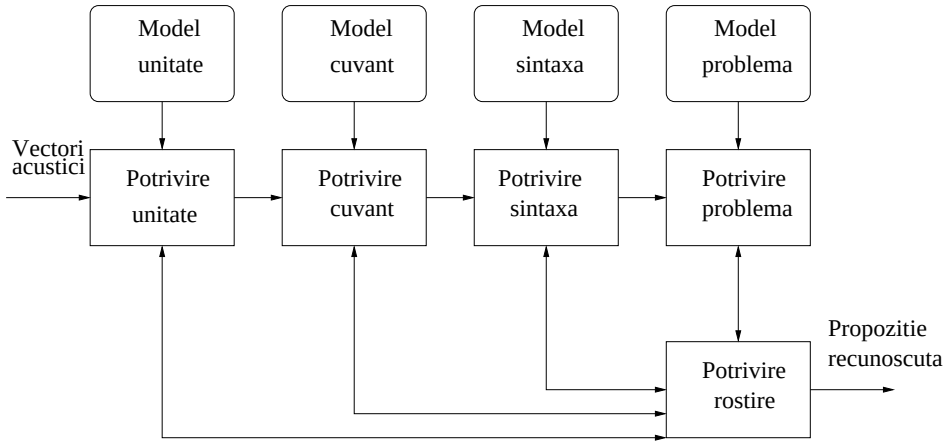


Figura 2.13: Un sistem ASR cu structură integrată

strategia integrată. În varianta integrată decizia asupra recunoaşterii se ia luând în considerare toate sursele de cunoaştere (vezi figura 2.13). Utilizând toate sursele acustice, lexicale, sintactice şi semantice se poate construi un automat finit, care va fi practic maşina de recunoaştere. Această strategie este extrem de des folosită în sistemele de recunoaştere a vorbirii.

Pe de altă parte, în abordarea secvenţială, prezentată în figura 2.14, recunoaşterea se obţine prin potrivire acustică, potrivire lexicală, potrivire sintactică şi potrivire semantică într-un mod secvenţial. Avantajul acestei abordări este în primul rând simplitatea şi posibilitatea de a implementa şi testa separat modulele componente. Din punct de vedere computaţional acest tip de sistem este mai abordabil. Există şi o serie de neajunsuri la această abordare, dintre care cel mai grav ar fi faptul că, în fiecare modul în parte se iau o serie de decizii fără a lua în considerare restul modulelor, dar care influenţează acestea, modulele fiind conectate în mod secvenţial. Un prim exemplu pentru o asemenea sursă de eroare ar fi ca, în primul modul, care realizează potrivirea unităţilor (de exemplu foneme), dacă se determină secvenţa cea mai probabilă de foneme corespunzătoare secvenţei acustice, fără să se ia în considerare lexicul limbajului, am putea obţine secvenţe de foneme inexistente în limbajul în cauză.

Căutarea poate fi făcută utilizând o singură trecere (single pass) în spaţiul acustic sau utilizând mai multe treceri (multi pass). Spaţiul de căutare poate fi expandat în mod static sau în mod dinamic. Expandarea statică a spaţiului de căutare a devenit posibilă doar în ultimii ani, primii care au realizat un astfel de sistem au fost cei de la AT&T [62].

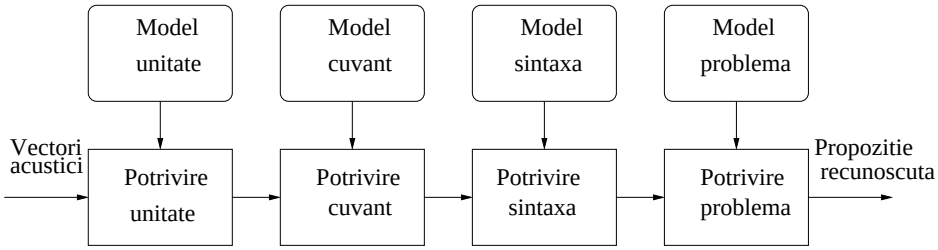


Figura 2.14: Un sistem ASR cu structură modulară

Tehnicile de căutare cele mai cunoscute sunt: căutare cadru-sincron cu o singură trecere, decodarea cu stivă.

2.6.4.1 Căutare cadru-sincron cu o singură trecere

Acest tip de căutare îşi propune găsirea celui mai probabil drum într-o reţea finită. O căutare totală în lăţime este de obicei impracticabilă datorită timpului uriaş de procesare. Pentru a reduce timpul de căutare se utilizează strategii de căutare suboptimale. O strategie este utilizarea aproximaţiei Viterbi, iar o altă strategie este căutarea cu fascicul. În căutarea cu fascicul cadru-sincron doar o parte din ipotezele aparţinând unui fascicul sunt expandate pentru fiecare moment de timp. Algoritmul astfel obţinut este unul care aproximează o căutare în lăţime necesitând întregul spaţiu de căutare pe tot parcursul căutării. Integrarea surselor de cunoştinţe lexicale pot reduce semnificativ numărul ipotezelor viabile. Lăţimea fasciculului se determină experimental şi este dependentă atât de model cât şi de problemă.

Definirea problemei-Aproximaţia Viterbi

La început algoritmul cu o singură trecere timp-sincron a fost proiectat pentru sisteme de recunoaştere a vorbirii cu un vocabular limitat, ca recunoaşterea secvenţelor de cifre. Acest algoritm însă s-a dovedit eficient şi în sisteme având vocabulare cu peste 20.000 de cuvinte [64]. Spaţiul de căutare poate fi considerat ca şi o maşină finită de stări, definită în termenii modelelor acustici şi ai modelului limbajului. Pentru o secvenţă de vectori acustici $X = \{x_1 x_2 \ldots x_T\}$ definim

secvenţa optimă de cuvinte prin

$$\left[w_1^N\right]_{opt} = \arg\max_{\left[w_1^N\right]} \left\{ P\left(w_1^N\right) \cdot \sum_{\left[s_1^T\right]} P\left(x_1^T; s_1^T | w_1^N\right) \right\} \tag{2.109}$$

unde suma se referă la toate secvenţele de stări posibile $\left[s_1^T\right]$ consistente cu secvenţa de cuvinte $\left[w_1^N\right]$. Aproximând suma cu maximul, obţinem aproximarea Viterbi

$$\left[w_1^N\right]_{opt} = \arg\max_{\left[w_1^N\right]} \left\{ P\left(w_1^N\right) \cdot \max_{\left[s_1^T\right]} P\left(x_1^T; s_1^T | w_1^N\right) \right\}. \tag{2.110}$$

Altfel spus, secvenţa de cuvinte cea mai probabilă se aproximează prin secvenţa de stări cea mai probabilă. În aproximarea Viterbi, maximul nu va fi o estimare a sumei, doar cele două expresii conduc la aproximativ aceeaşi secvenţă de cuvinte.

În cazul aproximării Viterbi problema căutării se defineşte în modul următor:

Dorim să asociem cu fiecare vector acustic corespunzător momentului de timp t o pereche (stare, cuvânt). Această asociere poate fi privită ca o aliniere a timpului la secvenţa de perechi (stare, cuvânt):

$$(s_1, w_1), \ldots, (s_t, w_t), \ldots, (s_T, w_T) \tag{2.111}$$

Un exemplu de aliniere pentru problema recunoaşterii unei secvenţe de cuvinte este arătat în figura 2.15. La capetele cuvintelor avem nevoie de condiţii de continuitate: să permitem tranziţii între stările $S(v)$, ultima stare a cuvântului v şi prima stare a oricărui cuvânt w. Strategia programării dinamice ne va permite să evaluăm probabilităţile următoare

$$P(w_1 \ldots w_N) \cdot P(x_1 x_2 \ldots x_T; s_1 s_2 \ldots s_T | w_1 w_2 \ldots w_N) \tag{2.112}$$

într-o manieră stânga-dreapta şi să determinăm secvenţa optimă de stări. Remarcăm faptul că, în această formulare a problemei secvenţa de stări şi secvenţa de cuvinte sunt estimate simultan.

Căutare cu fascicul cadru-sincron

Utilizarea metodei programării dinamice presupune că, soluţia optimă a problemei se poate construi din soluţii optime ale subproblemelor [23]. De aceea se

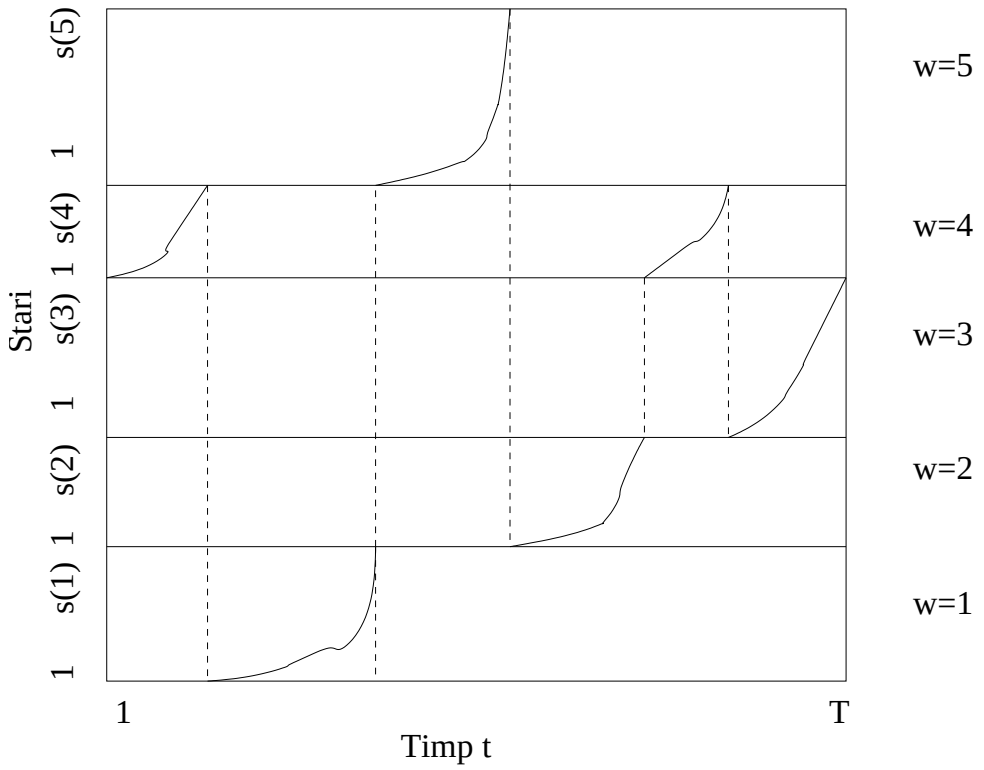


Figura 2.15: Aliniere timp-perechi stări cuvinte

utilizează un tablou pentru stocarea soluţiilor optimale ale subproblemelor. Pentru reconstituirea drumului se utilizează un tablou suplimentar. Notăm aceste tablouri în modul următor:

$Q(t, s; w)$: scorul maxim obţinut de drumul care se termină în starea s al cuvântului w la momentul de timp t.

$B(t, s; w)$: timpul de început al drumului optim care se termină în starea s al cuvântului w la momentul de timp t.

Figura 2.16 ne arată că avem două tipuri de tranziţii: între stările aceluiaşi cuvânt şi între stări ale diferitelor cuvinte. Din această cauză vom defini două seturi de reguli pentru completarea tabelelor. Pentru stările aceluiaşi cuvânt definim următoarele relaţii recurente:

$$Q(t, s; w) = \max_{\sigma} \{q(x_t, s|\sigma; w) \cdot Q(t-1, \sigma; w)\} \tag{2.113}$$

$$B(t, s; w) = B(t-1, \sigma_{max}(t, s; w); w) \tag{2.114}$$

unde

$$q(x_t, s|\sigma; w) = a(s|\sigma; w) \cdot b(x_t|\sigma; w) \tag{2.115}$$

este probabilitatea condiţională ca în starea σ a cuvântului w să fie observat vectorul acustic x_t şi să se treacă în starea s. $a(s|\sigma; w)$ este probabilitatea de tranziţie din starea s în starea σ, iar $b(x_t|\sigma; w)$ probabilitatea de emitere a vectorului acustic x_t în starea σ al cuvântului w, iar $\sigma_{max}(t, s; w)$ este starea optimă precedentă a ipotezei $(t, s; w)$. Pentru a include capetele cuvintelor, introducem o stare specială $s = 0$ utilizată pentru a începe un cuvânt nou. În momentul atingerii unui capăt de cuvânt, vom combina acest cuvânt cu cuvintele precedente. Pentru acest scop definim

$$H(w; t) := \max_{v} \{p(w|v) \cdot Q(t, S(v); v)\} \tag{2.116}$$

unde $p(w|v)$ este probabilitatea condiţionată al modelului limbaj de tip bigram, adică probabilitatea ca cuvântul v să fie urmat de cuvântul w. Cu simbolul $S(v)$ se notează starea terminală a cuvântului v. În acest scop trebuie să transferăm atât scorul, cât şi indicele de timp:

$$Q(t-1, 0; w) = H(w; t-1) \tag{2.117}$$

$$B(t-1, 0; w) = t-1. \tag{2.118}$$

Ecuaţiile 2.117 şi 2.118 presupun că prima dată sunt evaluate stările normale

Algoritm 11 Algoritmul cu o singură trecere Single-best

Pentru $t = 1 \ldots T$ execută
  Procesare la nivelul stărilor acustice
    P1. Iniţializare cu ecuaţiile 2.117 şi 2.118
    P2. Aliniere, completarea tabelelor conform recurenţelor 2.113 şi 2.114
    P3. Eliminarea ipotezelor necorespunzătoare
  Procesrea la nivelul cuvintelor (stare specială)
    Pentru fiecare pereche (w, t) execută
      P1. $H(w; t) = max_v \left[p(w|v)Q_v(t, S(w); w)\right]$
      P2. $v_0(w; t) = \arg\max_v \left[p(w|v)Q_v(t, S(w); w)\right]$
        se stochează predecesorul cel mai bun $v_0 = v_0(w, t)$
        se stochează marginea cea mai bună $\tau_0 = B(t, S(v_0); v_0)$

$s = 1, \ldots, S(w)$ pentru fiecare cuvânt w şi după aceea se va evalua starea $s = 0$. Astfel $Q(t, s; w)$ va conţine atât probabilităţile furnizate de către modelele acustice cât şi probabilităţile modelului de limbaj.

Algoritmul este dat în continuare (alg. 11). Secvenţa de vectori acustici este procesată de la stânga la dreapta. Algoritmul de căutare este o căutare în lăţime, în care toate ipotezele sunt extinse în mod paralel pentru fiecare vector acustic.

Deoarece toate ipotezele se referă la aceeaşi secvenţă de intrare, scorurile acestora sunt comparabile. De aceea putem evita căutarea exhaustivă, eliminând ipotezele nepromiţătoare. Pentru fiecare vector acustic se determină ipoteza cea mai bună, iar toate acele ipoteze care au scoruri mai mici cu un factor fixat, decât aceasta, sunt eliminate. Experimentele arată că, pentru acest tip de căutare cu fascicul, care utilizează la fiecare moment vectorul acustic şi constrângerile impuse de modelul limbaj, doar un mic procent de ipoteze rămân viabile, toate celelalte fiind eliminate.

În algoritmul 11. lexiconul a fost organizat într-un mod liniar. Fiecare cuvânt a fost reprezentat separat ca o secvenţă liniară de foneme, într-un mod independent de celelalte foneme. Într-un vocabular mare (cu mai multe mii de cuvinte), există foarte multe cuvinte cu acelaşi prefix, ceea ce sugerează organizarea lexiconului sub formă de arbore.

Experimentele arată că, majoritatea ipotezelor sunt concentrate pe stări aparţinând fonemelor de la începutul cuvintelor. De aceea este ideală organizarea vocabularului de pronunţii sub formă de arbore.

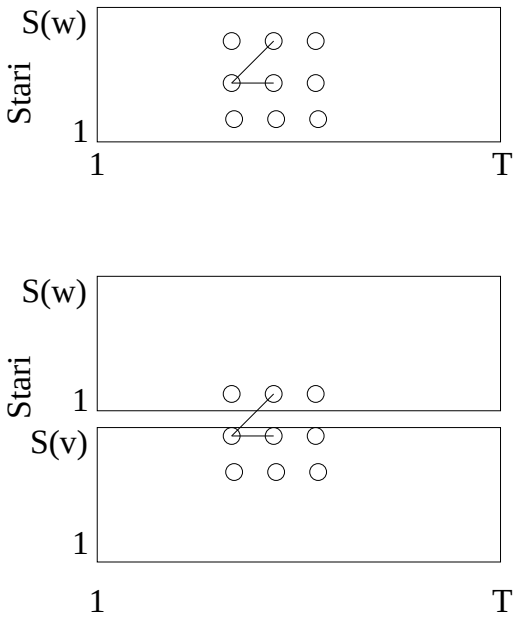


Figura 2.16: Tipuri de tranziţii

2.6.4.2 Decodarea cu stivă

Această tehnică este bazată pe strategia „primul cel mai bun" (Best-First). De obicei această strategie este implementată cu ajutorul unei stive pentru fiecare moment de timp, care menţine o listă ordonată a ipotezelor. În practică s-a dovedit că, găsirea unei funcţii euristice care să permită selectarea drumului optim este aproape imposibilă. Din această cauză căutarea cu stivă este în general mai puţin optimală decât căutarea cadru-sincron cu fascicul.

Decodarea cu stivă este o variantă a căutării A^* bazată pe algoritmul forward, unde funcţia de evaluare este bazată pe probabilitatea forward. Această căutare este de fapt o căutare într-un arbore. Căutarea cu stivă, fiind un algoritm de căutare în arbore, caută acel drum în arbore, ale cărui ramuri corespund cuvintelor vocabularului V, nodurile neterminale corespund propoziţiilor incomplete, iar cele terminale propoziţiilor complete. Figura 2.17 arată un arbore de căutare pentru un vocabular format din 3 cuvinte $V = \{w_1,\ w_2,\ w_3\}$.

Un avantaj al decodorului cu stivă ar fi potrivirea cu algoritmul de antrenare forward-backward. Deoarece căutarea Viterbi este de fapt o căutare într-un graf, probabilităţile ataşate drumurilor nu pot fi însumate, deoarece aceste drumuri pot corespunde unor secvenţe diferite de cuvinte. În general căutarea Viterbi returnează secvenţa de stări optime în loc de secvenţa optimă de cuvinte. Pe

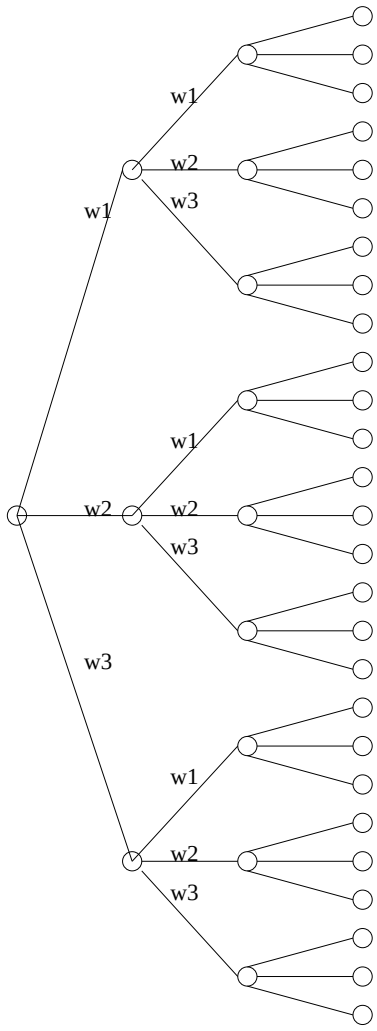


Figura 2.17: Arbore de căutare pentru decodorul cu stivă

de altă parte, decodorul cu stivă, fiind o căutare într-un arbore, pentru fiecare nod există un singur drum de la rădăcină la nodul respectiv, permiţând utilizarea algoritmului forward pentru evaluarea modelelor de cuvinte.

Comparând cu algoritmul cadru-sincron Viterbi putem afirma despre acest algoritm următoarele:

- nu este cadru-sincron
- lucrează cu drumuri de lungimi diferite

Căutarea începe prin adăugarea tuturor ipotezelor de un singur cuvânt la o listă OPEN. Se alege ipoteza cea mai bună, se scoate din lista OPEN şi se adaugă

toate ipotezele de drumuri la care care se poate extinde ipoteza cea mai bună. Se continuă căutarea până când se găseşte drumul cel mai bun din toate drumurile din lista OPEN.

Deoarece căutarea cu stivă se realizează în mod asincron, este necesar un mecanism efectiv pentru a determina sfârşitul fonemelor, respectiv al cuvintelor. Astfel pentru a putea utiliza căutarea cu stivă sunt necesare următoarele:

- o funcţie euristică pentru a estima secvenţa de vectori acustici încă neprelucraţi
- o metodă pentru a determina sfârşitul fonemelor/cuvintelor

2.7 Baze de date

Pentru experimentele din această lucrare am folosit două baze de date de sunet. Baza de date TIMIT este o bază de date cu vorbitori în engleză, înregistrată în condiţii ideale, de birou. Conţine material vocal de la 630 de vorbitori, fiecare rostind 10 propoziţii. Rata de eşantionare este 16kHz cu eşantioane de 16 biţi. Întregul material fonetic a fost segmentat manual şi împărţit în parte de antrenare şi parte de testare. Partea de antrenare conţine întregul material de la 462 vorbitori iar partea de testare întregul material audio de la 168 vorbitori. Fiecărei înregistrări audio îi este ataşat un fişier de date, care conţine segmentarea în foneme indicând capetele în milisecunde ale acestora cât şi etichetele fonemelor. În total sunt 61 de simboluri fonetice.

Cea de-a doua bază de date utilizată a fost OASIS Numbers, proiectată pentru recunoaşterea numerelor din limba maghiară. Detalii despre această bază de date găsim în [47]. Frecvenţa de eşantionare este 22.05kHz cu eşantioane de 16 biţi, condiţii de birou. Partea segmentată a bazei de date este formată dintr-un material audio rostit de către 26 de vorbitori, dintre care un copil, 9 femei şi 16 bărbaţi. Fiecare vorbitor citeşte de două ori cifrele $0-9$, numerele $10-100$, 1000, 1000000 şi patru variante care se utilizează în formarea unor numere compuse cum ar fi 28. Orice număr din limba maghiară se poate forma din concatenarea acestor 26 de cuvinte. Segmentarea şi anotarea bazei de date a fost făcută manual şi s-au utilizat 31 de simboluri fonetice, ceea ce reprezintă doar o parte din fonemele limbii maghiare.

Capitolul 3

Recunoaşterea fonetică a vorbitorului

3.1 Introducere

Recunoaşterea vorbitorului reprezintă procesul automat al recunoaşterii sau al verificării aceluia care vorbeşte, pe baza caracteristicilor individuale ce derivă din semnalul vocal al vorbitorului. Multe din aceste caracteristici sunt independente de conţinutul lingvistic şi sunt considerate în general surse de degradare în recunoaşterea vorbirii.

Recunoaşterea vorbitorului în general prezintă două probleme diferite: identificarea vorbitorului şi verificarea vorbitorului. Identificarea vorbitorului este procesul în care se determină care din vorbitorii înregistraţi a pronunţat fraza în cauză. Cu alte cuvinte, scopul este de a determina care din vocile cunoscute se apropie cel mai mult de vocea înregistrată. Verificarea vorbitorului este în esenţă o problemă de verificare de ipoteze, care va trebui să valideze sau să elimine o identitate presupusă. Scopul este apoi de a determina dintr-o voce înregistrată, dacă persoana este într-adevăr cea care se susţine a fi. În identificarea vorbitorului numărul alternativelor este egal cu mărimea populaţiei (posibil plus unu în cazul problemei deschise, când vorbitorul de test nu face parte din vorbitorii înregistaţi). În cazul verificării vorbitorului există două cazuri: acceptare şi respingere, indiferent de mărimea populaţiei. Structura simplificată a sistemelor de recunoaştere a vorbitorului este redată în figura 3.1. În blocul Extragerea parametrilor se extrag parametrii acustici (care în mod ideal vor fi specifici vorbitorului) din semnalul vocal. Cu comutatorul în poziţia 1 ne aflăm în modul de antrenare, adică crearea modelelor pentru fiecare client potenţial. Modelele statistice cele mai populare

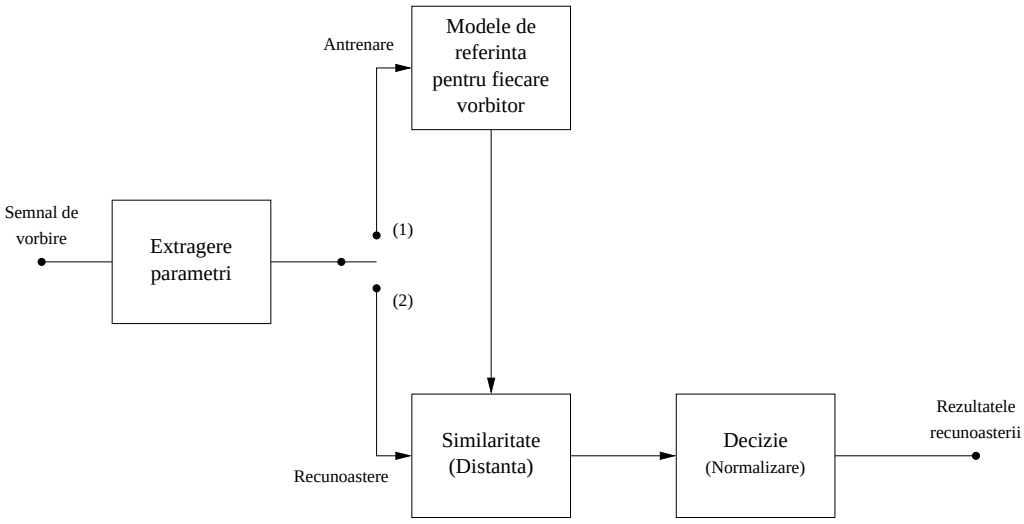


Figura 3.1: Structura unui sistem de recunoaştere a vorbitorului

sunt GMM [16, 71, 70], HMM [36], VQ [21, 45, 46] sau reţele neuronale ANN [15]. Cu comutatorul în poziţia 2 vom recunoaşte secvenţele de test prin compararea parametrilor extraşi din semnalul vocal cu modelele de referinţă ale vorbitorilor înregistraţi. În cazul identificării vorbitorilor vom selecta pur şi simplu vorbitorul cu gradul maxim de similaritate. În cazul verificării vorbitorului decizia este luată după calculul similarităţii si compararea acesteia cu o valoare de prag.

Tehnologia de verificare a vorbitorului are multe aplicaţii: face posibilă verificarea identităţii persoanelor prin control acces vocal, fiind considerată mult mai convenabilă decât folosirea codurilor PIN, cartelelor, cheilor sau altor metode artificiale. Alte aplicaţii potenţiale ar fi: tranzacţii bancare prin telefon, acces la baze de date, comerţ electronic prin telefon, control de securitate pentru zone cu informaţii confidenţiale. În sfârşit verificarea şi identificarea vorbitorului pot fi folosite în aplicaţii militare.

Verificarea vorbitorului poate fi bazată pe metode dependente de text sau independente de text. În cazul metodelor dependente de text vorbitorul foloseşte aceeaşi frază sau parolă atât în antrenare, cât şi în recunoaştere, iar în cazul metodelor independente de text nu există această constrângere. Verificarea vorbitorului prin metoda dependentă de text se va folosi de informaţii lexicale în afară de caracteristicile vocale ale vorbitorului, pe când metodele independente de text vor folosi doar acestea din urmă. În general, din cauza variaţiilor acustice-fonetice mari ale înregistrărilor independente de text, este nevoie de un material de antreanre mai vast pentru a caracteriza corect vorbitorul, comparativ cu metodele

dependente de text. Metodele dependente de text se bazează în general pe tehnici de potrivire a formelor, în care axa timpului secvenţei de intrare este aliniată cu fiecare formă de referinţă, iar similaritatea calculată se acumulează de la începutul până la sfârşitul rostirii. Deoarece această metodă foloseşte direct identităţile vocale asociate fiecărei foneme sau silabe, în general atinge rate de recunoaştere mai mari decât sistemele independente de text. Există câteva aplicaţii în care nu pot fi folosite cuvinte predeterminate. De exemplu rostiri ale aceluiaşi cuvânt nu pot fi comparate în investigaţii criminale. Foarte des apare cazul în care vorbitorii nu sunt cooperanţi şi îşi schimbă intenţionat rata şi felul vorbirii. De altfel oamenii pot recunoaşte vorbitorii indiferent de conţinutul rostirii.

Ambele metode de verificare au o problemă serioasă: vocea vorbitorului poate fi înregistrată şi apoi redată. Pentru a preveni această problemă, apare soluţia de verificare a vorbitorului cu text promptat constând într-un text sintetic şi vorbitorul este rugat să repete o secvenţă aleatoare de cuvinte cheie, care sunt luate dintr-un dicţionar restrâns (de exemplu cifre). În acest caz recunoaşterea vorbirii independentă de vorbitor, se efectuează înaintea verificării vorbitorului, pentru a se asigura că vorbitorul într-adevăr a repetat ceea ce i s-a cerut. Astfel rostirile preînregistrate nu vor fi de nici un folos impostorului, deoarece acesta nu va mai putea prevedea secvenţa cerută. Până şi acest sistem poate fi păcălit cu un sistem avansat care reproduce cuvintele cheie în ordinea cerută. Astfel s-a propus o nouă soluţie în care secvenţele aleatoare promptate sunt luate dintr-un vocabular foarte mare.

În servicii reale este de dorit ca un utilizator să-şi poată alege singur parola la care se efectuează verificarea: verificarea vorbitorului prin parolă definită de utilizator. Această metodă permite securitate mărită, deoarece accesul fraudulos necesită cunoaşterea parolei clientului, iar utilizatorul îşi poate schimba parola oricând doreşte.

Cercetările recente în domeniul recunoaşterii vorbitorului încearcă să utilizeze informaţii de nivel înalt, caracteristic vorbitorului, cum ar fi stilul de vorbire, duratele specifice ale fonemelor vorbitorului, informaţii lexicale şi informaţii legate de prozodie. Desigur, aceste informaţii de nivel înalt nu vor înlocui informaţiile acustice, ci le vor extinde. O primă încercare în acest sens a fost făcută de către [28], cu modelarea vorbitorilor prin metode cunoscute din recunoaşterea vorbirii şi anume cu utilizarea n-gramelor. Metoda bazată pe duratele specifice ale diferitelor unităţi lingvistice, specifice vorbitorilor, a fost utilizată cu succes în lucrarea [34]. Modele structurate fonetic au fost utilizate în [33], iar lucrarea [60] prezintă investigaţii în scopul fuzionării a mai multor tipuri de caracteristici

de nivel înalt pentru reprezentarea vorbitorului.

Scopul nostru este de a studia posibilităţile de integrare a conţinutului fonetic în modelul vorbitorului. Pentru acesta vom începe cu studierea conţinutului din punct de vedere fonetic al unui model clasic, format din componente gaussiene. După acesta vom construi modele de vorbitori utilizând doar câte o anumită clasă fonetică, cum ar fi vocalele sau fricativele. Pe baza acestor modele vom putea determina puterea de discriminare a acestor clase fonetice pentru aplicaţii ca recunoaşterea vorbitorilor. În final vom construi modele de vorbitori structurate fonetic şi vom compara performanţa acestor modele cu cele clasice. O publicaţie proprie dedicată studiului conţinutului fonetic ale modelelor GMM de vorbitori este [8], iar publicaţii legate de puterea de discriminare a claselor fonetice sunt [8, 9, 11].

3.2 Modelarea vorbitorului

În această secţiune vom prezenta diferite metode de modelare a vorbitorilor. De fapt, metodele au fost prezentate în capitolul 2, aici vom prezenta doar utilizarea lor pentru rezolvarea acestei probleme specifice, identificarea vorbitorului.

În cazul utilizării metodei VQ prezentată în 2.4.4, fiecare model de vorbitor s-a obţinut separat utilizând partea de antrenare a materialului înregistrat de la fiecare vorbitor. Astfel s-a obţinut câte un dicţionar de coduri, care reprezintă clasele acustice ale vorbitorului. Presupunând că un sistem de recunoaştere are N vorbitori înregistraţi, vom nota modelul pentru vorbitorul i cu

$$\lambda_i = \left(m_1^{(i)},\ m_2^{(i)}, \ldots, m_M^{(i)}\right),$$

unde cu M am notat dimensiunea modelului. Fiecare vorbitor a fost reprezentat cu un model de aceeaşi dimensiune.

În faza de identificare a vorbitorului notăm cu $X = \{x_1,\ x_2, \ldots, x_T\},\ x_i \in R^d$ vectorii acustici extraşi din semnalul vocal al vorbitorului care a cerut identificarea. Identificarea necesită calculul a N distanţe, adică se va calcula distanţa mulţimii X faţă de fiecare model de vorbitor. Pentru modelul λ_i distanţa este definită de către următoarea formulă

$$d(X, \lambda_i) = \frac{1}{T} \sum_{k=1}^{T} \min_{j=1..M} d_E\left(x_k, m_j^{(i)}\right) \tag{3.1}$$

unde d_E este metrica Euclidiană definită în R^d. Vorbitorul se va identifica, ca

fiind acel vorbitor, pentru care distorsiunea definită de 3.1 este minimă, adică

$$Id = \arg \min_{i=1,N} d(X, \lambda_i) \tag{3.2}$$

Cel de-al doilea model matematic utilizat pentru reprezentarea unui vorbitor a fost modelul mixturilor gaussiene prezentată în secţiunea 2.4.6. Şi în acest caz modelele formate din ponderi, valori medii şi matrici de covarianţe sunt notate cu $\{\lambda_1,\ \lambda_2, \ldots, \lambda_N\}$. Obiectivul este de găsi acel vorbitor, pentru care probabilitatea a posteriori pentru secvenţa de vectori acustici $X = \{x_1,\ x_2, \ldots, x_T\}$ este maximă. Prin formulă se exprimă în modul următor

$$Id = \arg \max_{1 \le i \le N} P(\lambda_i|X) = \arg \max_{1 \le i \le N} \frac{p(X|\lambda_i)}{p(X)} P(\lambda_i). \tag{3.3}$$

Considerând că fiecare vorbitor are aceeaşi probabilitate, iar $p(X)$ constant, ecuaţia 3.3 devine

$$Id = \arg \max_{1 \le i \le N} p(X|\lambda_i). \tag{3.4}$$

Pentru a calcula probabilitatea unei secvenţe de vectori acustici într-un model de vorbitor, pentru simplificarea calculelor, vom presupune că aceşti vectori acustici sunt independenţi. Astfel calculul probabilităţii se realizează cu formula

$$p(X|\lambda_i) = \prod_{t=1}^{T} p(x_t|\lambda_i). \tag{3.5}$$

Pentru a evita produsul unor valori subunitare foarte mici, în loc de probabilitate se calculează logaritmul probabilităţii şi se împarte cu T, numărul vectorilor acustici din secvenţa X. Formula exactă utilizată în aplicaţie:

$$\frac{1}{T} \log p(X|\lambda_i) = \frac{1}{T} \sum_{t=1}^{T} \log p(x_t|\lambda_i), \tag{3.6}$$

iar pentru calculul $p(x_t|\lambda_i)$ se utilizează formula 2.25.

Au fost făcute o serie de experimente pentru a testa o parte din algoritmii de grupare prezentaţi în secţiunile precedente, în scopul recunoaşterii vorbitorilor. Publicaţiile relevante în acest domeniu sunt [1, 2, 3, 4]. În [3] experimentele au fost făcute pe o bază de date proprie cu 66 de vorbitori, 37 vorbitori nativi români şi 29 vorbitori nativi maghiari. Din cauza condiţiilor de înregistrare (zgomot şi utilizarea diferitelor tipuri de microfoane) rezultatele au fost sub cele obişnuite (în jur de 80%). Mai mult, deoarece nu au fost date suficiente pentru estimarea

parametrilor modelelor GMM, metoda VQ s-a dovedit mai robustă pentru acest corpus.

Primele rezultate obţinute pe o bază de date publică TIMIT au fost publicate în [2]. În aceste experimente semnalul a fost împărţit în cadre de 32ms cu o suprapunere de 20 de ms şi din fiecare cadru au fost extraşi primii 12 de coeficienţi Mel-cepstrali [40]. Pentru experimente s-au folosit doar 168 de vorbitori dintr-un total de 630 de vorbitori în limba engleză. Antrenarea modelelor vorbitorilor a fost făcută aproximativ pe 24s de vorbire de la fiecare vorbitor, iar pentru testare s-au folosit aproximativ 6s de vorbire de la fiecare vorbitor în parte. Au fost antrenate atât modele VQ cât şi modele GMM, cele GMM în acest caz atingând rata de recunoaştere 100%.

Articolul [4] este de fapt un studiu privind eficienţa diferiţilor algoritmi de grupare pentru identificarea vorbitorilor. Au fost comparaţi algoritmul generalizat K-means, algoritmul de cuantizare vectorială binară, algoritmul LVQ şi GMM. Toate experimentele au fost făcute pe toţi 630 de vorbitori din corpusul TIMIT, obţinându-se pentru fiecare tehnică de grupare o eroare de identificare sub 1%. Parametrii acustici au fost cei obişnuiţi, 12 coeficienţi mel-cepstrali.

Toate studiile efectuate în vederea recunoaşterii vorbitorului au arătat că în cazul când semnalul vocal este înregistrat cu microfoane de calitate, în condiţii de birou, se pot obţine rezultate excepţionale utilizând coeficienţi cepstrali, aproape cu orice algoritm de grupare. Algoritmul de grupare va determina clasele acustice specifice vorbitorului şi aceste clase vor fi reprezentate în mod specific algoritmului, centroid în cazul algoritmului k-means şi valoare medie, respectiv matrice de covarianţă în cazul modelelor GMM. Modelele GMM mai conţin un set de ponderi, care ajută la ponderarea acestor clase acustice.

3.3 Distribuţia conţinutului fonetic

3.3.1 Modele VQ

În acest capitol vom prezenta o analiză statistică a conţinutului fonetic al unui model de vorbitor. Pentru simplitate am ales modelul VQ de vorbitor prezentat în 2.4.4. Acest model este format din centroizi, care reprezintă grupe de vectori acustici similari. Obiectivul nostru este de a verifica, dacă acest model separă vectorii acustici în categorii fonetice, sau nu. Paşii analizei pentru fiecare vorbitor au fost următorii:

1. Am extras vectorii acustici din datele vorbitorului, excluzând porţiunile de

linişte, obţinând mulţimea $X = \{x_1,\ x_2, \ldots, x_T\}, \quad x_i \in R^d$.

2. Am etichetat vectorii acustici cu categoria din care fac parte aceştia şi am obţinut mulţimea $Y = \{(x_1, f_1),\ (x_2, f_2), \ldots, (x_T, f_T)\}$, unde $f_i \in \{A,\ F,\ N,\ S,\ V,\ W\}$. Etichetele din această mulţime reprezintă categoriile fonetice prezentate în tabelul 5.

3. Utilizând mulţimea Y am aplicat un algoritm VQ pentru un număr M de grupe, fixat în prealabil şi am analizat conţinutul acestor grupe din punctul de vedere al categoriilor fonetice.

Figura 3.2 conţine distribuţia categoriilor fonetice în modele de vorbitori formate din 2, 3, 4, 5, respectiv 6 clustere. Prima diagramă arată distribuţia în două clustere. În acest caz primul cluster este populat de nazale, vocale şi semivocale şi cel de-al doilea de africate, fricative şi plozive. Această tendinţă de separare este explicabilă cunoscând proprietăţile acustice ale acestor categorii fonetice. Categoria africatelor este cea mai mică şi vectorii aparţinând acestei categorii fac parte întotdeauna din acelaşi cluster.

Ultima figură conţine exact 6 clustere, cât este şi numărul de categorii de foneme. Analizând această figură se poate vedea că, marea majoritate a fricativelor este concentrată în acelaşi cluster, doar o mică parte a acestora este distribuită în restul clusterelor. Acelaşi lucru este valabil şi pentru categoria nazalelor. Vocalele, semivocalele şi plozivele sunt distribuite în mai multe clustere.

Acest experiment ne arată că, grupând vectorii acustici pe baza similarităţilor dintre acestea, obţinem modele de vorbitori ale căror clustere nu separă vectorii acustici în categorii de foneme.

Varietatea acustică prezentă într-o categorie fonetică se poate caracteriza prin varianţă. Am calculat această varianţă pentru categoriile fonetice ale fiecărui vorbitor şi am determinat media aritmetică pe categorii fonetice. Pentru calcule am utilizat cei 462 de vorbitori din partea de test a bazei de date TIMIT şi vectori acustici cu 12 coeficienţi Mel-cepstrali. Valorile obţinute se pot vedea în tabelul 3.

3.3.2 Modele GMM

În acest experiment am urmărit tendinţa de separare în componente gaussiene ale fonemelor din datele înregistrate de la un vorbitor. Din cauză că, cadrele de la început şi de la sfârşit sunt afectate de coarticulare, am utilizat din fiecare fonem doar un singur cadru, cadrul din mijloc. Fie $X = \{x_1,\ x_2, \ldots, x_T\}$ vectorii

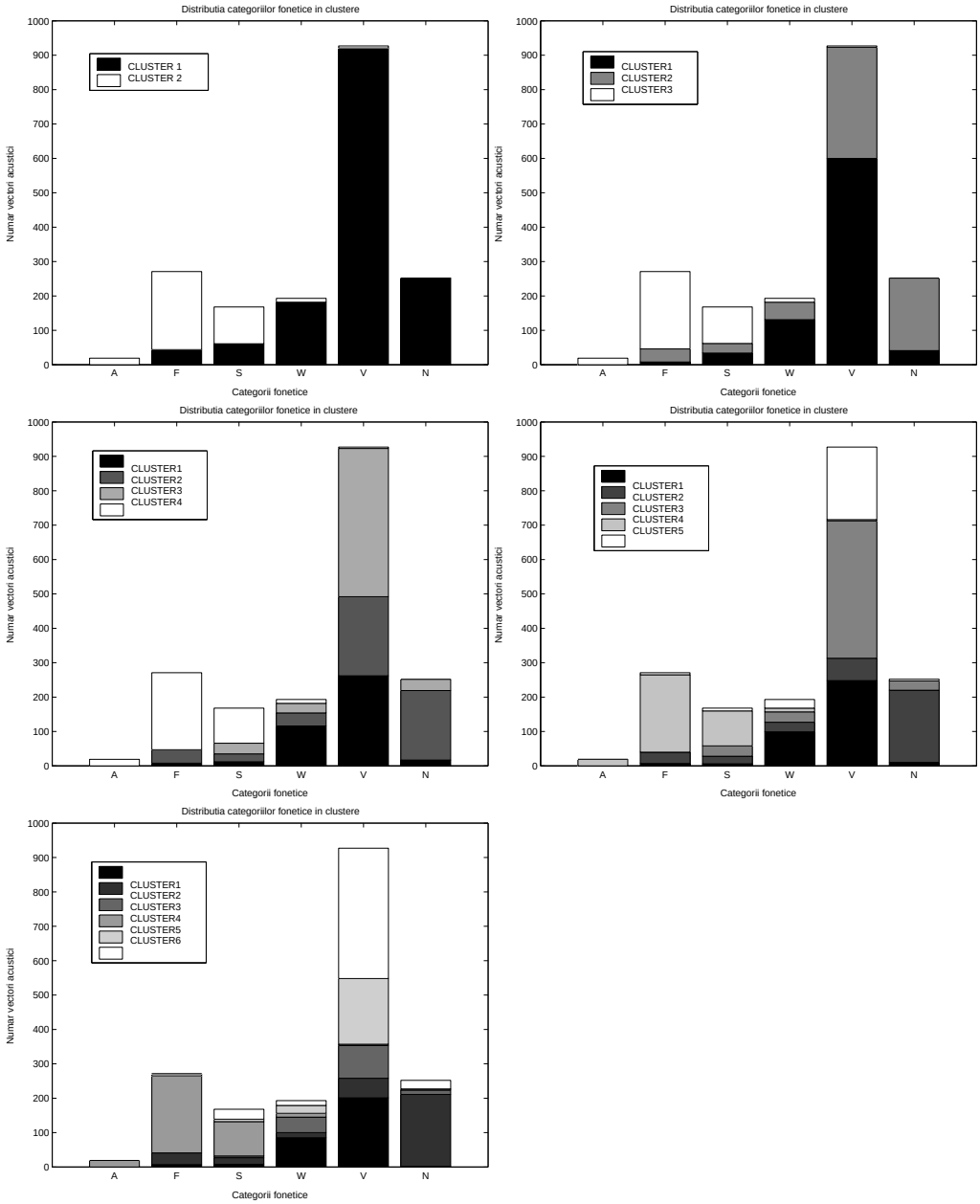


Figura 3.2: Distribuţia categoriilor fonetice în clustere

Categorie fonetică	Varianţă
Africate	23.22
Fricative	47.35
Plozive	37.85
Semivocale	46.76
Vocale	41.33
Nazale	20.53

Tabelul 3: Varianţe pe categorii fonetice

acustici obţinuţi de la un vorbitor, selectând din fiecare fonem doar cadrul din mijloc. Selecţia cadrelor din mijloc este prezentată în figura 3.3.

Am etichetat fiecare vector acustic obţinut dintr-un cadru extras din mijlocul fonemului cu eticheta acestuia. Ca etichete am utilizat codurile celor 39 de grupe de foneme prezentate în tabelul 4 şi am obţinut

$$Y = \{(x_1, f_1),\ (x_2, f_2), \ldots, (x_T, f_T)\}, \tag{3.7}$$

unde $f_i \in \{0,\ 1, \ldots,\ 38\}$. Am creat modele GMM cu câte 32 de componente gaussiene. După crearea modelelor, fiecare vector acustic a fost etichetat cu indicele acelei componente gaussiene care ne-a furnizat probabilitatea maximă pentru acel vector acustic. În final am obţinut mulţimea

$$Z = \{(x_1, f_1, g_1),\ (x_2, f_2, g_2), \ldots, (x_T, f_T, g_T)\}, \tag{3.8}$$

unde $g_i \in \{1,\ 2, \ldots, 32\}$. Mulţimea $\{(f_1, g_1),\ (f_2, g_2), \ldots, (f_T, g_T)\}$a fost reprezentată sub formă de diagramă. Figurile 3.4 şi 3.5 prezintă diagramele astfel obţinute pentru vorbitorul feminin *felc0*, respectiv unul masculin *mdab0*. Diagramele ne arată că, componentele gaussiene nu sunt pure, ele conţinând vectori acustici din mai multe foneme. Plozivele şi fricativele de obicei vor fi în aceeaşi componentă, iar vocalele sunt împrăştiate în mai multe componente.

3.4 Modele GMM pure

Pentru a putea crea modele de vorbitori utilizând doar vectori acustici aparţinând unei categorii fonetice specifice, vectorii acustici extraşi din datele vorbitorilor trebuie etichetaţi cu categoria lor fonetică. Această etichetă se poate obţine în două moduri: prin utilizarea unei baze de date segmentate, cum ar fi TIMIT, care conţine şi segmentarea fonetică a materialului audio sau prin crearea unui

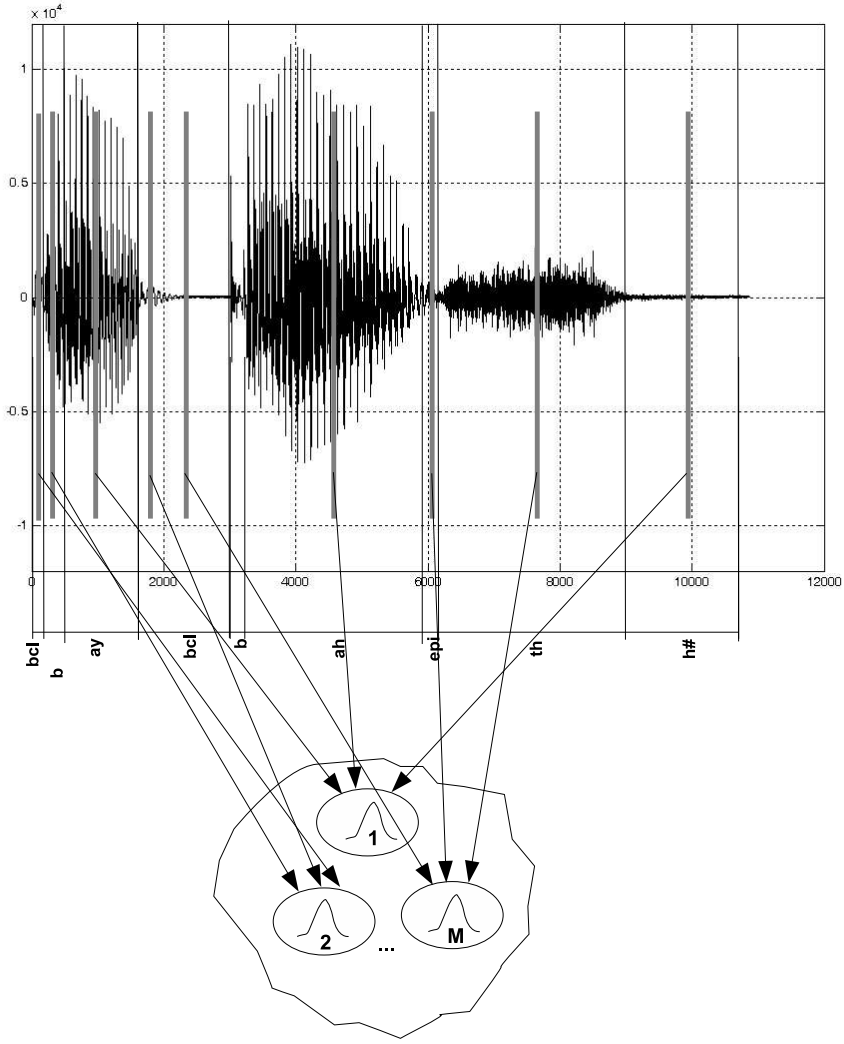


Figura 3.3: Selecţia cadrelor din mijlocul fonemelor

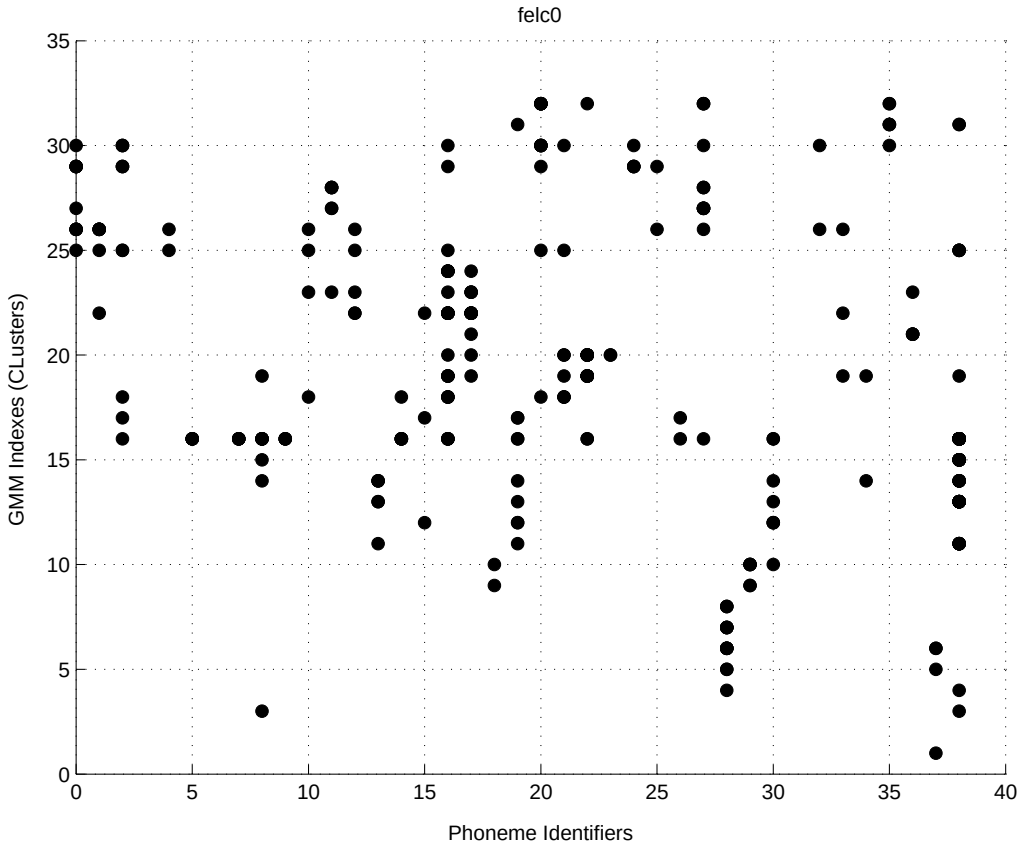


Figura 3.4: felc0-Distribuţia fonemelor în componentele gaussiene ale unui model de vorbitor

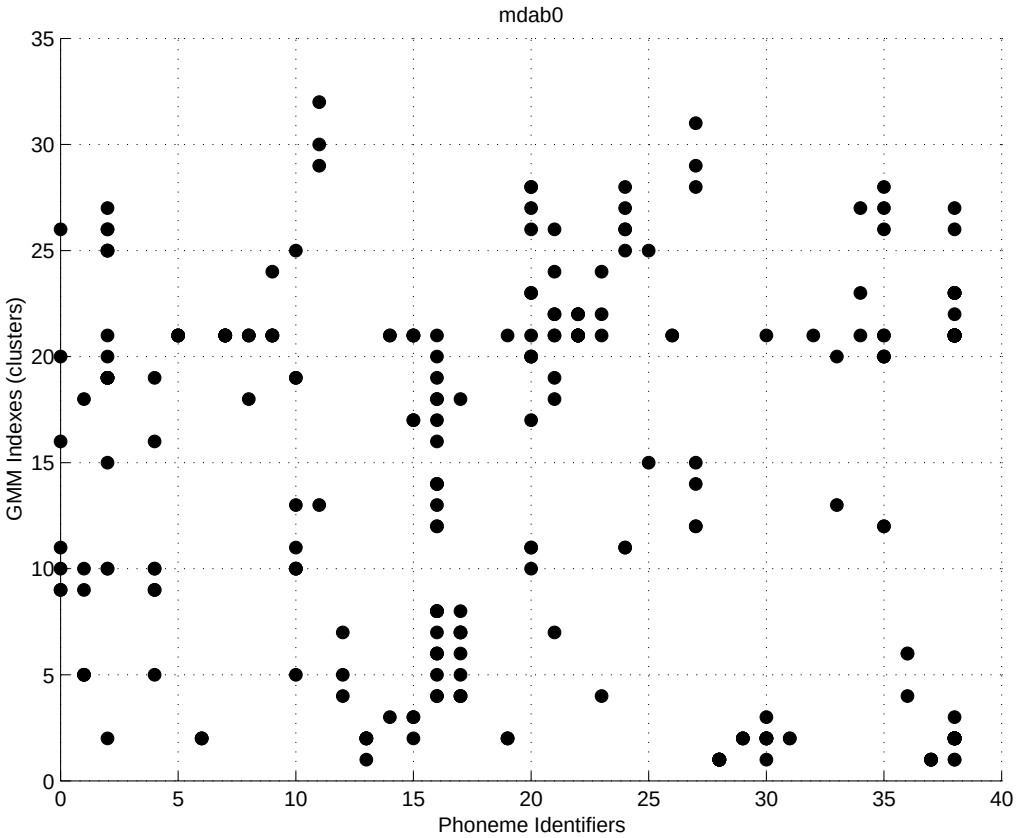


Figura 3.5: mdab0-Distribuţia fonemelor în componentele gaussiene ale unui model de vorbitor

clasificator, care face clasificarea vectorilor acustici în categorii fonetice. Noi am abordat ambele metode, dar în acest capitol, pentru a nu fi nevoiţi să considerăm erorile datorate acestui clasificator, vom prezenta doar rezultate obţinute pentru prima variantă. Acurateţea sistemului nostru de clasificare în categorii fonetice a fost publicată în [7] şi vom reveni la prezentarea acestor rezultate în capitolul 4.

Baza de date TIMIT conţine 61 foneme distincte. Grupând fonemele similare se obţin 39 de grupe, care pot fi grupate în categorii de foneme ca vocale (V), nazale (N), fricative (F), africate (A), plozive (S), semivocale (W) şi linişte, împreună cu părţile de închidere ale plozivelor (O). Pentru notaţia fonemelor se utilizează alfabetul fonetic ARPAbet descris în capitol doi şi sunt prezentate în tabelul 4. Tabelul 5 prezintă numărul fonemelor, respectiv a grupelor de foneme pe categorii.

Am realizat o statistică privind cantitatea medie de date de antrenare, respectiv privind cantitatea medie de date de test medie existent pentru un vorbitor. Aceste date sunt prezentate în tabelul 6. Deoarece sunt vorbitori pentru care africatele lipsesc ori din partea de antrenare, ori din partea de testare, această categorie nu a fost luată în considerare. Pentru fiecare vorbitor am creat câte un model GMM utilizând vectorii acustici etichetaţi cu această categorie. După crearea modelelor utilizând datele de test, am măsurat rata de identificare a vorbitorilor, acesta fiind raportul dintre numărul de vorbitori identificaţi corect şi numărul total de vorbitori. Pentru a determina numărul de componente optim pentru fiecare categorie în parte, la modelarea fiecărei categorii am pornit cu un singur gaussian şi am crescut numărul acestora până când rata de identificare a atins maximul. Tabelul 6 prezintă pentru fiecare categorie în parte numărul de componente optim utilizat, rata de identificare, precum şi un coeficient normalizat de pondere calculat din rata de identificare, care va fi folosit în crearea modelelor GMM structurate fonetic.

3.5 Modele GMM structurate fonetic

Testarea unui model structurat fonetic se desfăşoară conform figurii 3.6. Testarea unui asemenea model se face similar cu testarea modelului standard, adică se calculează probabilitatea fiecărui vector acustic în fiecare componentă gaussiană, se înmulţeşte această probabilitate cu ponderea ataşată fiecărei componente şi se însumează, obţinându-se probabilitatea vectorului acustic în model. Formula

Fonem	Cod	Cod grupă	Categorie	Fonem	Cod	Cod grupă	Categorie
aa	0	0	V	em	18	21	N
ae	1	1	V	en	19	22	N
ah	2	2	V	eng	20	23	N
ao	3	0	V	m	37	21	N
aw	4	3	V	n	38	22	N
ax	5	2	V	ng	39	23	N
ax-h	6	2	V	nx	40	22	N
axr	7	11	V	el	17	20	W
ay	8	4	V	hh	28	15	W
eh	16	10	V	hv	29	15	W
er	22	11	V	l	36	20	W
ey	23	12	V	r	47	27	W
ih	30	16	V	w	57	35	W
ix	31	16	V	y	58	36	W
iy	32	17	V	bcl	21	38	O
ow	41	24	V	dcl	27	38	O
oy	42	25	V	epi	44	38	O
uh	53	32	V	gcl	26	38	O
uw	54	33	V	h#	27	38	O
ux	55	33	V	kcl	35	38	O
ch	11	6	A	pau	44	38	O
jh	33	18	A	pcl	45	38	O
dh	14	8	F	q	46	38	O
f	24	13	F	tcl	51	38	O
s	48	28	F	b	9	5	S
sh	49	29	F	d	12	7	S
th	52	31	F	dx	15	9	S
v	56	34	F	g	25	14	S
z	59	37	F	k	34	19	S
zh	60	29	F	p	43	26	S
				t	50	30	S

Tabelul 4: Foneme TIMIT

Categorie	Număr grupe de foneme	Număr foneme
Vocale (V)	14	20
Nazale (N)	3	7
Fricative (F)	7	8
Semivocale (W)	5	7
Plozive (S)	7	7
Africate (A)	2	2
Altele (O)	1	10

Tabelul 5: Categorii de foneme TIMIT

Categorie	Antrenare	Testare	Nr. mixt.	Rata id.	Pondere
Vocale	9.65s	2.27s	**8**	**95.39%**	0.37
Nazale	1.34s	0.38s	**1**	**70.31%**	0.27
Fricative	3.27s	0.95s	4	44.60%	0.16
Semivocale	2.38s	0.47s	4	41.74%	0.16
Plozive	1.43s	0.35s	4	10.47%	0.04

Tabelul 6: Rata de identificare pentru 630 de vorbitori utilizând GMM-uri fonetic pure

exactă conform figurii 3.6

$$p(x|\lambda) = \sum_{i=1}^{5} w_i p_i(x), \quad \sum_{i=1}^{5} w_i = 1 \tag{3.9}$$

unde $p_i(x)$ este probabilitatea calculată în submodelul GMM corespunzător categoriilor fonetice. Pentru un submodel ataşat unei categorii fonetice probabilitatea se calculează conform formulei următoare

$$p_i(x) = \sum_{k=1}^{n_i} c_{ik} b_{ik}(x), \quad \sum_{k=1}^{n_i} c_{ik} = 1, \tag{3.10}$$

unde $b_{ik}(x)$ este densitatea de probabilitate d-dimensională dată de formula 2.23.

În experimentele efectuate cu aceste modele structurate fonetic, pentru ponderile ataşate submodelelor am folosit două abordări: ponderi egale, ponderi determinate pe baza puterii de discriminare a categoriilor fonetice. Fiecare categorie fonetică a fost modelată cu câte o componentă gaussiană, cu excepţia vocalelor, pentru care am folosit două componente gaussiene, în total şase. În primul experiment, notat cu modele PSGMM1, am utilizat ponderi egale pentru fiecare com-

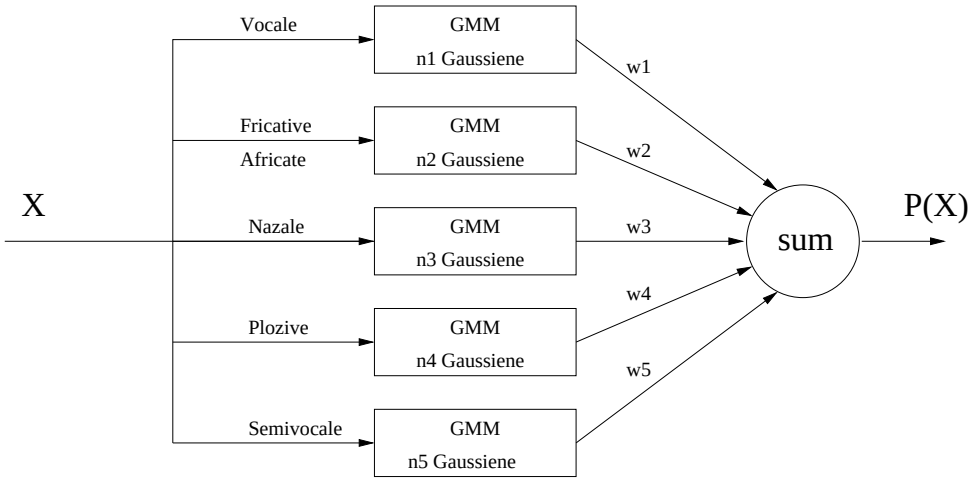


Figura 3.6: Calculul similarităţii într-un model de vorbitor structurat fonetic

ponentă gaussiană, iar în cel de-al doilea experiment, notat cu modele PSGMM2, ponderile utilizate au fost cele din ultima coloană a tabelului 6. Deoarece vocalele au fost modelate cu două componente gaussiene, ponderile obţinute pentru aceste două componente, c_{11} şi c_{12}, au fost înmulţite cu $w_1 = 0.37$ satisfăcându-se astfel toate constrângerile necesare pentru model. Deoarece în partea de test a materialului vocal am avut două propoziţii rostite, am repetat experimentul câte o dată pentru fiecare propoziţie şi am calculat media aritmetică a ratei de recunoaştere. Propoziţiile rostite au o durată medie de 3 secunde. Experimentele au fost repetate şi pentru cele două propoziţii, obţinându-se în acest caz o îmbunătăţire semnificativă a ratei de recunoaştere. Pentru comparaţie am inclus şi rata de recunoaştere obţinută pentru un model standard GMM având acelaşi număr de componente, adică 6.

Tabelul 7 prezintă rezultatele comparative.

Tip model	Nr.mixt.	Rata id. 3s test	Rata id. 6s test
GMM	6	93.02%	98.58%
PSGMM1	6	92.14%	98.74%
PSGMM2	6	92.06%	98.16%

Tabelul 7: Rate de recunoaştere pentru modele de vorbitori structurate fonetic - TIMIT 630 de vorbitori

Aşa cum arată rezultatele, utilizând doar informaţii relative la categoria fonetică a vectorilor acustici, modelele create au performanţe similare cu cele standard.

Am realizat un studiu cu privire la vorbitorii identificaţi în mod incorect utilizând modelele standard GMM şi cele două modele structurate fonetic. Tabelul 8 prezintă lista vorbitorilor identificaţi în mod incorect. Aşa cum se poate vedea, ponderile nu influenţează în mod semnificativ rata recunoaşterii, mai mult, listele vorbitorilor identificaţi în mod greşit de către metodele PSGMM1 şi PSGMM2 sunt similare. Nu acelaşi lucru se poate spune, dacă comparăm modelul standard cu cel structurat fonetic. În acest caz listele conţin vorbitori diferiţi. De aici rezultă că, există posibilitatea de a combina cele două modele.

GMM	PSGMM1	PSGMM2
mgwt0	fram1	fram1
mmdm2	mcrc0	mjvw0
mthc0	mrdd0	mcrc0
mwar0	fdxw0	mrfk0
fmmh0	mdps0	mdlh0
mlbc0	mrfk0	fkkh0
fkkh0	fjxp0	flod0
mjpg0	fkkh0	mrmh0
mcre0	flod0	
	mcre0	
	mrmh0	

Tabelul 8: Lista identificatorilor vorbitorilor identificaţi în mod incorect

Pentru a putea caracteriza cele două tipuri de modele de vorbitori, am reprezentat probabilităţile cu care au fost identificaţi vorbitorii (vezi figura 3.7). Am ales o listă de vorbitori identificaţi în mod corect de ambele metode. Aşa cum rezultă şi din figură, modelul standard (GMM) indică aproape întotdeauna o probabilitate mai mare, decât modelele structurate fonetic (PSGMM1, PSGMM2).

3.6 Concluzii

În aceast capitol am prezentat diferite tehnici de modelare a vorbitorului. Toate modelele prezentate în acest capitol au fost implementate de către autor şi testate pe baze de date publice. Publicaţii relevante pentru această problemă sunt:

[1, 3]: Compară metodele GMM şi VQ pentru identificarea vorbitorului pe o bază de date proprie cu 66 de vorbitori. Deoarece nu au fost date suficiente pentru estimarea robustă a parametrilor modelului GMM, în acest caz modelele VQ au avut o performanţă uşor ridicată faţă de performanţele modelelor GMM.

[2]: Compară metodele GMM şi VQ pentru 168 de vorbitori din baza de date

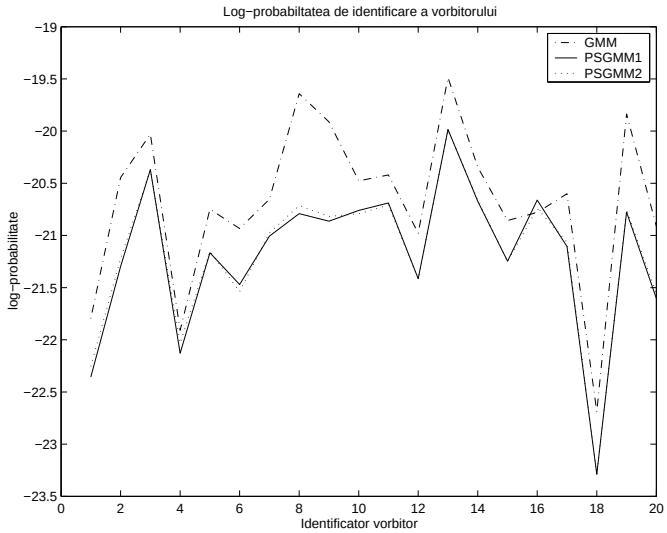


Figura 3.7: Probabilităţi de identificare a vorbitorilor pentru primii 20 de vorbitori

TIMIT. Modelele GMM s-au dovedit superioare în acest caz.

[4]: Lucrararea prezintă comparaţia diferiţilor algoritmi de clustering pentru recunoaşterea vorbitorului. Au fost comparate algoritmii: Randomized Generalized Lloyd Algorithm (RGLA), Modified Generalized Lloyd Algorithm (MGLA), Learning Vector Quantization (LVQ) şi Gaussian Mixture Models (GMM). Concluzia acestei lucrări ar fi că, orice algoritm de clustering implementat în mod adecvat este corespunzător pentru modelarea vorbitorilor. Contează mai mult iniţializarea clusterelor, decât algoritmul în sine.

[6]: Lucrarea studiază puterea de discriminare a categoriilor fonetice pe un număr de 100 de vorbitori din baza de date TIMIT.

[8]: În această lucrare am comparat coeficienţii mel cepstrali (MFCC) cu coeficienţii cepstrali obţinuţi din parametrii LPC (LPCC). Toate măsurătorile au fost făcute pentru toţi cei 630 de vorbitori din baza de date TIMIT. Am studiat şi efectul utilizării doar a cadrelor din mijloc din fiecare fonem asupra recunoaşterii vorbitorului. Am verificat, dacă clusterele unui model de vorbitor sunt sau nu fonetic pure.

Paramerii LPCC s-au dovedit uşor mai bune decât parametrii MFCC. Utilizând doar un singur cadru din fiecare fonem rezultatele au scăzut doar cu un singur procent faţă de utilizarea tuturor cadrelor. În sfârşit am arătat că un model de vorbitor nu separă fonemele vorbitorului în clustere diferite, cu toate că fiecare fonem are clusterele specifice. De obicei vocalele sunt împrăştiate în mai multe

clustere, iar plozivele şi fricativele în maxim două clustere. În acest experiment numărul clusterelor a fost 32 şi cel cel al fonemelor 39.

[9, 11]: În aceste două lucrări am studiat puterea de discriminare a categoriilor fonetice. În acest scop am creat modele pure de vorbitori, adică conţinând vectori acustci dintr-o singură categorie fonetică. Am ordonat categoriile fonetice din punct de vedere al puterii de discriminare al vorbitorului. Nazalele şi vocalele s-au dovedit a fi cele mai bune din acest punct de vedere.

Rezultatele obţinute pentru modele de vorbitori structurate fonetic urmează să fie publicate.

Capitolul 4

Clasificarea fonemelor cu metode statistice

4.1 Introducere

Pentru a construi un sistem de recunoaştere automată a vorbirii este necesară modelarea unei părţi a vorbirii, din care se poate compune orice propoziţie iar numărul acestor părţi de vorbire să fie relativ restrâns. S-au propus mai multe asemenea unităţi, cum ar fi silaba, demisilaba, fenonul. Sistemul TANGORA de la IBM este bazat pe modele de tip fenon [25]. Totuşi, majoritatea sistemelor comerciale pentru recunoaşterea vorbirii sunt bazate pe foneme. Pentru a modela coarticularea prezentă în vorbire se utilizează trifoneme, adică triplete de foneme. Aceste modele se numesc şi modele dependente de context, deoarece modelează nu numai fonemul ci şi contextul acestuia. Într-un sistem de recunoaşterea vorbirii o componentă utilă ar fi recunoaşterea fonemelor. Recunoaşterea însă, pe lângă modelarea fonemelor, necesită şi tehnici de căutare prezentate în capitolul 2.6.4. Clasificarea fonemelor este utilă pentru evaluarea modelelor. Dacă la acestea se adaugă tehnici corespunzătoare de căutare, se poate ajunge şi la recunoaşterea fonemelor.

Au fost publicate câteva studii prezentând rezultate privind clasificarea fonemelor pe baza de date TIMIT utilizând modele neuronale recurente [73] sau modele Markov ascunse [22]. Majoritatea articolelor prezintă şi rezultate experimentale pentru recunoaşterea fonemelor [51, 73, 78]. Clasificarea presupune cunoaşterea capetelor fonemelor, iar recunoaşterea nu are informaţii legate de poziţia fonemelor în semnalul vocal. Bineînţeles, recunoaşterea este o problemă mai complexă.

Un studiu propriu, care efectuează clasificarea fonemelor din baza de date TIMIT este [5], iar articolul [7] propune o nouă schemă de clasificare pe două niveluri. În continuare descriem modelele folosite în articolele menţionate cu rezultate experimentale uşor îmbunătăţite faţă de cele publicate. Aceste îmbunătăţiri se datorează faptului că a fost îmbunătăţită modalitatea de calcul a diferenţelor de ordin unu şi doi ale coeficienţilor cepstrali.

4.2 Analiza acustică

Forma de undă a fost împărţită pe cadre scurte de 20 de milisecunde cu o suprapunere de 50% între cadrele consecutive. Din fiecare cadru am extras 13 coeficienţi Mel-cepstrali, conform procedurii descrise în [40]. Am utilizat 28 de filtre triangulare, trece-bandă, aşezate conform scalei Mel. Am calculat coeficienţii cepstrali cu formula

$$C_p = \sum_{r=1}^{28} S_r cos\left(p \cdot \left(r - \frac{1}{2}\right) \cdot \frac{\pi}{28}\right), \quad 0 \leq p < 28 \tag{4.1}$$

unde S_r reprezintă logaritmul energiei la ieşirea filtrului cu numărul de ordine r. Am utilizat doar primii 13 coeficienţi, C_0, $C_1, \ldots, C_{12}$. Coeficienţii dinamici au fost obţinuţi cu următoarea formulă

$$\Delta C_m(t) = \frac{\sum_{k=-K}^{K} kC_m(t+k)}{\sum_{k=-K}^{K} k^2} \tag{4.2}$$

În formula 4.2 pentru constanta K am utilizat valoarea 2. Coeficienţii cepstrali împreună cu derivatele de ordin unu şi doi, în total 39 de coeficienţi, au format un vector acustic.

4.3 Evaluarea clasificării

Clasificarea a fost efectuată pe cele două baze de date segmentate TIMIT şi OASIS. Rata de recunoaştere pentru un fonem este dată de formula

$$R_k = \frac{numar\_foneme\_k\_corect\_clasificate}{numar\_total\_foneme\_k} \tag{4.3}$$

iar rata globală de recunoaştere de formula

$$R_{tot} = \frac{numar\_foneme\_corect\_clasificate}{numar\_total\_foneme} \tag{4.4}$$

4.4 Rezultate experimentale

4.4.1 TIMIT

Baza de date TIMIT conţine 61 foneme distincte. Grupând fonemele similare se obţin 39 de grupe, care pot fi grupate în categorii de foneme ca vocale (V), nazale (N), fricative (F), africate (A), plozive (S), semivocale (W) şi linişte împreună cu părţile de închidere ale plozivelor (O). Pentru notaţia fonemelor se utilizează alfabetul fonetic ARPANET descris în capitolul 2 şi este prezentat în tabelul 4. Ca şi modele au fost utilizate modele GMM, valorile medii fiind iniţializate cu valorile medii determinate de algoritmul de cuantizare vectorială binară (alg. 5). Măsurătorile au fost făcute odată pe setul de 61 foneme, după care pe setul de 39 de foneme. Modelele diferitelor foneme au avut acelaşi număr de componenete, acestea variind între 1 şi 32. Rezultatele obţinute pentru cele două seturi de foneme sunt prezentate în tabelele 10 şi 11. Toate experimentele au fost făcute pe întreaga bază de date, adică antrenarea făcându-se cu fonemele a 462 vorbitori, iar evaluarea cu datele celor 168 de vorbitori rămaşi. Numărul exact de foneme din setul de test şi cel de antrenare este prezentat în tabelul 9.

Figurile 4.1, 4.2 şi 4.3 prezintă ratele de clasificare obţinute pentru cele 61 de foneme din baza de date TIMIT cu 8, 16 respectiv 32 de componente.

Din matricea de confuzie am determinat erorile cele mai frecvente. Tabelul 12 conţine primele 10, cele mai frecvente erori. Aceste erori sunt similare cu cele publicte în [73], obţinute cu reţele neurale recurente. După eticheta fonemului, simbolul din paranteză reprezintă categoria fonemului.

Pe baza tabelului 12 şi a matricei de confuzie (vezi figura 4.4), din care reiese că majoritatea fonemelor clasificate incorect au fost confundate cu alte foneme din propria categorie, se propune o clasificare pe două niveluri prezentată în figura 4.5.

Pentru a realiza clasificatorul pe două niveluri, în primul pas am creat un clasificator pentru cele şapte categorii de foneme. Categoriile de foneme au fost modelate cu mixturi gaussiene, odată cu 32 de componente şi încă odată cu 64 de componente. Performanţele acestui clasificator sunt prezentate în tabelul 13 şi în figura 4.6. Ratele cele mai bune de clasificare au furnizat categoriile fonetice: africate, nazale şi vocale.

Rezultatele obţinute cu clasificatorul pe două niveluri sunt prezentate în tabelul 14. În paranteză sunt ratele de clasificare obţinute cu metoda obişnuită, prezentate şi în tabelul 10. Aşa cum arată aceste rezultate, utilizând un clasificator cu două niveluri, performanţa sistemului nu scade în mod semnificativ, dar

Fonem	Test	Antr.	Fonem	Test	Antr.	Fonem	Test	Antr.
aa	1133	3064	eng	5	38	nx	360	971
ae	1407	3997	epi	536	1464	ow	777	2136
ah	879	2306	er	800	2046	oy	263	684
ao	1156	2940	ey	806	2282	p	956	2588
aw	216	729	f	912	2216	pau	391	952
ax	1346	3610	g	750	2017	pcl	965	2644
ax-h	118	375	gcl	808	2223	q	1243	3590
axr	1383	3407	h#	3360	9240	r	2525	6539
ay	852	2390	hh	356	957	s	2639	7475
b	878	2181	hv	369	1154	sh	796	2238
bcl	776	1909	ih	1709	5051	t	1534	4364
ch	259	822	ix	2945	8642	tcl	2334	6644
d	1242	3548	iy	2710	6953	th	267	751
dcl	1643	4942	jh	372	1209	uh	221	535
dh	1053	2826	k	1614	4874	uw	170	555
dx	940	2709	kcl	1964	5859	ux	580	1908
eh	1440	3853	l	2356	5801	v	710	1994
el	343	951	m	1524	3903	w	1239	3140
em	47	124	n	2501	7068	y	634	1715
en	251	723	ng	414	1330	z	867	3773
						zh	44	151

Tabelul 9: Frecvenţa fonemelor în mulţimea de test şi cea de antrenare

Număr componente	Rata de recunoaştere
1	45.55%
2	48.95%
4	52.98%
8	57.61%
16	60.84%
32	**63.34%**

Tabelul 10: Rata de recunoaştere pentru setul TIMIT-61

Număr de componente	Rata de recunoaştere
1	52.68%
2	55.12%
4	58.81%
8	63.09%
16	65.48%
32	**68.13%**

Tabelul 11: Rata de recunoaştere pentru setul TIMIT-39

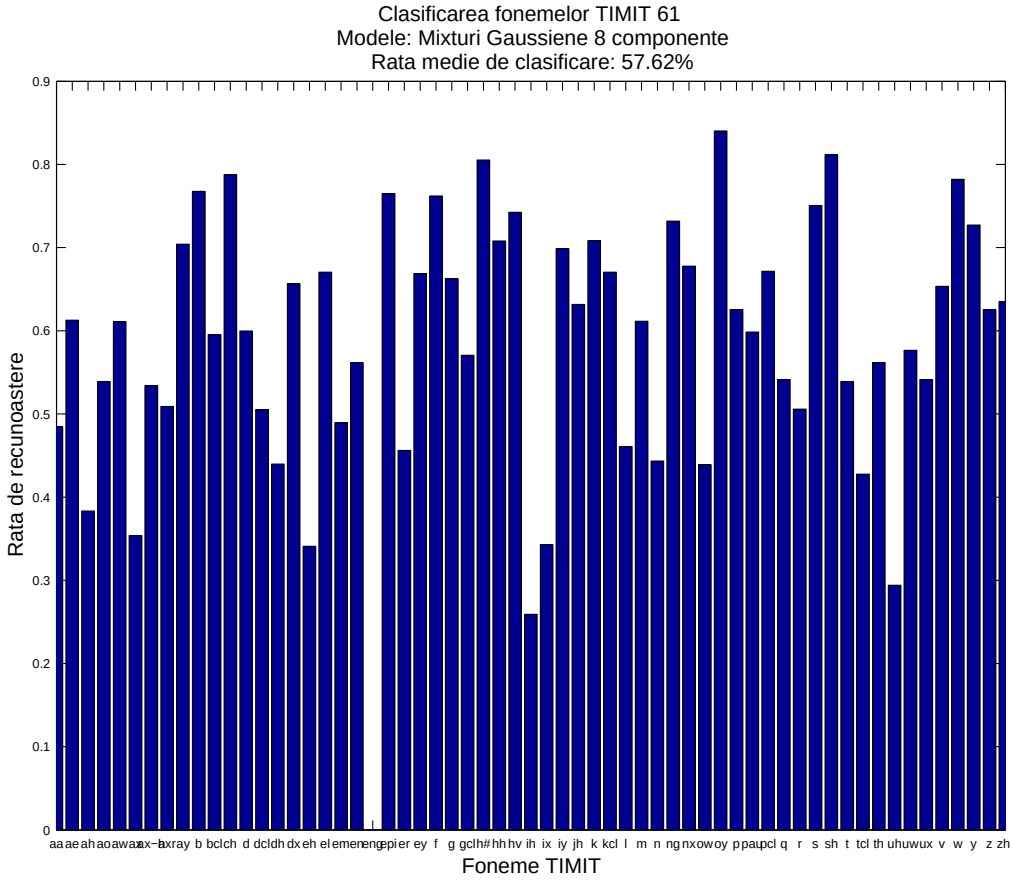


Figura 4.1: Clasificare foneme - TIMIT 61 - GMM 8

Nr.	Fonem iniţial	Fonem recunoscut	Procent de eroare
1	/ix/ (V)	/ih/ (V)	1.59%
2	/r/ (W)	/er/ (V)	1.39%
3	/ix/ (V)	/ax/ (V)	1.35%
4	/s/ (F)	/z/ (F)	1.29%
5	/z/ (F)	/s/ (F)	1.29%
6	/ih/ (V)	/ix/ (V)	1.25%
7	/eh/ (V)	/ae/ (V)	0.95%
8	/eh/(V)	/ah/ (V)	0.95%
9	/n/ (N)	/m/ (N)	0.95%
10	/r/ (W)	/axr/ (V)	0.95%

Tabelul 12: Cele mai frecvente erori - TIMIT 61

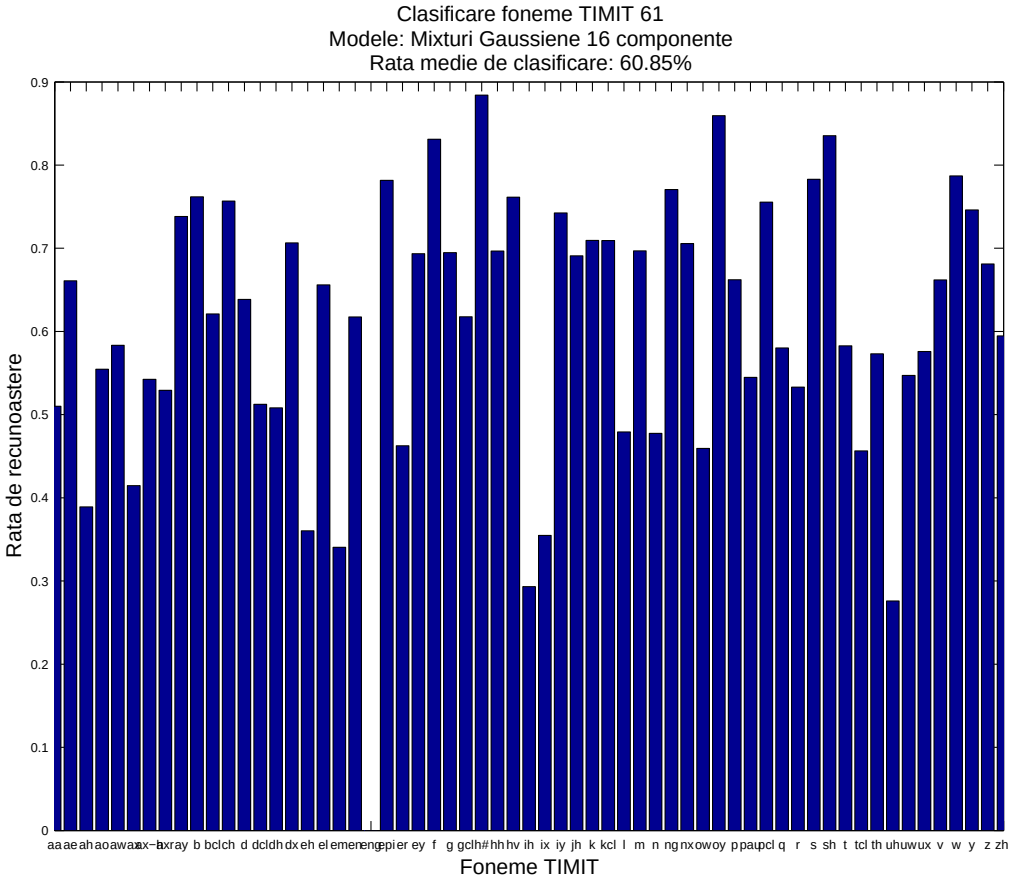


Figura 4.2: Clasificare foneme - TIMIT 61 - GMM 16

Categorie fonetică	GMM 32	GMM 64
Vocale (V)	83.99%	85.10%
Nazale (N)	88.85%	89.95%
Fricative (F)	78.82%	81.20%
Semivocale (W)	77.26%	80.02%
Plozive (S)	82.46%	84.80%
Africate (A)	92.55%	92.55%
Linişte (O)	87.90%	88.30%
Medie	**83.68%**	**85.13%**

Tabelul 13: Clasificare categorii fonetice TIMIT

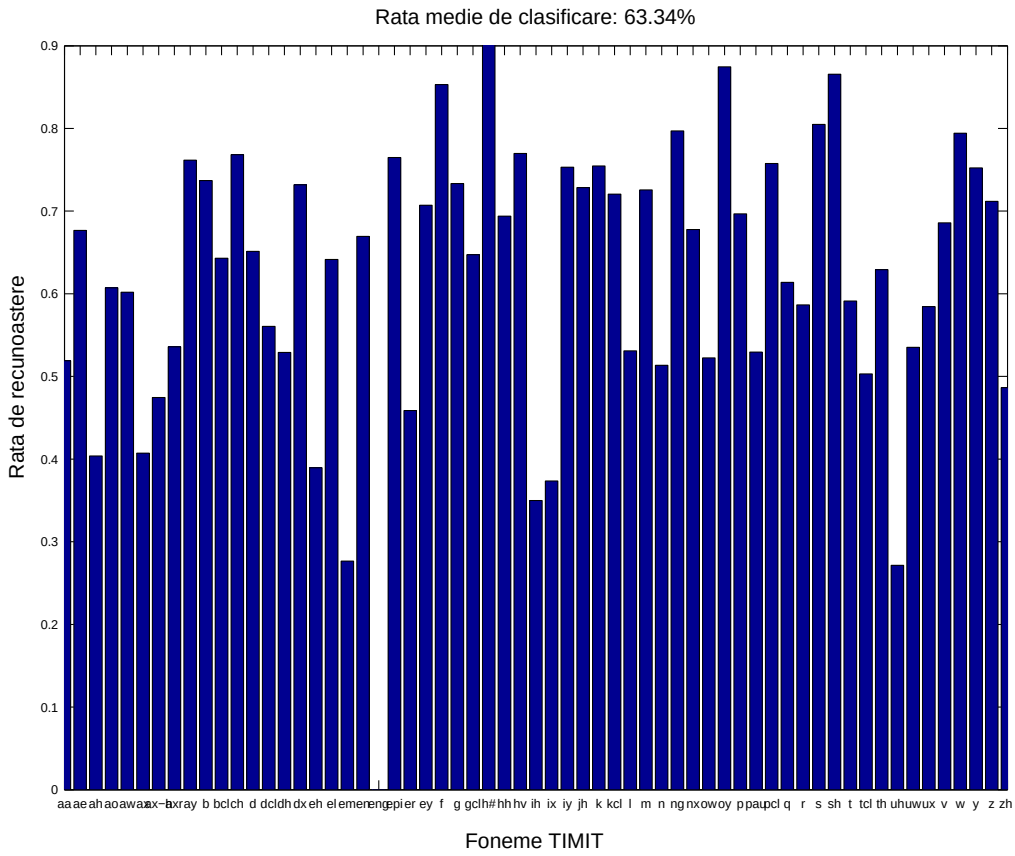


Figura 4.3: Clasificare foneme - TIMIT 61 - GMM 32

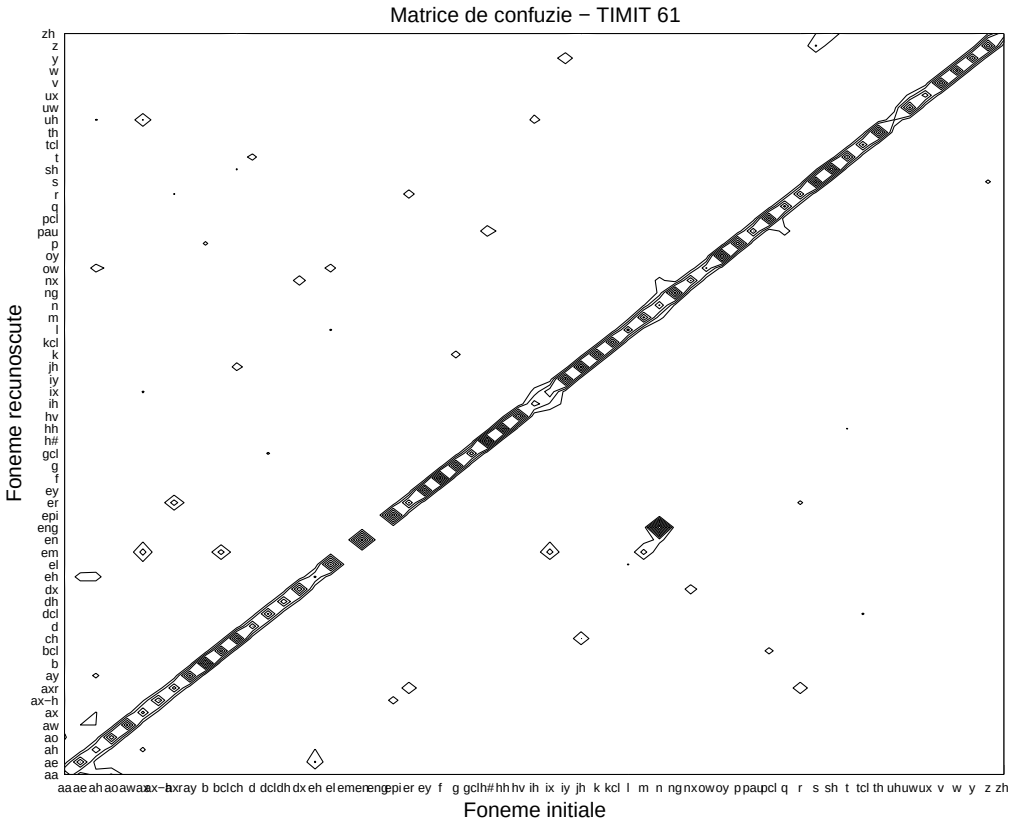


Figura 4.4: Matricea de confuzie TIMIT 61

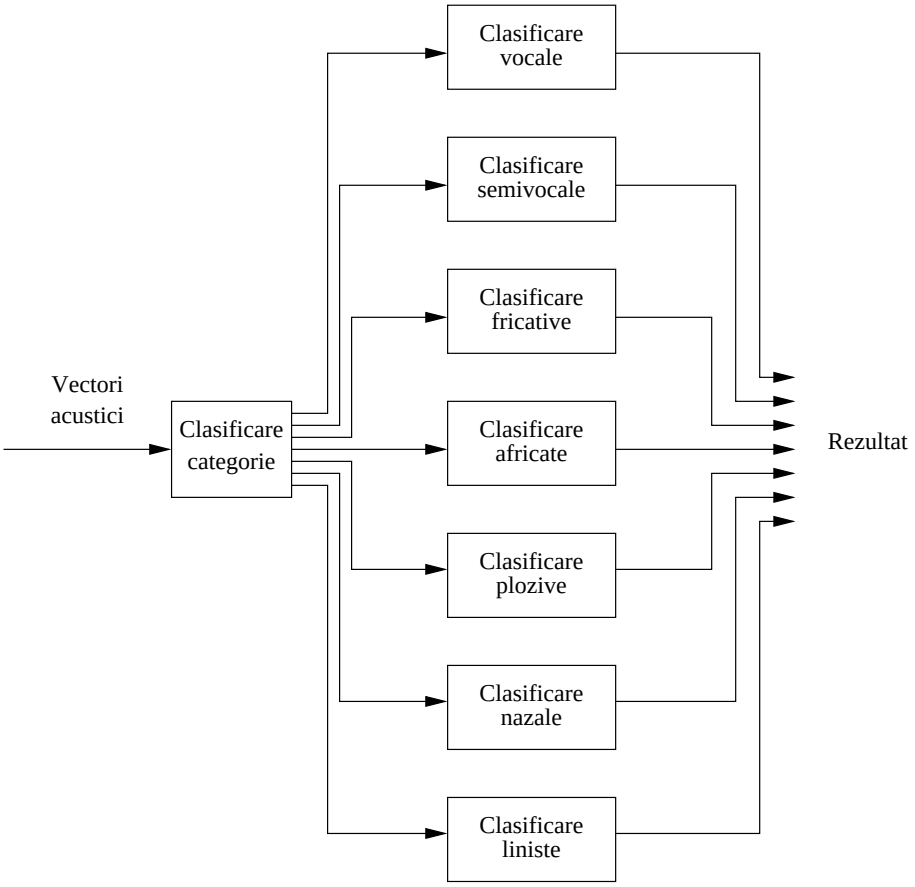


Figura 4.5: Clasificator cu două niveluri

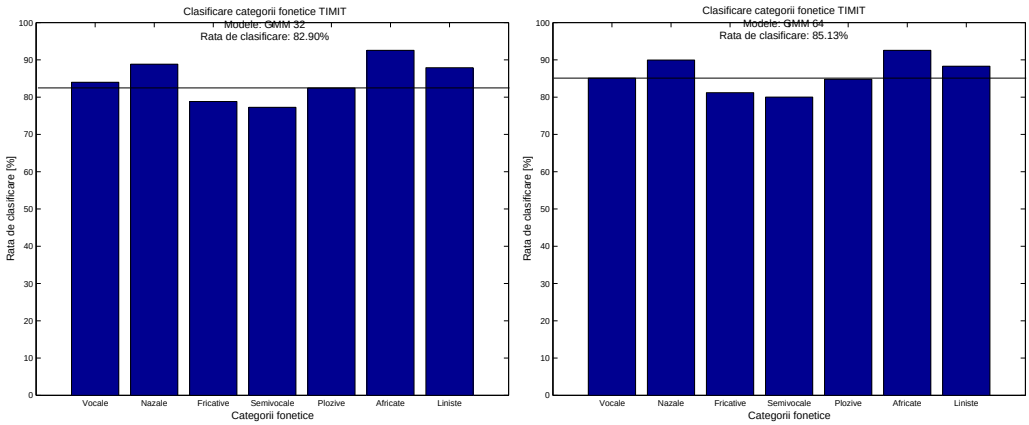


Figura 4.6: Clasificare categorii fonetice TIMIT

timpul de procesare se reduce considerabil. În cazul vocalelor se vor face 7+20 de comparaţii în loc de 61. În primul nivel întotdeauna se fac 7 comparaţii, deoarece în total avem şapte categorii, iar pe al doilea nivel numărul comparaţiilor depinde de numărul fonemelor din categoria respectivă. În total sunt 20 de vocale. În cazul nazalelor se vor face 7+7 comparaţii, iar în cazul africatelor 7+2 comparaţii.

	Model categ. GMM 32	Model categ. GMM 64
Model fonem GMM 8	55.51% (57.61%)	56.64% (57.61%)
Model fonem GMM 16	58.00% (60.84%)	59.21% (60.84%)
Model fonem GMM 32	59.61% (63.34%)	60.84% (63.34%)

Tabelul 14: Clasificare foneme TIMIT 61 cu clasificatorul pe două niveluri

Ultimele experimente au fost făcute pentru clasificarea cadrelor (vectorilor acustici). Acest tip de clasificare se mai numeşte şi clasificare cadru cu cadru (frame by frame classification). Şi acest tip de clasificare urmăreşte evaluarea modelelor. Experimentul a fost repetat de trei ori. Prima dată s-a folosit câte un cadru, după aceea am crescut cu doi numărul cadrelor din secvenţa de clasificat. Tabelul 15 prezintă rezultatele obţinute în cazul modelelor gaussiene cu 32 componente.

Nr. cadre	Rată de clasificare cadru cu cadru
1	44.29%
3	46.88%
5	49.21%
7	49.30%
9	48.27%

Tabelul 15: Clasificare cadru cu cadru-TIMIT 61

4.4.2 OASIS Numbers

Rezultatele prezentate în această lucrare se referă la toate cele 31 foneme, chiar dacă câteva pot fi fuzionate, reprezentând alofoanele aceluiaşi fonem. În articolele [48, 76] rezultatele sunt prezentate referitor la cele 28 de foneme. Rezultatele noastre în cazul aplicării metodei GMM sunt mai bune decât cele prezentate în [48] (79.20%) cu aceeaşi metodă, deşi ei au lucrat doar cu 28 de foneme, grupând alofoanele. Cele mai bune rezultate au fost obţinute de ei prin aplicarea transformării LDA pentru vectorii acustici (86.76%). În [76] modelarea fonemelor a fost realizată cu pachetul HTK, utilizând modele Markov continue cu 3 stări ascunse

şi obţinând o rată de recunoaştere de 90.48%.

Tabelul 16 prezintă frecvenţele de apariţie a diferitelor foneme în partea de antrenare respectiv testare.

Fonem	Antrenare	Test	Fonem	Antrenare	Test
-	519	156	i:	40	12
-:	40	12	j	80	24
'd'	118	36	J	80	24
'k	200	60	l	200	60
't	399	120	l:	40	12
'ts	200	60	m	120	36
+	120	36	n	559	168
~	2080	624	O	240	72
2	80	24	o	160	48
:2	40	12	o:	40	12
A:	120	36	r	160	48
E	600	180	s	160	48
e:	160	48	u	80	24
h	306	96	u:	40	12
i	240	72	v	240	72
			z	240	72

Tabelul 16: Frecvenţa fonemelor în mulţimea de antrenare şi test

Tabelul 17 prezintă ratele de clasificare obţinute pe această bază de date folosind ca modele de foneme mixturi gaussiene cu 1, 2, 4, 8, 16 şi 32 de componente.

Figurile 4.7 şi 4.8 prezintă matricea de confuzie respectiv ratele de clasificare pentru cele 31 de foneme OASIS obţinute cu modele având 16 componente gaussiene.

Din matricea de confuzie am determinat erorile cele mai frecvente. Tabelul 18 conţine primele 20, cele mai frecvente erori. Putem constata că erorile cele mai frecvente sunt între foneme, care au câte două forme, una scurtă şi una lungă. În cazul limbilor finugorice fiecare vocală are atât formă lungă cât şi formă scurtă. Pentru recunoaşterea acestor limbaje, modelarea duratei fonemelor este o problemă critică.

Ultimele experimente au fost făcute pentru clasificarea cadrelor (vectorilor acustici). Acest tip de clasificare se mai numeşte şi clasificare cadru cu cadru (frame by frame classification). Şi acest tip de clasificare urmăreşte evaluarea modelelor. Experimentul a fost repetat de cinci ori. Prima dată s-a folosit câte un cadru, după aceea am crescut cu doi numărul cadrelor din secvenţa de clasificat. Cadrul din mijloc determină întotdeauna apartenenţa la o fonemă a secvenţei de

Număr componente	Rata de recunoaştere
1	80.40%
2	87.78%
4	90.84%
8	92.27%
16	**92.74%**
32	90.84%

Tabelul 17: Clasificare foneme OASIS 31

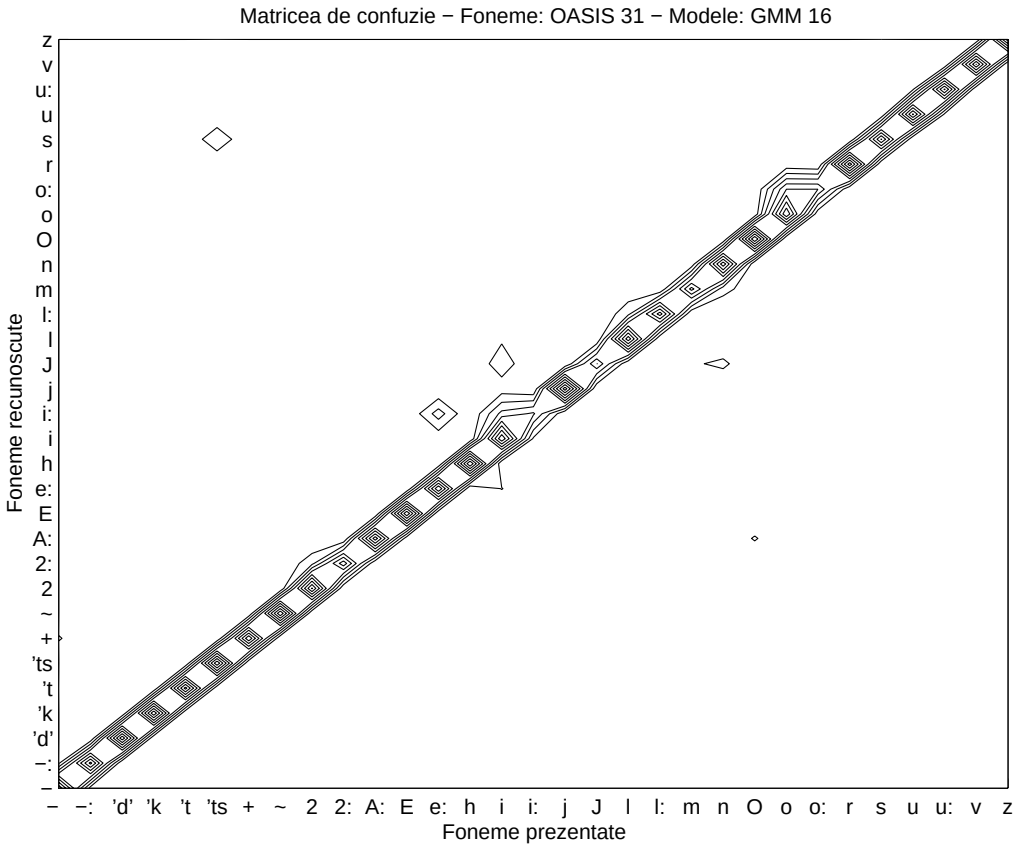


Figura 4.7: Matricea de confuzie OASIS 31

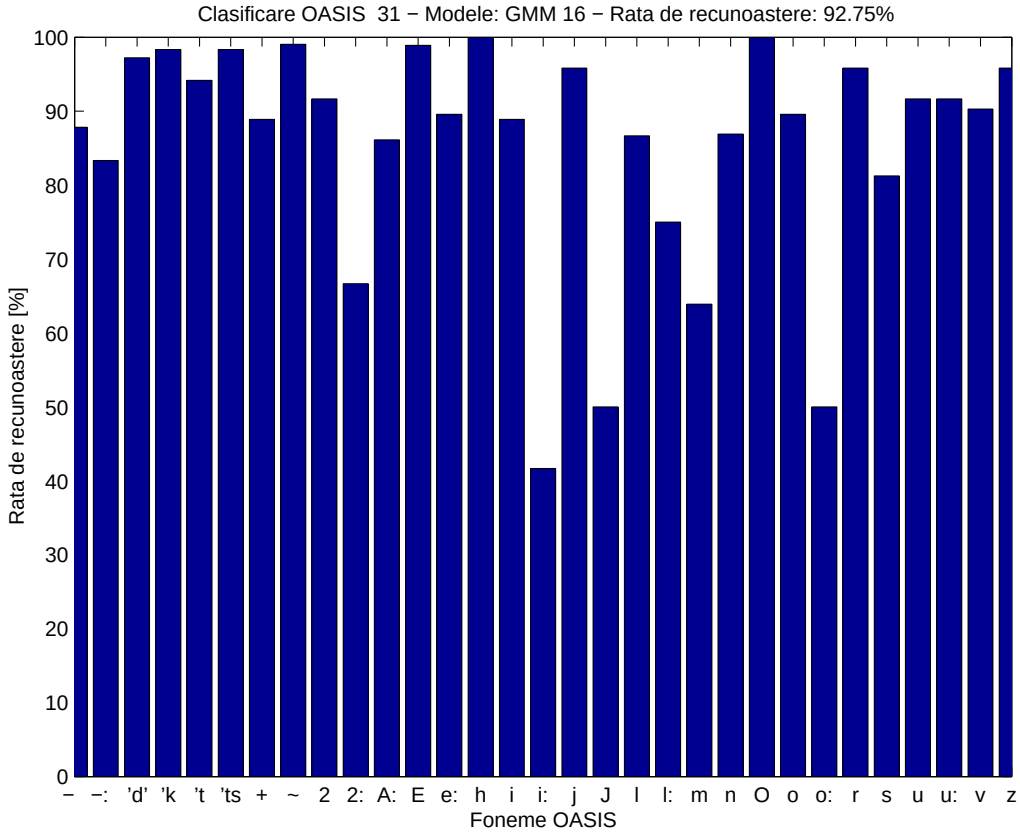


Figura 4.8: Ratele de clasificare OASIS 31, modele cu 16 componente gaussiene

Nr.	Fonem iniţial	Fonem recunoscut	Procent de eroare
1.	/-/	/-:/	5.35%
2.	/s/	/'ts/	5.35%
3.	/i/	/i:/	3.57%
4.	/o:/	/o/	3.57%
5.	/e:/	/i/	2.97%
6.	/l/	/i/	2.97%
7.	/m/	/n/	2.97%
8.	/-/	/'ts/	2.38%
9.	-	+	2.38%
10.	+	-	2.38%
11.	/A:/	/O/	2.38%
12.	/i:/	/i/	2.38%
13.	/J/	/i/	2.38%
14.	/n/	/~/	2.38%
15.	/n/	/i/	2.38%
16.	/n/	/m/	2.38%
17.	/'t/	/-/	1.78%
18.	/~/	/h/	1.78%
19.	/i:/	/e:/	1.78%
20.	/J/	/n/	1.78%

Tabelul 18: Cele mai frecvente erori - TIMIT 61

cadre. Tabelul 19 prezintă rezultatele obţinute în cazul modelelor gaussiene cu 16 componente.

Număr cadre	Rată de clasificare
1	63.36%
3	70.44%
5	74.03%
7	76.10%
9	77.36%

Tabelul 19: Clasificare cadru cu cadru - OASIS 31

4.5 Concluzii

Clasificarea fonemelor se utilizează în recunoaşterea vorbirii pentru evaluarea modelelor. Cu modele de foneme performante avem şansa de a construi sisteme de recunoaşterea vorbirii cu o acurateţe ridicată. În acest capitol am prezentat sistemul realizat de autor pentru clasificarea fonemelor. Ca şi modele de foneme am utilizat modele GMM şi am arătat că aceste modele ating performanţele modelelor HMM cu 3 stări. Cu aceasta am demonstrat faptul că probabilităţile de tranziţie între stările unui model Markov ascuns sunt nesemnificative comparativ cu probabilităţile de emisie ale stărilor.

Pentru clasificarea fonemelor din baza de date TIMIT, toate articolele prezintă rezultate obţinute pe setul de 39 de foneme, de aceea am putut compara doar rezultatele obţinute pe această mulţime. Tabelul 20 este un tabel comparativ care se referă la clasificarea celor 39 de grupuri de foneme.

Articol	Vector acustic	Model	#mixt.	Clasif.
[22]	MFCC26	HMM-3	15	58.97%
[56]	LPCC24	HMM-3	24	57.10%
[61]	MFCC36	HMM-3	nespec.	63.00%
Această lucrare	**MFCC39**	**GMM**	**32**	**68.13%**

Tabelul 20: Rezultate comparative pentru clasificarea fonemelor TIMIT 39

În afară de clasificarea fonemelor şi a grupurilor de foneme am clasificat şi categoriile fonetice. Toate rezultatele au fost publicate în articolele [5, 7]. Articolul [7] introduce un nou tip de clasificator, clasificatorul cu două niveluri. Rezultatele prezentate în acest capitol sunt îmbunătăţite faţă de cele publicate, deoarece am îmbunătăţit extragerea coeficienţilor cepstrali, derivatele de ordin 1 şi 2.

Pe lângă baza de date TIMIT, am testat performanţele sitemului şi pe baza de date OASIS Numbers. Am comparat şi pe această bază de date rezultatele noastre cu cele obţinute de către alte grupuri de cercetători. Rezultatele obţinute de autor cu modele GMM sunt la fel de bune ca cele obţinute cu modele HMM. Tabelul 21 prezintă rezultatele comparative. Cele două lucrări, cu care am comparat rezultatele noastre, au utilizat un set de 29 de foneme, grupând alofoanele. Noi am făcut experimentele fără această grupare, utilizând toate cele 31 de foneme.

Articol	Vector acustic	Model	#mixt.	Clasif.
[77]	**MFCC39**	HMM	24	90.48%
[48]	**MFCC39+LDA**	GMM	3	86.76%
Această lucrare	**MFCC39**	**GMM**	**16**	**92.74%**

Tabelul 21: Rezultate comparative pentru clasificarea fonemelor OASIS 31

Capitolul 5

Recunoaşterea fonemelor cu modele GMM

5.1 Introducere

Analiza acustică a fost identică cu cea de la clasificarea fonemelor (vezi 4.2). Ca modele am utilizat modele Markov ascunse cu observaţii continue descrise în secţiunea 2.5.4.1. Fie $O = \{o_1, o_2 \ldots, o_T\}$ secvenţa de observaţii, pentru care trebuie determinată secvenţa de foneme, pe care aceasta o reprezintă. Dacă notăm cu F mulţimea tuturor secvenţelor de foneme posibile, problema recunoaşterii se poate formula, ca determinarea acelei secvenţe de foneme $\hat{f} \in F$, care verifică următoarea egalitate

$$\hat{f} = \arg\max_{f \in F} P(f|O) = \arg\max_{f \in F} \frac{P(O|f) \cdot P(f)}{P(O)} \tag{5.1}$$

unde $P(f)$ este modelul limbaj (la nivel de foneme). Presupunând că toate secvenţele de observaţii sunt echiprobabile, ecuaţia 5.1 devine

$$\hat{f} = \arg\max_{f \in F} P(O|f) \cdot P(f) \tag{5.2}$$

Deoarece fiecare fonem a fost modelat cu un model Markov cu o singură stare, problema găsirii celei mai probabile secvenţe de stări este identică cu găsirea celei mai probabile secvenţe de foneme, problemă care este rezolvată de algoritmul Viterbi şi care se bazează pe calcularea probabilităţilor înainte pentru fiecare secvenţă de timp conform ecuaţiei

$$\alpha_t(j) = \max_{1 \leq i \leq n} \alpha_{t-1}(i) \cdot a_{ij} \cdot b_j(o_t). \tag{5.3}$$

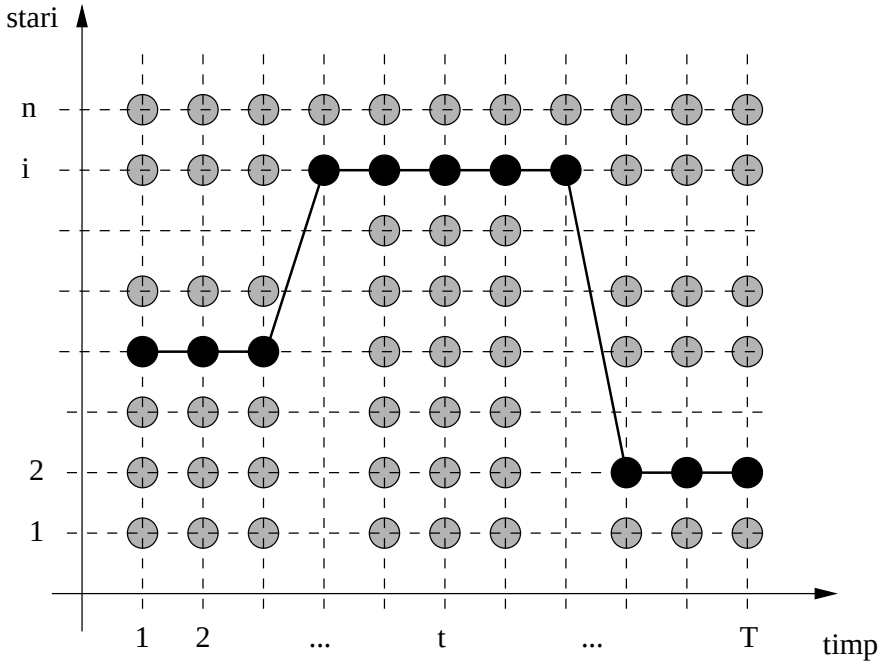


Figura 5.1: Algoritmul Viterbi - Căutare în spaţiul stărilor

Algoritmul Viterbi realizează o căutare în spaţiul stărilor (fonemelor), determinând secvenţa cea mai probabilă de stări (foneme) pentru un şir de vectori acustici. Practic realizează segmentarea şirului de vectori acustici în stări (foneme). Rezultatul căutării este reprezentat în figura 5.1.

În această lucrare a fost utilizată forma logaritmată a formulei 5.3, adică

$$\log \alpha_t(j) = \max_{1 \leq i \leq n} \{\log \alpha_{t-1}(i) + \log a_{ij}\} + \log b_j(o_t) \tag{5.4}$$

Deoarece am utilizat modele HMM cu o singură stare, nu există probabilitate de trecere de la o stare la alta (lipseşte a_{ij}). Aplicând algoritmul Viterbi clasic am obţinut rezultate foarte slabe, numărul inserărilor fiind foarte mare. De aceea, am introdus în formula 5.4 în loc de a_{ij} o penalizare de inserare notată cu

$$I_{ij} = \begin{cases} \beta, \ daca \ i = j \\ 0, \ caz \ contrar \end{cases} \tag{5.5}$$

cu care am reuşit să controlăm numărul inserărilor. De fapt această valoare empirică a fost utilizată şi pentru balansarea erorilor de ştergere şi inserare. Astfel

formula 5.4 se modifică în

$$\log \alpha_t(j) = \max_{1 \leq i \leq n} \{\log \alpha_{t-1}(i) + I_{ij}\} + \log b_j(o_t) \tag{5.6}$$

Constanta β a fost determinată în mod empiric. În secţiunile următoare prezentăm determinarea acesteia pentru cele două baze de date TIMIT şi OASIS Numbers.

5.2 Rezultate experimentale

Pentru evaluarea rezultatelor am utilizat metrica standard, rata de eroare la nivel de foneme, notat cu PER. Această măsură calculează diferenţa dintre secvenţa de foneme de intrare şi secvenţa de foneme recunoscută. Distanţa se calculează cu algoritmul prezentat în secţiunea 2.6.3, doar că în acest caz se notează cu PER în loc de WER, fiind vorba de recunoaşterea fonemelor şi nu a cuvintelor.

$$PER = 100 \cdot \frac{INS + SUBST + DEL}{N} \tag{5.7}$$

unde N reprezintă numărul de foneme în propoziţia de recunoscut. Acurateţea sistemului se va calcula cu formula

$$ACCURACY = 100 - PER \tag{5.8}$$

O altă măsură utilizată este numărul corect de foneme returnate de către sistemul de recunoaştere. Această valoare se calculează tot cu algoritmul 10. Noi o vom nota în continuare cu $CORRECT$.

5.2.1 TIMIT

Măsurătorile în acest caz au fost făcute pentru două seturi de date de test, numite şi **timit_test_core** respectiv **timit_test**. Setul de test **timit_test_core** conţine în total 192 de propoziţii, iar **timit_test** conţine câte 10 propoziţii de la 168 de vorbitori, deci în total 1680 de propoziţii. Rezultatele pentru setul **timit_test** sunt aproximativ cu 4% mai bune decât cele pentru setul **timit_test_core**. Tabelele 22 şi 23 prezintă rezultatele obţinute pentru recunoaşterea celor 61 de foneme din baza de date TIMIT obţinute pe seturile de test **timit_test_core** respectiv **timit_test**. În ambele cazuri am folosit ca şi modele de foneme mixturi gaussiene cu câte 32 de componente. Setul **timit_test_core** conţine în total 7525 de foneme, iar setul **timit_test** în total 65825 de foneme.

Menţionăm că rezultatele au fost obţinute fără a utiliza model de limbaj la nivel de foneme. Acest lucru înseamnă că nu a fost făcută nici un fel de constrângere, orice fonem poate să urmeze după oricare altul. Conţinutul tabelului 23 este reprezentat grafic în figura 5.2.

β	DEL	INS	SUBST	ACCURACY	CORRECT
10	6.18%	9.49%	38.77%	45.55%	55.04%
11	7.41%	7.68%	37.76%	47.14%	54.81%
13	9.75%	4.81%	36.37%	49.06%	53.87%
20	**18%**	**1.26%**	**30.88%**	**49.86%**	**51.12%**

Tabelul 22: Recunoaşterea fonemelor - **timit_test_core**

β	DEL	INS	SUBST	ACCURACY	CORRECT
11	6.79%	7.39%	34.54%	51.27%	58.66%
13	9.13%	4.76%	33.12%	52.98%	57.74%
15	**11.42%**	**3.24%**	**31.66%**	**53.67%**	**56.91%**
20	17.16%	1.31%	28.12%	53.40%	54.71%

Tabelul 23: Recunoaşterea fonemelor - **timit_test**

5.2.2 OASIS Numbers

În această secţiune prezentăm rezultatele sistemului pentru recunoaşterea fonemelor din baza de date OASIS Numbers. Fonemele au fost modelate cu mixturi gaussiene cu 16 componente şi setul de test conţine în total 2316 foneme. Tabelul 24 prezintă rezultatele obţinute pentru diferite valori ale parametrului β. Figura 5.3 prezintă sub formă grafică rezultatele din tabelul 24.

β	DEL	INS	SUBST	ACCURACY	CORRECT
11	0.52%	27.93%	5.18%	66.36%	94.30%
13	0.82%	21.80%	5.53%	71.85%	93.65%
20	2.37%	11.27%	6.00%	80.35%	91.62%
25	**4.06%**	**7.77%**	**5.78%**	**82.38%**	**90.15%**
27	4.83%	7.08%	5.83%	82.25%	89.33%
30	6.43%	5.82%	5.70%	82.03%	87.86%

Tabelul 24: Recunoaşterea fonemelor - **oasis_test**

Rezultatele noastre se compară în mod favorabil cu cele publicate în [80] pe aceeaşi bază de date. Fără utilizarea cunoştinţelor rezultate din vocabular, adică

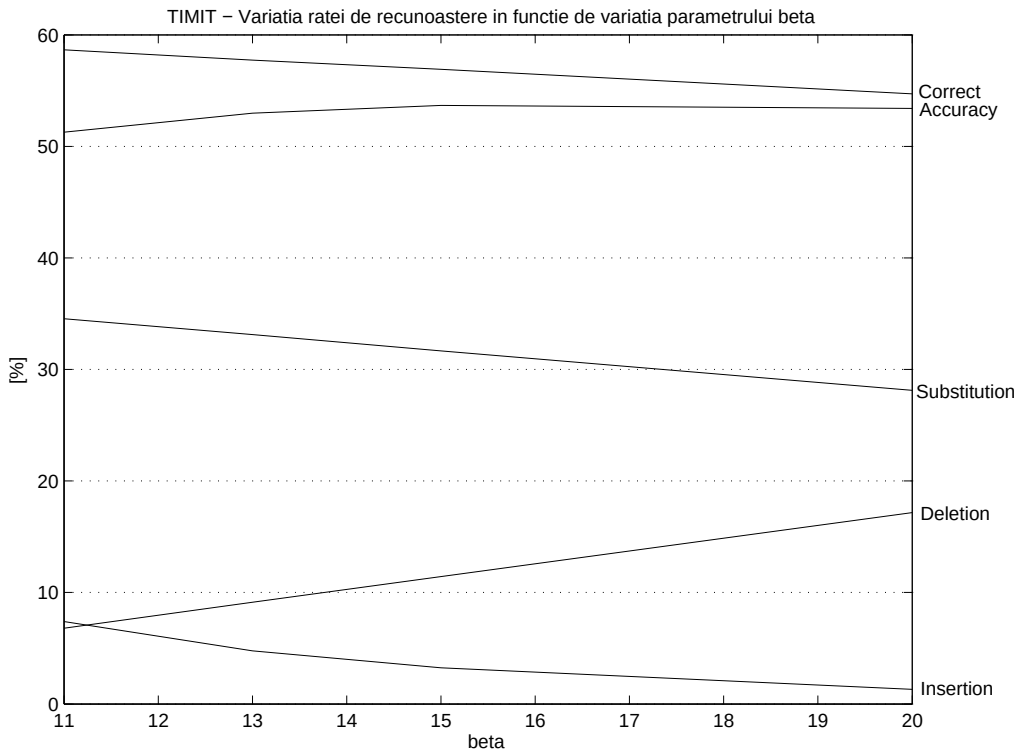


Figura 5.2: TIMIT-Variaţia ratei de recunoaştere în funcţie de variaţia parametrului β

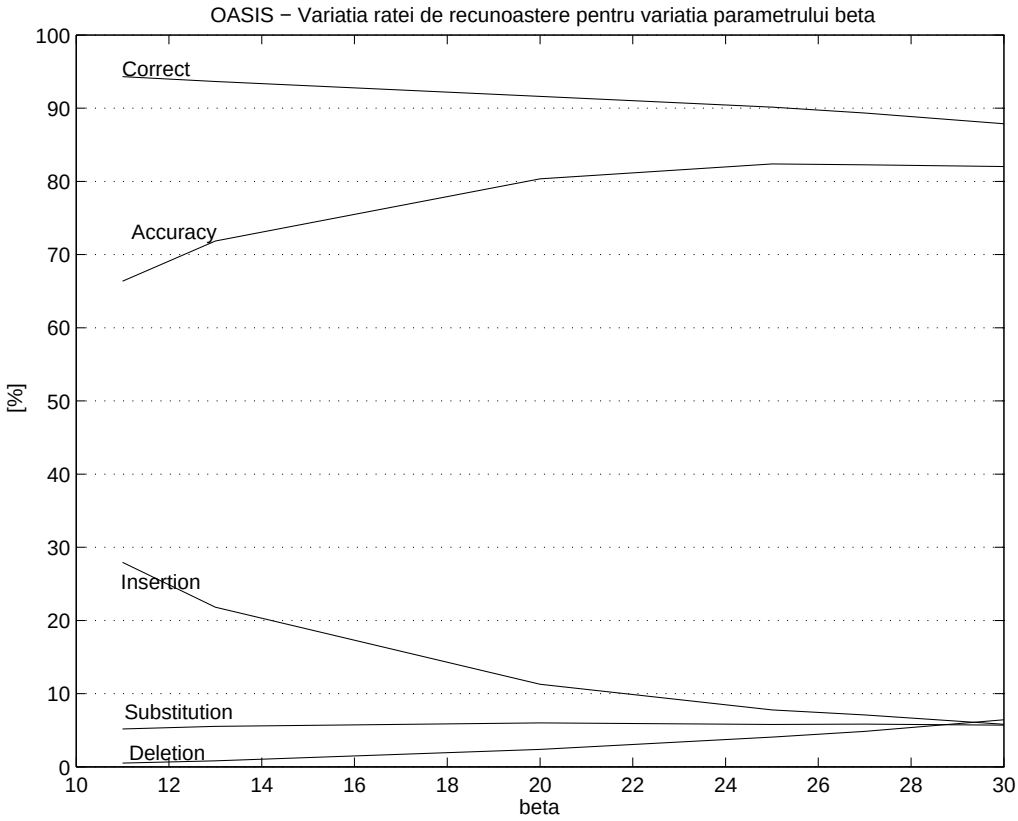


Figura 5.3: OASIS-Variaţia ratei de recunoaştere în funcţie de variaţia parametrului β

orice fonem poate să urmeze după oricare altul, creatorii acestei baze de date au obţinut rata de recunoaştere maximă de 82.05%. Noi am obţinut rata maximă de 82.38%, iar cu balansarea erorilor de inserare respectiv de ştergere, 82.03% (ultimul rând din tabelul 24). Utilizând un model bigram la nivel de foneme această rată de recunoaştere a fost ridicată la 96.87%.

5.3 Concluzii

În acest capitol am prezentat sistemul realizat de autor pentru recunoaşterea fonemelor. De fapt acest capitol se bazează pe cel precedent, clasificarea fonemelor, deoarece modelele de foneme utilizate au fost evaluate în capitolul precedent. Pentru a putea detecta capetele fonemelor în vorbirea continuă este nevoie de un algoritm eficient de căutare. În literatura de specialitate sunt recomandaţi algoritmii: Viterbi clasic, Viterbi fascicular, respectiv algoritmul cu stivă. Noi am aplicat algoritmul Viterbi clasic. Din cauză că, acesta producea un număr mare de erori de inserare pentru modelele de foneme independente de context, am introdus o modificare în acest algoritm. Această modificare se referă la introducerea unei constante care controlează trcerea de la o stare la alta. Cu cât este mai mare valoarea acestei constante, cu atât este mai defavorizată trecerea la o stare (fonem) nouă. Cu această constantă se poate regla şi raportul dintre numărul de erori de tip inserare, respectiv ştergere. Cu cât este mai mare valoarea acestei constante, cu atât este mai mare numărul erorilor de tip ştergere, deci este direct proporţională cu acest tip de eroare. Cu cât este mai mare valoarea acestei constante, cu atât este mai mic numărul erorilor de tip inserare, deci este invers proporţională cu acesta. Am prezentat o serie de experimente cu diferite valori ale acestei constante atât pentru baza de date TIMIT, cât şi pentru OASIS Numbers. Aşa cum arată experimentele, valoarea acestui parametru depinde de baza de date, adică de durata medie a fonemelor din limba respectivă.

Rezultatele au fost comparate doar în cazul bazei de date OASIS, deoarece pentru TIMIT nu există rezultate publicate pentru setul de 61 de foneme utilizând modele statistice. Toate rezultatele privind recunoaşterea fonemelor pentru baza de date TIMIT, utilizând modele statistice, sunt publicate pentu setul restrâns de 39 de foneme. Doar articolele care prezintă experimentele realizate cu reţele neuronale prezintă rezultate pentru toate cele 61 de foneme. Însă aceste modele nu sunt comparabile cu modelele de foneme GMM independente de context, eventual cu cele dependente de context, deoarece reţelele neuronale descrise au capacitatea de a învăţa contextul. Primul articol care publică rezultate pentru

setul **timit_test** cu 39 de foneme este [51], obţinând 58.77% cu modele independente de context şi fără model de limbaj. În articolul [78] rezultatele sunt prezentate tot pentru setul de 39 de foneme cu modele hibride (HMM/ANN) obţinând 58.40% rată de recunoaştere cu vectori acustici similari. Rezulatul nostru cel mai bun pentru setul de 61 de foneme 53.67% se compară în mod favorabil cu aceste rezulate.

Pentru baza de date OASIS, rezultatele noastre cele mai bune (82.38%) sunt aproape identice cu cele obţinute de autorii bazei de date (82.05%) cu acelaşi tip de vectori acustici şi acelaşi tip de modele GMM. Singura diferenţă în ambele cazuri este algoritmul de decodare, al nostru fiind o variantă Viterbi foarte eficient.

Capitolul 6

Modelarea duratei fonemelor

6.1 Introducere

Unul dintre punctele slabe ale modelelor Markov ascunse este că, nu oferă o reprezentare potrivită pentru structura temporală a vorbirii. Acest lucru se datorează faptului că, probabilitatea de a rămâne în aceeaşi stare descreşte exponenţial cu timpul, adică aceste modele incorporează în mod implicit un model de durată exponenţial. Probabilitatea de a rămâne în starea i exact de t ori se poate exprima cu formula

$$d_i(t) = a_{ii}^t(1 - a_{ii}) \tag{6.1}$$

în cazul modelelor Markov clasice, care permit două tipuri de tranziţii dintr-o stare. a_{ii} este probabilitatea de a rămâne în starea i, iar $(1 - a_{ii})$ este probabilitatea de a trece în starea următoare. Figura 6.1 ne arată structura unui astfel de model Markov cu trei stări.

Durata poate fi inclusă în modelele Markov. O primă încercare în acest sens a fost făcută în [54], care modifică algoritmul Viterbi pentru a include şi duratele. Fie $O = \{o_1,\ o_2, \ldots, o_T\}$ secvenţa de observaţii care trebuie decodată într-o secvenţă de foneme şi cu $\alpha_t(j)$ notăm probabilitatea maximă de a fi în starea j şi observând secvenţa de observaţii $o_1,\ o_2, \ldots o_t$. Atunci valorile $\alpha_t(j)$ se pot determina cu ajutorul formulei următoare

$$\alpha_t(j) = \max_{1 \le i \le n} \left\{ \max_{1 \le \tau \le D} \left\{ \alpha_{t-\tau}(i) a_{ij} d_j(\tau) \prod_{\Theta=0}^{\tau-1} b_j(o_{t-\Theta}) \right\} \right\} \tag{6.2}$$

unde D este durata maximă admisă pentru o fonemă. Acest algoritm creşte complexitatea algoritmului Viterbi cu un factor D, care poate fi inadmisibil pentru

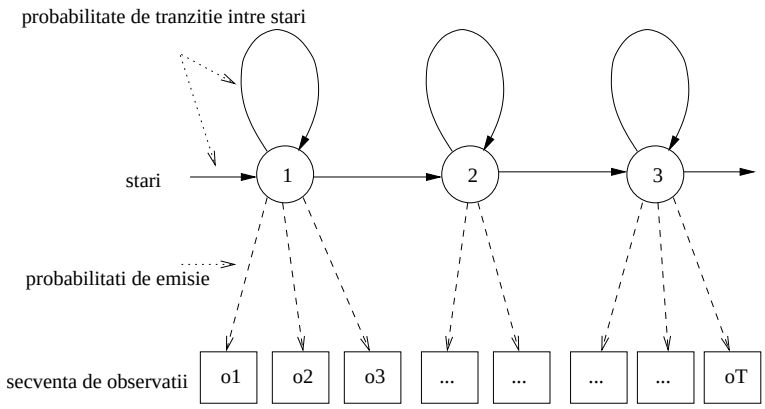


Figura 6.1: Model Markov ascuns cu trei stări

recunoaşterea vorbirii continue, însă acceptabil pentru recunoaşterea cuvintelor izolate.

Iniţial au fost utilizate distribuţii neparametrice. Acest lucru înseamnă că, fiecare $d_i(\tau)$ a fost estimat pentru fiecare $1 \leq \tau \leq D$. Acest tip de estimare însă necesită o cantitate foarte mare de date de antrenare. Din această cauză au fost propuse câteva distribuţii parametrice împreună cu formulele de estimare ale acestora. Distribuţiile propuse sunt: Poisson, Gamma şi cea gaussiană.

6.2 Rezultate experimentale

6.2.1 OASIS Numbers

În această secţiune prezentăm rezultatele sistemului pentru recunoaşterea fonemelor din baza de date OASIS Numbers utilizând modelarea duratei. Ca modele acustice pentru foneme au fost folosite modele Markov ascunse cu o singură stare.

Figura 6.2 prezintă duratele minime, respectiv maxime a celor 31 de foneme OASIS. Durata medie este marcată cu steluţă. Duratele minime, respectiv cele maxime vor fi folosite în ecuaţia 6.4.

O primă înercare este să verificăm comportarea sistemului fără a utiliza vre-un model pentru durata fonemelor, însă impunând nişte durate minime şi maxime pentru acestea. Pentru acesta formula 6.2 a fost simplificată rezultând în

$$\alpha_t(j) = \max_{1 \leq i \leq n} \left\{ \max_{\tau_{min} \leq \tau \leq \tau_{max}} \left\{ \alpha_{t-\tau}(i) \cdot a_{ij} \cdot \prod_{\Theta=0}^{\tau-1} b_j(o_{t-\Theta}) \right\} \right\} \tag{6.3}$$

pentru $1 \leq j \leq n$ şi $1 \leq t \leq T$, unde τ_{min} şi τ_{max} sunt duratele permise

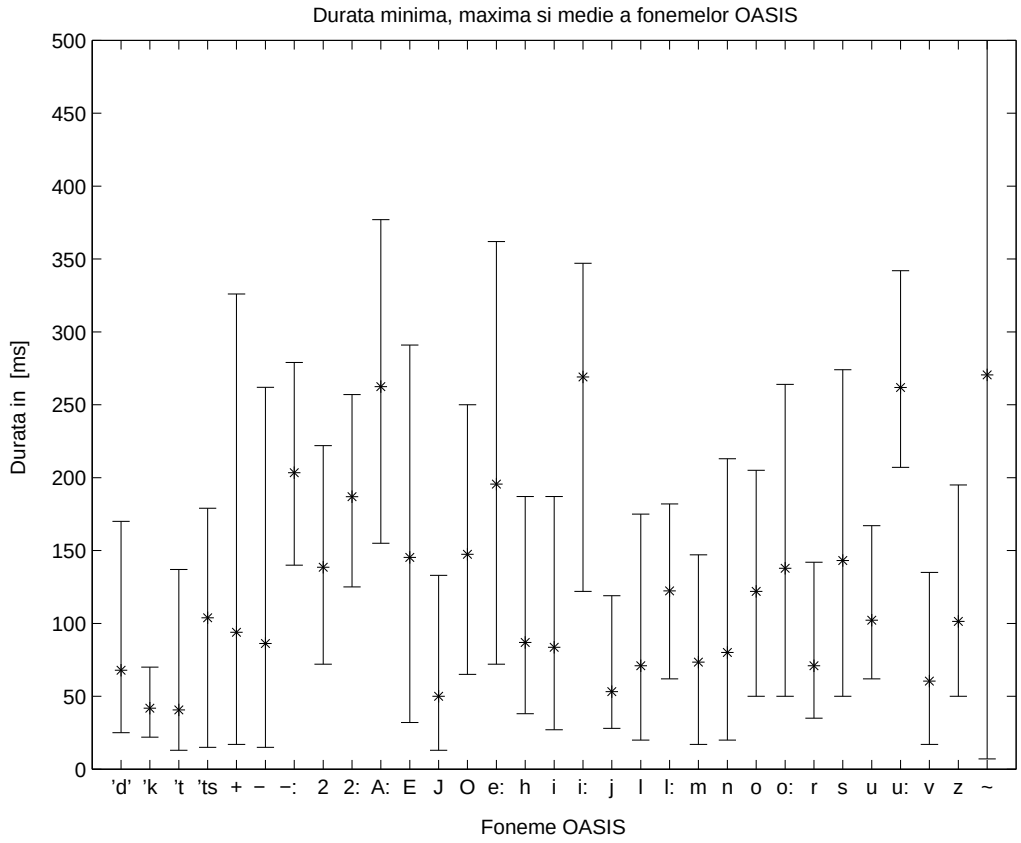


Figura 6.2: Durata minimă, maximă şi medie a fonemelor OASIS

pentru unităţile fonetice. Deoarece fonemele sunt modelate cu o singură stare, semnificaţia a_{ij} se modifică, acesta reprezentând probabilitatea de trecere de la fonemul f_i la fonemul f_j. Noi am utilizat doar valori 0 şi 1, 0 reprezentând faptul că f_i nu poate fi urmat de f_j, iar valoarea de 1 contrariul. Elementele matricii a_{ij} se determină pe baza vocabularului şi reprezintă un model bigram al limbajului, la nivel fonetic.

Pentru a reduce cantitatea de calcule am calculat, tot be baza înregistrărilor segmentate la nivel fonetic, durata maximă, respectiv durata minimă pentru fiecare fonem în parte (vezi figura 6.2). Aceste valori se notează cu $\tau_{min}(j)$ respectiv $\tau_{max}(j)$ pentru $j = 1 \ldots n$. Utilizând aceste valori, formula 6.3 se modifică în

$$\alpha_t(j) = \max_{1 \leq i \leq n} \left\{ \max_{\tau_{min}(j) \leq \tau \leq \tau_{max}(j)} \left\{ \alpha_{t-\tau}(i) \cdot a_{ij} \cdot \prod_{\Theta=0}^{\tau-1} b_j(o_{t-\Theta}) \right\} \right\} \tag{6.4}$$

Tabelul 25 prezintă rezultatele obţinute pentru recunoaştere indicând duratele minime şi maxime utilizate. Aceste rezultate au fost publicate în [10]. Toate măsurătorile au fost făcute pe două seturi de modele, unul folosind 8 componente gaussiene, iar al doilea folosind 16 componente gaussiene. Duratele $\tau_{min} \ldots \tau_{max}$ reprezintă faptul că, în acest caz am folosit duratele specifice pentru fiecare fonem în parte.

Nr. mixt.	Durate	CORR.	ACC.	INS	SUBST	DEL
8	1..50	97.23%	11.70%	85.53%	2.63%	0.12%
8	2..50	96.93%	60.70%	36.22%	2.84%	0.21%
8	3..50	94.68%	76.29%	18.39%	4.23%	1.07%
8	$\tau_{min} \ldots \tau_{max}$	92.96%	76.64%	16.32%	5.95%	1.07%
16	1..50	97.40%	24.30%	73.10%	2.54%	0.04%
16	2..50	96.37%	68.65%	27.72%	3.45%	0.17%
16	3.50	96.64%	77.54%	17.09%	4.66%	0.69%
16	$\tau_{min} \ldots \tau_{max}$	92.65%	76.94%	15.71%	6.21%	1.12%

Tabelul 25: Recunoaşterea fonemelor - OASIS - 31

6.2.2 TIMIT

Figura 6.3 prezintă duratele minime respectiv maxime a celor 61 de foneme TIMIT. Durata medie este marcată cu steluţă.

Distribuţia duratelor pentru fiecare fonem în parte sunt reprezentate cu câte o histogramă (vezi anexa 1).

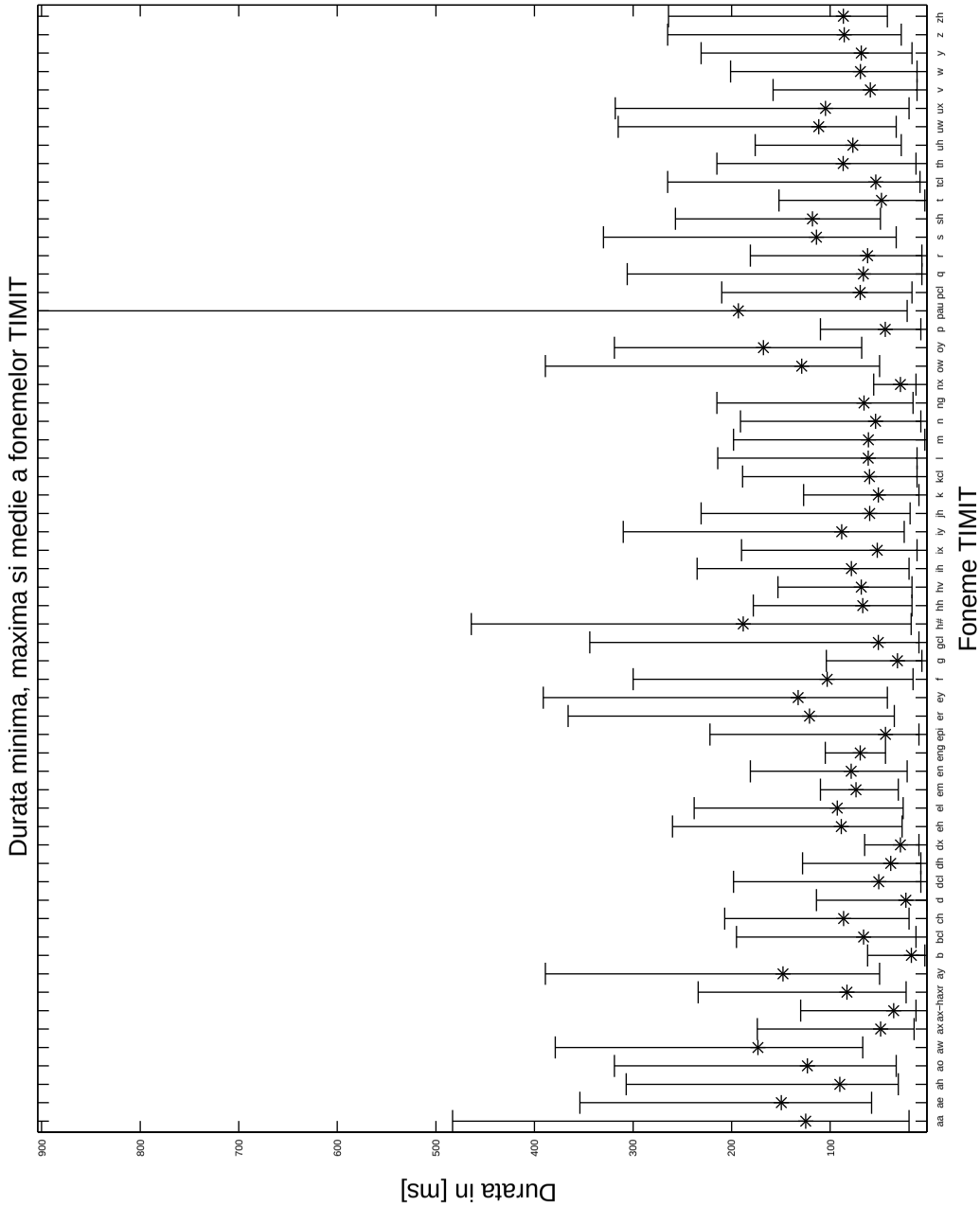


Figura 6.3: Durata minimă, maximă şi medie a fonemelor TIMIT

6.3 Concluzii

Pe baza duratelor specifice ale fonemelor am stabilit o relaţie între parametrul empiric şi durata medie de fonem. Tabelul 26 prezintă duratele medii, respectiv valorile optime ale parametrului β. Pentru parametrul empiric am considerat acea valoare, pentru care acurateţea de recunoaştere era maximă (vezi tabelele 23 şi 24). Relaţia cu care se poate calcula acest parametru empiric ar fi

$$\beta = \frac{durata\_medie\_fonem}{5} \tag{6.5}$$

	Durată medie	β
TIMIT	80 ms	15
OASIS	125 ms	25

Tabelul 26: Durata medie de fonem şi valoare β

O concluzie referitoare la duratele experimentate pentru baza de date OASIS ar fi că, impunerea unei durate minime este foarte importantă, acesta poate să furnizeze rezultate la fel de bune ca şi includerea în formulă a duratelor specifice fiecărei foneme (τ_{min} şi τ_{max}). De remarcat este şi faptul că, rezultatele obţinute în capitolul precedent pentru recunoaşterea fonemelor din baza de date OASIS au fost mai bune (82.38%), decât cele obţinute cu modelarea explicită a duratei (77.54%). Ambele rezultate au fost obţinute cu modele de foneme formate din 16 componente gaussiene. O parte din aceste rezultate a fost publicată în [10].

Capitolul 7

Recunoaşterea cuvintelor prin modelare fonetică

7.1 Introducere

În acest capitol prezentăm rezultatele experimentale obţinute pentru recunoaşterea cuvintelor pronunţate în mod izolat. Au fost făcute două rânduri de experimente, în primul modelând cuvinte întregi şi în al doilea modelând fonemele.

În cel de-al doilea caz cuvintele au fost reprezentate cu ajutorul unui vocabular care conţine pronunţiile fonetice ale cuvintelor. Există diferite abordări ale creării unui asemenea vocabular şi anume [72]:

- fiecare cuvânt are o singură reprezentare fonetică
- fiecare cuvânt poate avea mai multe reprezentări fonetice
- fiecare cuvânt poate avea mai multe reprezentări fonetice, fiecărei reprezentări îi corespunde câte o probabilitate

Pentru exemplificare considerăm cuvântul din limba engleză *bottle*. Acest cuvânt admite mai multe pronunţii corecte cum ar fi: `[ b aa dx el ]`, `[ b aa t el ]`, `[ b aa dx ax l ]`, `[ b aa t ax l ]`. Introducând multe variante de pronunţie creşte dimensiunea vocabularului, însă ridică acurateţea sistemului. Probabilităţile ataşate pronunţiilor se determină în funcţie de frecvenţa acestor pronunţii.

O altă problemă, care se ridică, se referă la reprezentarea eficientă a vocabularului. Există două abordări:

- liniar

- organizat sub formă de arbore (trie)

Fiecare metodă are avantaje şi dezavantaje. Avantajul modelului liniar ar fi aplicarea simplă a modelului de limbaj, iar dezavantajul este că, ocupă multă memorie. Modelul organizat sub formă de arbore reprezintă o singură dată prefixurile comune ale cuvintelor, deci este mult mai avantajos din punctul de vedere al memoriei, însă apar dificultăţi în cazul aplicării modelului de limbaj.

7.2 Modele de cuvinte

Modelarea cuvintelor poate fi făcută în mai multe moduri, chiar dacă ne referim doar la metodele statistice. Sistemele de recunoaşterea cuvintelor izolate, care conţin un număr redus de cuvinte, cum ar fi cifrele, pot utiliza şi metoda alinierii dinamice în timp pentru recunoaşterea cuvintelor. Descrierea acestor tipuri de modele se găsesc în [26, 67, 79]. Modelele Markov ascunse cu observaţii discrete furnizează rezultate foarte bune pentru această problemă [68]. O serie de rezultate au fost publicate şi pentru limba română pentru modelarea cuvintelor folosind atât modelele statistice, cât şi diferite tipuri de reţele neuronale [20, 31, 79].

Noi am utilizat modele Markov ascunse cu o singură stare (GMM) pentru a demonstra că, şi aceste modele simple, care reprezintă doar conţinutul acustic al cuvintelor, fără a modela distribuţia acestora în timp, pot discrimina un număr de zece cuvinte. Antrenarea cuvintelor (cifrelor) s-a făcut cu câte 40 de exemplare, 20 de vorbitori rostesc de câte două ori fiecare cifră. Testarea fiecărui cuvânt s-a făcut pe 12 exemplare, câte două de la fiecare dintre cei 6 vorbitori. Figura 7.1 prezintă procesul de evaluare, iar rezultatele sunt prezentate în tabelul 27. Pentru a determina numărul de mixturi potrivite am început cu o singură mixtură pentru fiecare cuvânt, dublând numărul acestora la fiecare experiment. Cu 16 mixturi am obţinut o rată de recunoaştere 100%.

Număr mixturi	Rata de recunoaştere
1	83%
2	82%
4	94%
8	98%
16	100%

Tabelul 27: Recunoaşterea cuvintelor cu modele GMM - OASIS-10 cifre

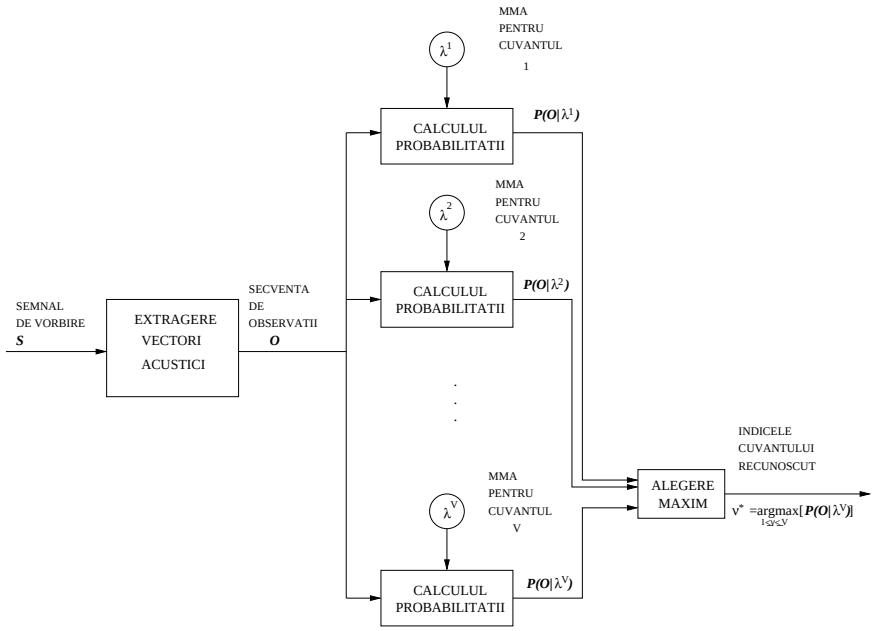


Figura 7.1: Recunoaşterea cuvintelor izolate (după [67])

7.3 Modele de foneme

În vocabular am inclus mai multe pronunţii pentru acelaşi cuvânt, dar numărul acestor pronunţii a fost limitat la două. Pentru reprezentarea vocabularului am ales cea de-a doua variantă, adică reprezentarea sub formă de arbore. Acest tip de arbore se numeşte şi *trie*, denumirea provenind de la information re*trie*val. Acest arbore permite organizarea eficientă a vocabularelor, mai mult, este structura de date, care permite o căutare eficientă de timp constant $\Theta(1)$, timpul de căutare al unui cuvânt depinzând doar de lungimea acestuia, nu şi de dimensiunea vocabularului. Acest tip de căutare este numit şi căutare digitală [50]. Pentru decodare am utilizat algoritmul Viterbi cu parametrul empiric, introdus în secţiunea 5.1. Figura 7.2 prezintă o parte din vocabular organizat sub formă de trie.

Recunoaşterea de cuvinte a fost făcută conform figurii 7.3. În primul pas am obţinut vectorii acustici din semnalul vocal (vezi secţiunea 4.2), după care am aplicat o variantă a algoritmului Viterbi. Acest algoritm ne-a furnizat o secvenţă de foneme, care eventual a fost postprocesată aplicând nişte corecţii pe baza regulilor fonetice şi apoi căutată în vocabularul reprezentat sub formă de arbore. Căutarea se începe întotdeauna de la rădăcina arborelui şi se caută un drum de la rădăcină la un nod, care reprezintă sfârşitul unui cuvânt. Dacă secvenţa de foneme nu a fost găsită, atunci acesta nu va fi identificată cu nici un cuvânt.

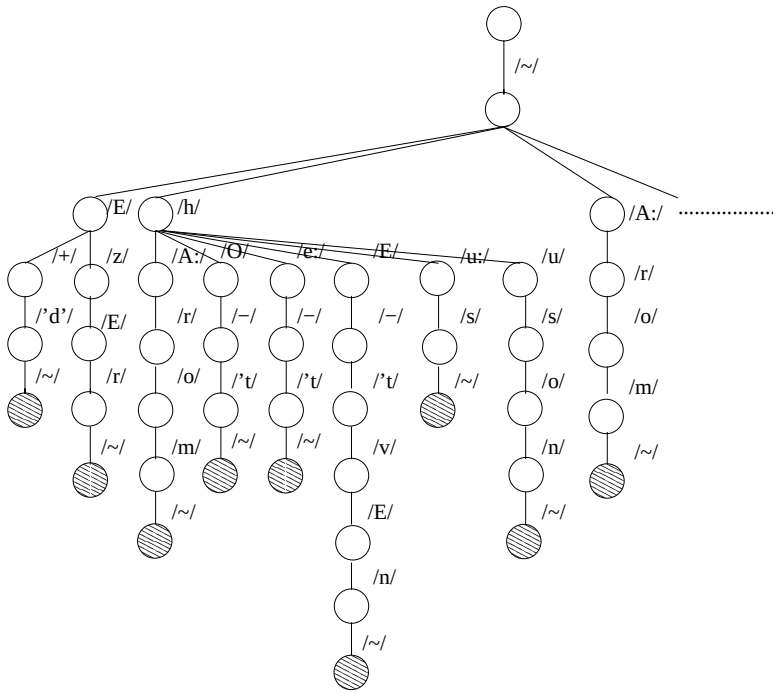


Figura 7.2: OASIS-Vocabular organizat sub formă de trie

În capitolul Recunoaşterea fonemelor am prezentat diferitele tipuri de erori de recunoaştere. Am văzut că, erorile de inserare sunt cele mai ridicate. În cazul acestor erori cea mai frecventă este inserarea diferitelor foneme, care reprezintă liniştea la începutul, dar mai ales la sfârşitul rostirilor. În faza de postprocesare deocamdată am comasat acele foneme, care reprezintă linştea. Exemplu:

Secvenţă de foneme la intrare: `~ E z E r ~`

Secvenţă de foneme decodată : `~ E z E r -: ~`

După comasare : `~ E z E r ~`

Tabelul 28 prezintă eroarea la nivel de cuvinte obţinută pentru setul de test, care conţine 312 cuvinte rostite de către 6 vorbitori diferiţi. Tabelul prezintă rezultatele cu şi fără comasarea fonemelor de tip linişte. Vocabularul conţine 34 de cuvinte, care reprezintă de fapt 26 de cuvinte, câteva cuvinte admit mai multe pronunţii fonetice.

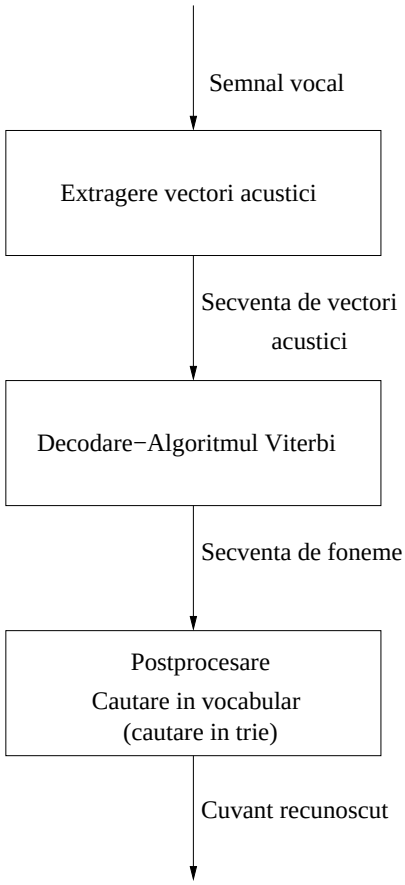


Figura 7.3: Recunoaşterea cuvintelor cu modele de foneme

Tip recunoaştere	DEL	INS	SUBST	REC. FO-NEME	CORR.	REC. CU-VINTE
fără postprocesare	2.37%	11.27%	6.00%	80.35%	91.62%	84.62%
cu postprocesare	2.46%	7.64%	5.52%	84.36%	92.01%	86.54%

Tabelul 28: Recunoaştere cuvinte - OASIS 26 de cuvinte - $\beta = 20$

7.4 Concluzii

În acest capitol am prezentat rezultate experimentale pentru recunoaşterea cuvintelor rostite izolat. Sunt prezentate două rânduri de experimente, primul introducând modelele GMM pentru modelarea cifrelor. Pentru un vocabular conţinând doar 10 cuvinte modelele GMM au fost adecvate. Folosind 16 componente gaussiene, am obţinut rata de recunoaştere 100%. Pentru un vocabular cu mai multe cuvinte şi cuvinte mai similare, acest model nu ar fi potrivit.

Partea a doua a capitolului utilizează cunoştinţele dobândite la recunoaşterea fonemelor şi prezintă sistemul de recunoaşterea cuvintelor izolate bazată pe foneme. Noutatea faţă de recunoaşterea fonemelor este, că este necesară introducerea modelării vocabularului. Noi am optat pentru modelul mai eficient din punct de vedere al spaţiului de memorie, şi am utilizat un vocabular sub formă de arbore. Rata de recunoaştere a cuvintelor, fără introducerea limbajului de model este de 84.62%. Cu introducerea cunoştinţelor fonetice acest rezulatat poate fi îmbunătăţit. Noi am aplicat o simplă postprocesare pentru comasarea erorilor de tip inserare, ceea ce a crescut deja cu 2% rata de recunoaştere, reducând-se mai ales erorile de tip inserare.

Capitolul 8

Concluzii finale

În această lucrare am prezentat structura generală a sistemelor de recunoaşterea vorbitorului şi recunoaşterea vorbirii continue. În acest scop am realizat o aplicaţie cu următoarele module, care se poate utiliza atât pentru recunoaşterea vorbitorului cât şi la recunoaşterea vorbirii:

- modul pentru extragerea vectorilor acustici (MFCC, LPCC)
- modul pentru tehnici de grupare conţinând tehnicile: k-means, cuantizare vectorială binară ierarhică, LVQ, GMM
- modul pentru modele Markov ascunse discrete (HMM discret)
- modul pentru reprezentarea lexiconului
- modul pentru tehnici de decodare

Menţionăm că, toate componentele acestui sistem sunt independente de limba, care trebuie recunoscută. Experimentele au fost făcute pentru două baze de date, una pentru limba engleză TIMIT şi una pentru limba maghiară OASIS Numbers. Aceste baze de date au fost puse la dispoziţia noastră de către *Research Group on Artificial Intelligence, University of Szeged.*

8.1 Contribuţii la recunoaşterea vorbitorului

În cazul recunoaşterii vorbitorului a fost studiată recunoaşterea independentă de text. Am studiat performanţele modelelor de vorbitori create cu ajutorul diferitelor tehnici de grupare. Tipul de model ales depinde şi de numărul de vorbitori, precum şi de cantitatea de material audio disponibil. Modelele GMM au mai

mulţi parametri, deci necesită o cantitate mai mare de date pentru estimarea robustă a parametrilor modelelor. Au fost studiate modelele de vorbitori din punct de vedere al conţinutului fonetic. Deoarece mulţi autori stabilesc dimensiunea modelului (numărul componentelor gaussiene, respectiv numărul clusterelor) în funcţie de numărul fonemelor unei limbi, am verificat dacă fonemele, respectiv categoriile fonetice sunt separate în componentele acestor modele. Pentru a elimina efectul de coarticulare, din foneme au fost luate în considerare doar mijlocul acestora, care nu sunt afectate de coarticulare. Surprinzător, modelele nu separă toate categoriile de foneme, mai ales vocalele şi semivocalele sunt împrăştiate în mai multe componente. Pornind de la acest experiment, am verificat performanţele modelelor structurate pe categorii fonetice. Concluzia este că, utilizând doar categoria fonetică şi construind modele cu 6 componente gaussiene, aceste modele vor avea performanţe similare cu modelele obişnuite. Am constituit lista vorbitorilor identificaţi în mod incorect atât la modelele obişnuite GMM cât şi la modelele structurate pe categorii fonetice. Deoarece cele două sisteme produc liste diferite, aceste două tipuri de modelări ar putea fi combinate în scopul realizării unui sistem mai performant. Acest aspect rămâne o direcţie de cercetare.

Un alt aspect studiat în această lucrare a fost puterea de discriminare a diferitelor categorii fonetice. Am reuşit să ordonăm grupele fonetice pe baza puterii lor de discriminare, nazalele şi vocalele demonstrându-se foarte bune în acest scop.

8.2 Contribuţii la recunoaşterea vorbirii

Deşi există mai multe abordări ale acestei probleme, componentele, care trebuie realizate în cazul acestor sisteme reprezintă deja un standard. Prima dată trebuie selectată unitatea acustică, care va fi modelată şi apoi se antrenează modelele acustice. Problema următoare, care trebuie rezolvată, este modelarea limbajului şi reprezentarea vocabularului limbii. După ce aceste două probleme sunt rezolvate, se poate implementa o tehnică de decodare, care este de fapt o problemă de căutare cu o serie de constrângeri.

Am realizat o serie de experimente pe cele două baze de date disponibile, clasificând foneme, grupe de foneme şi categorii de foneme. Am reuşit să obţinem cu modele GMM performanţe similare cu cele ale modelelelor Markov ascunse cu trei stări, astfel demonstrând efectul neglijabil al probabilităţilor de tranziţie între stările unui model de fonem.Pentru cele 61 de foneme TIMIT am obţinut o rată de clasificare de 63.34%, pentru cel 39 de grupe de foneme TIMIT o rată de 68.13%, ambele cu modele de foneme mixturi gaussiene cu 32 de componente. În cazul

bazei de date OASIS Numbers, rata de clasificare este de 92.74% tot cu modele GMM, în acest caz modelele având 16 de componente. Numărul componentelor este puternic influenţat de cantitatea de date de antrenare. Pentru reducerea calculelor aici s-a propus un clasificator cu două niveluri ale cărui performanţe au fost testate pe baza de date TIMIT şi care, deşi este mai eficient din punct de vedere computaţional, nu reduce în mod semnificativ rata de clasificare.

În cazul recunoaşterii fonemelor am modificat algoritmul Viterbi pentru modele de foneme cu o singură stare. Introducând o constantă, a cărei valori se determină în mod empiric, am reuşit să reducem erorile de tip inserare. Recunoaşterea a fost testată doar pentru varianta de 61 de foneme, rezultate mult mai bune s-ar fi putut obţine cu guparea alofoanelor (varianta cu 39 de grupe). În acest caz pe TIMIT am obţinut 53.67, ca rată maximă de recunoaştere. Pentru OASIS Numbers rata maximă a fost 82.38%.

Studiul duratei fonemelor este o problemă importantă, mai ales în cazul recunoaşterii limbilor, care au pentru acelaşi sunet atât variantă scurtă, cât şi lungă, durata fiind singura diferenţă între acestea. Limbile finugorice fac parte din categoria acestor limbi. Această problemă este însă una mult studiată şi pentru limba engleză. Am prezentat o analiză statistică a celor două baze de date disponibile din punctul de vedere al duratei fonemelor şi pe baza cunostinţelor dobândite am propus a variantă pentru modelarea duratei fonemelor. Modelarea duratei se face doar la nivel de decodare, însă această variantă ridică foarte mult timpul de decodare. Din această cauză rezultate experimentale sunt prezentate doar pentru OASIS Numbers, care conţine doar cuvinte. Pentru baza de date TIMIT acest tip de decodare este nepractic, ridicând insuportabil timpul necesar pentru decodare. Rezultatele experimentale ne arată că, impunerea unei durate minime este chiar mai importantă decât impunerea duratelor specifice fiecărei foneme.

În final sunt prezentate rezultate pentru recunoaşterea cuvintelor. În cazul utilizării unui număr redus de cuvinte, cum ar fi cifrele, se recomandă folosirea modelelor de cuvinte. Cu modele având 16 componete gaussiene am obţinut o rată de recunoaştere 100% în cazul unui sistem independent de vorbitor. Dacă avem de recunoscut un număr mai mare de cuvinte, atunci se recomndă crearea modelelor de cuvinte din modelele de foneme ale limbii respective. Rezultatul este de 84.36% pe 26 de cuvinte, câteva dintre ele fiind foarte similare, deosebindu-se doar printr-un fonem. În acest caz nu au fost utilzate cunoştinţe lingvistice, orice fonem poate să urmeze după oricare altul.

Majoritatea rezultatelor prezentate în această lucrare au fost publicate în reviste de specialitate şi în volume ale unor conferinţe naţionale şi internaţionale,

acestea fiind totodată citate în capitolele corespunzătoare din lucrare.

8.3 Direcţii de cercetare

Recunoaşterea vorbitorului:

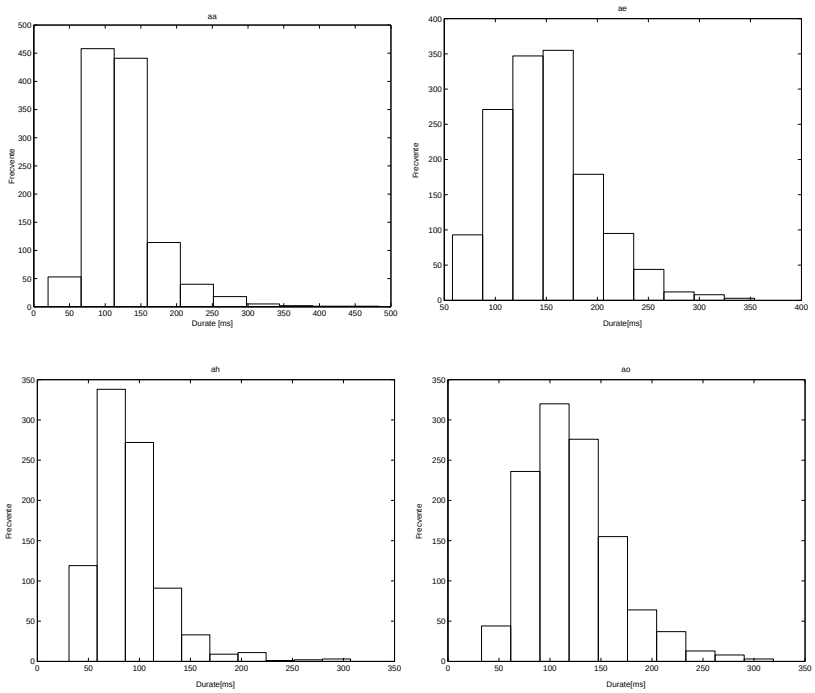
- O continuare firească ar fi construirea modelelor de vorbitori prin adaptarea modelelor generale de foneme. Pentru această investigare ar fi nevoie de o altă bază de date, deoarece în baza de date TIMIT există foneme care lipsesc sau sunt foarte slab reprezentate (cazul africatelor).
- O altă direcţie de cercetare ar fi verificarea identităţii pe baza unor informaţii lingvistice, cum ar fi durata unor foneme sau cuvinte specifice.
- În final, ar fi de dorit combinarea mai multor informaţii atât de nivel inferior cât şi de nivel superior. Aceasta iarăşi necesită crearea unei baze de date care permite aceste studii.

Recunoaşterea vorbirii:

- În cazul recunoaşterii fonemelor, deoarece modelele Markov ascunse reprezintă o modelare total neadecvată a duratelor fonemelor, ar fi necesar un studiu complet al tuturor posibilităţilor de integrare al modelării acestor durate fără a mări complexitatea algoritmului de decodare. Actualmente toate încercările de modelare a duratei fonemelor măresc complexitatea calculelor.

Anexa A

Duratele fonemelor din baza de date TIMIT



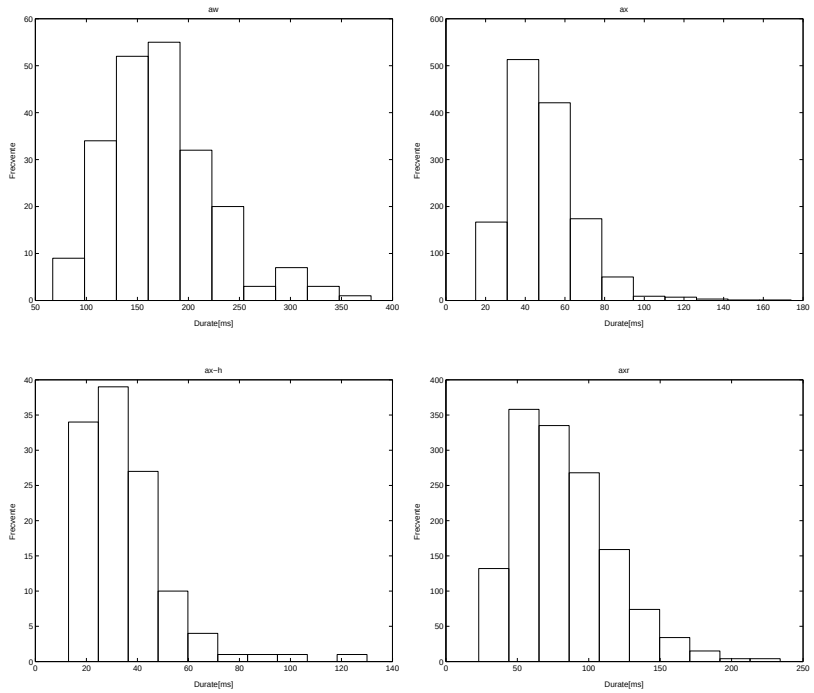


Figura A.1: Histograme ale duratelor fonemelor: /aa/, /ae/, /ah/, /ao/, /aw/, /ax/, /ax-h/, /axr/

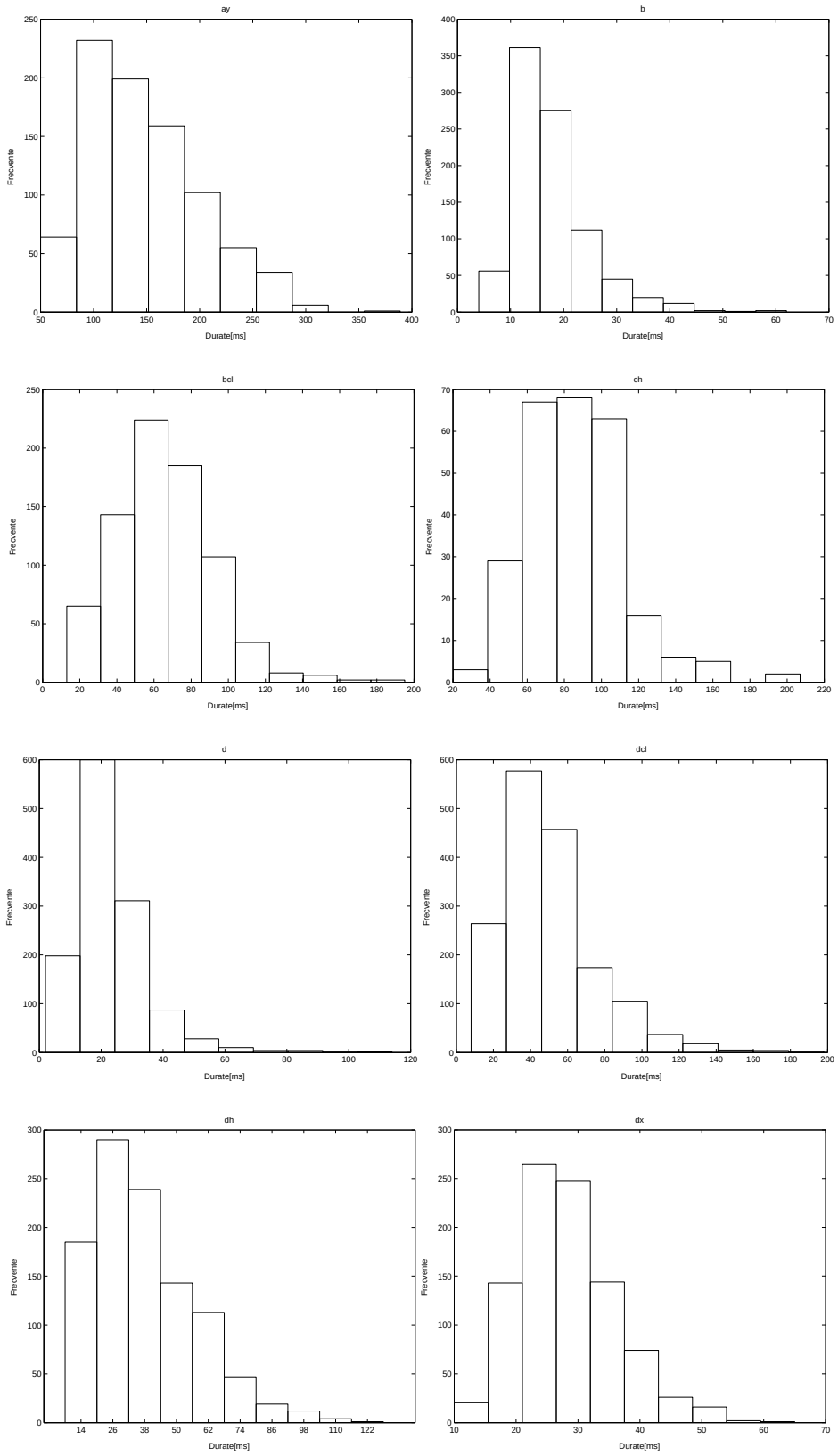


Figura A.2: Histograme ale duratelor fonemelor: /ay/, /b/, /bcl/, /ch/, /d/, /dcl/, /dh/, /dx/

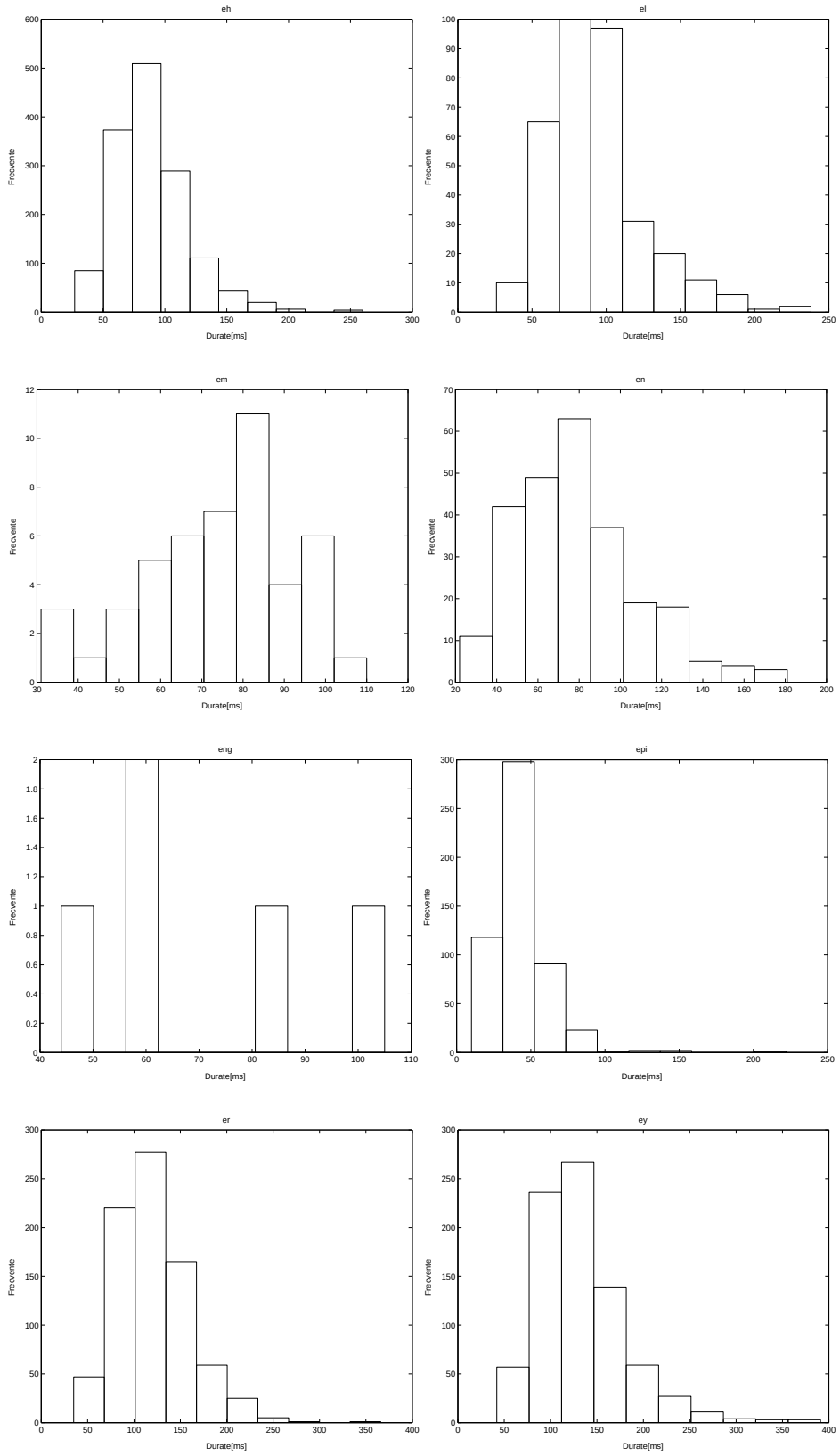


Figura A.3: Histograme ale duratelor fonemelor: /eh/, /el/, /em/, /en/, /eng/, /epi/, /er/, /ey/

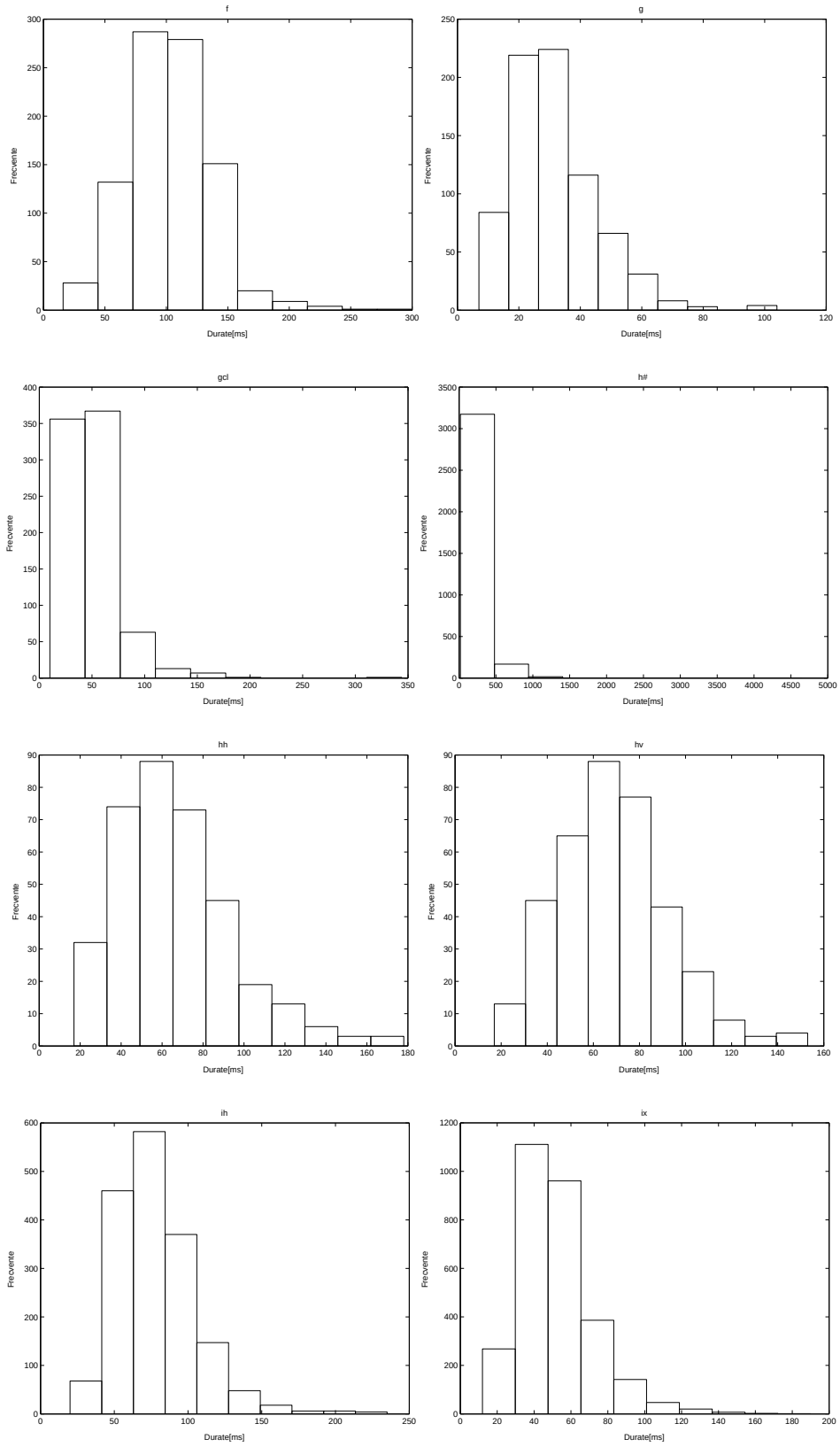


Figura A.4: Histograme ale duratelor fonemelor: /f/, /g/, /gcl/, /h#/, /hh/, /hv/, /ih/, /ix/

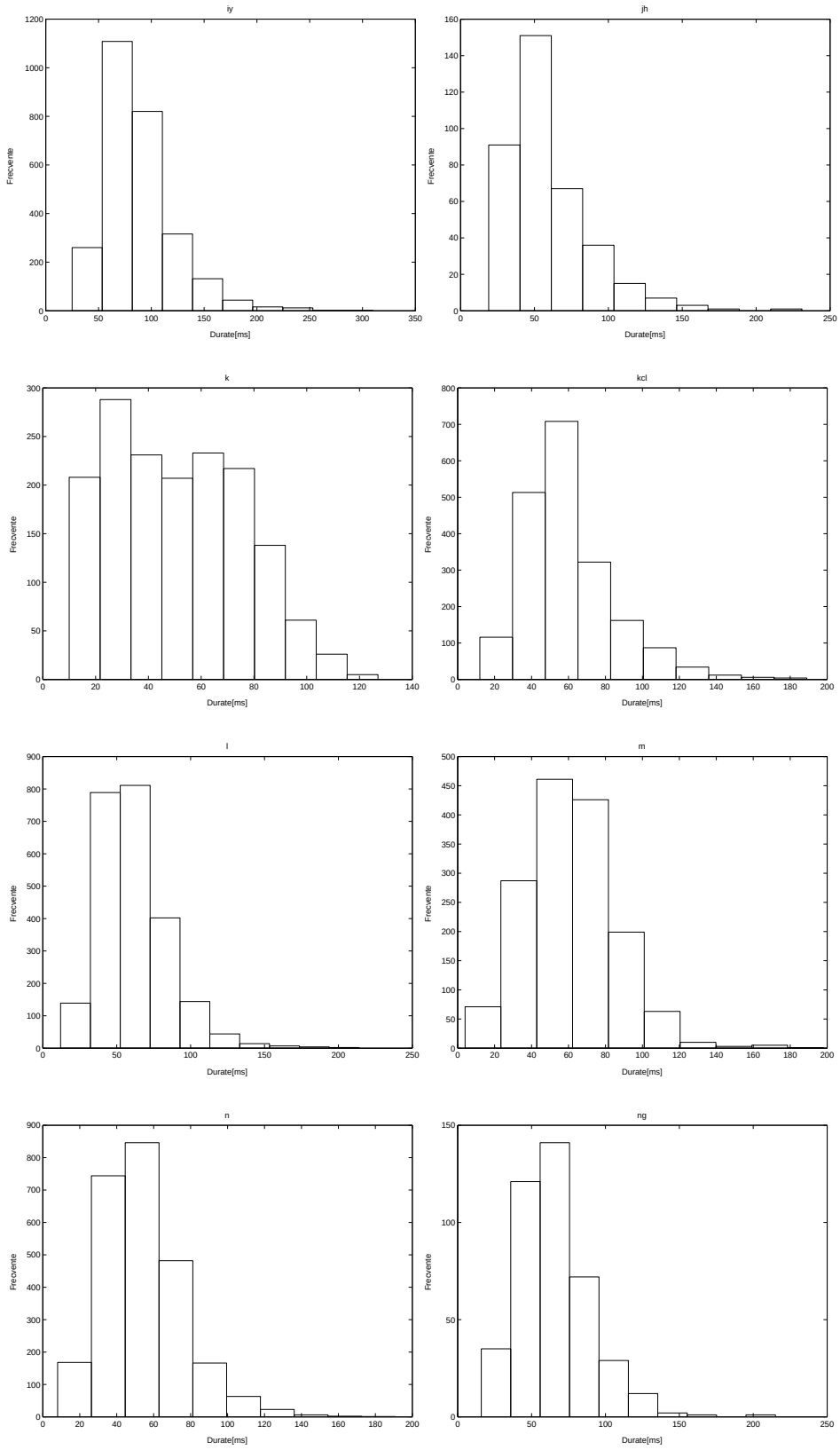


Figura A.5: Histograme ale duratelor fonemelor: /iy/, /jh/, /k/, /kcl/, /l/, /m/, /n/, /ng/

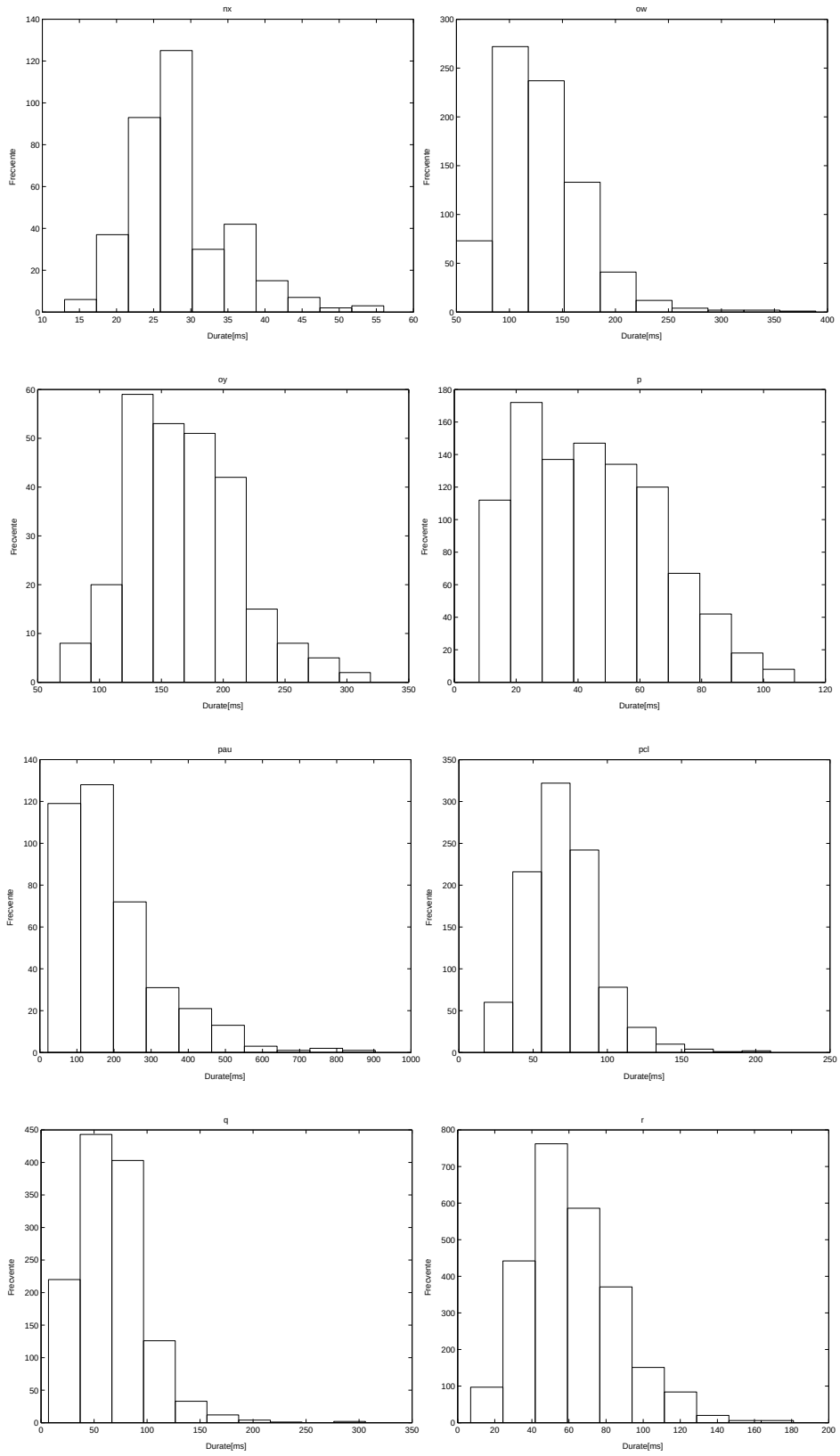


Figura A.6: Histograme ale duratelor fonemelor: /nx/, /ow/, /oy/, /p/, /pau/, /pcl/, /q/, /r/

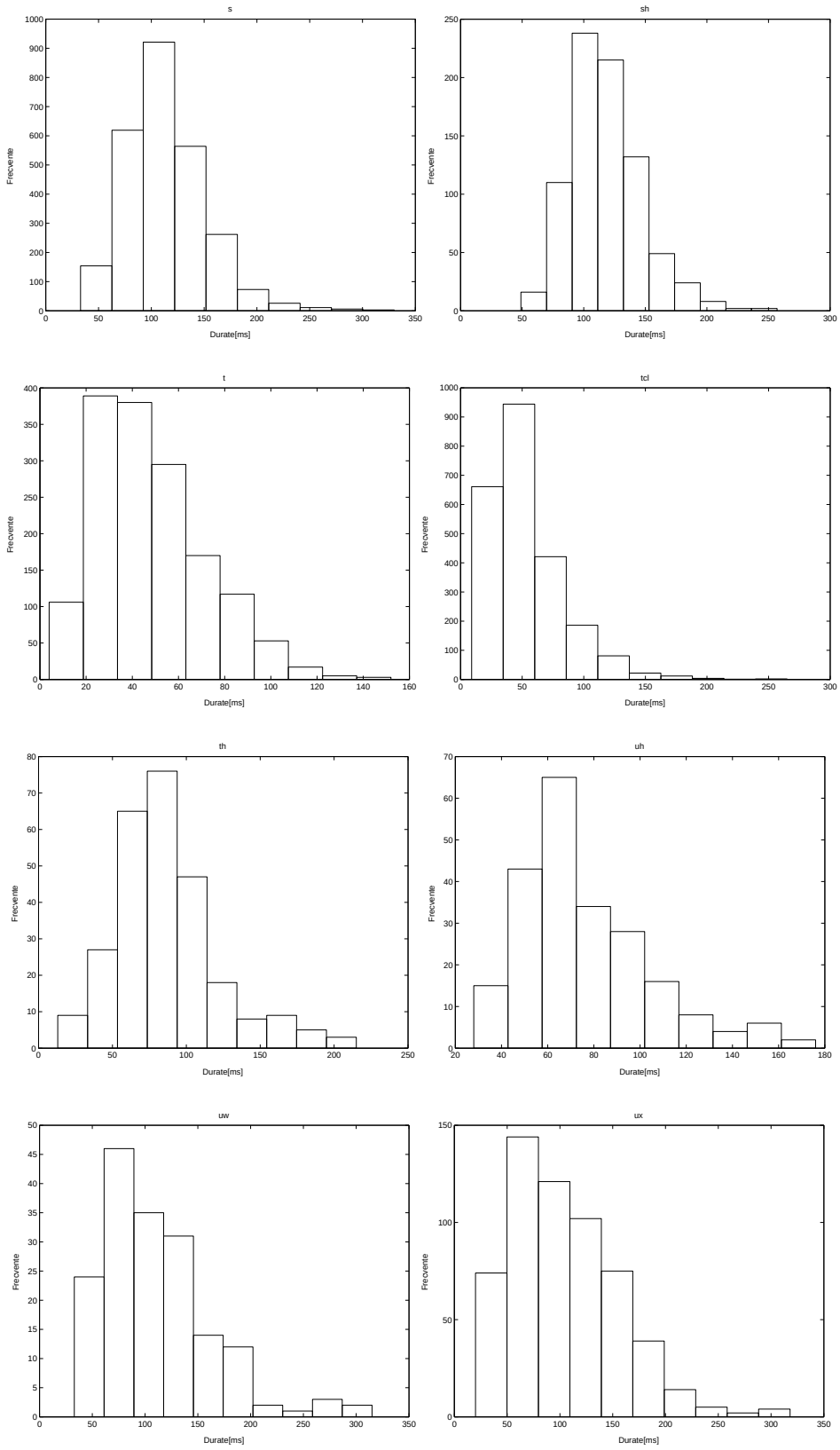


Figura A.7: Histograme ale duratelor fonemelor: /s/, /sh/, /t/, /tcl/, /th/, /uh/, /uw/, /ux/

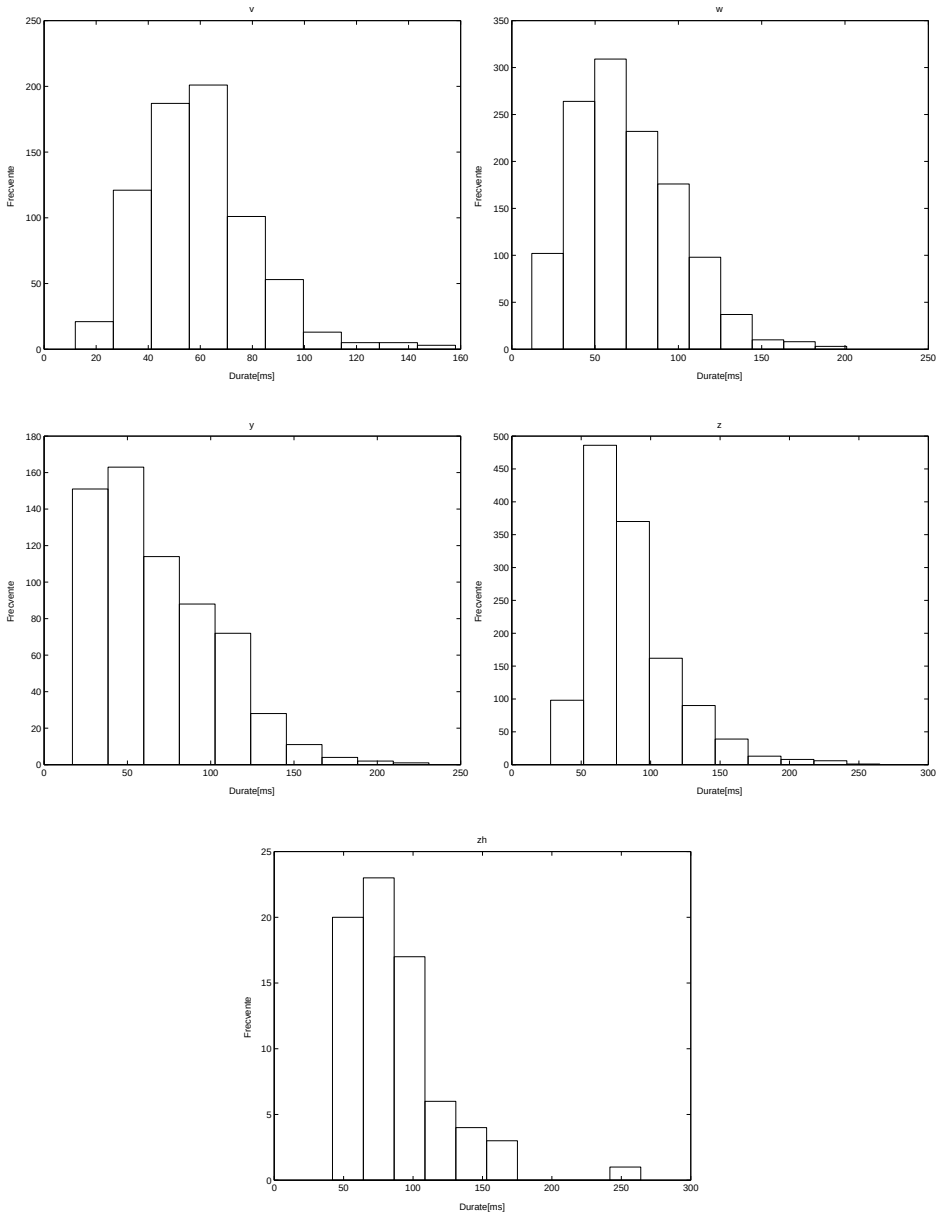
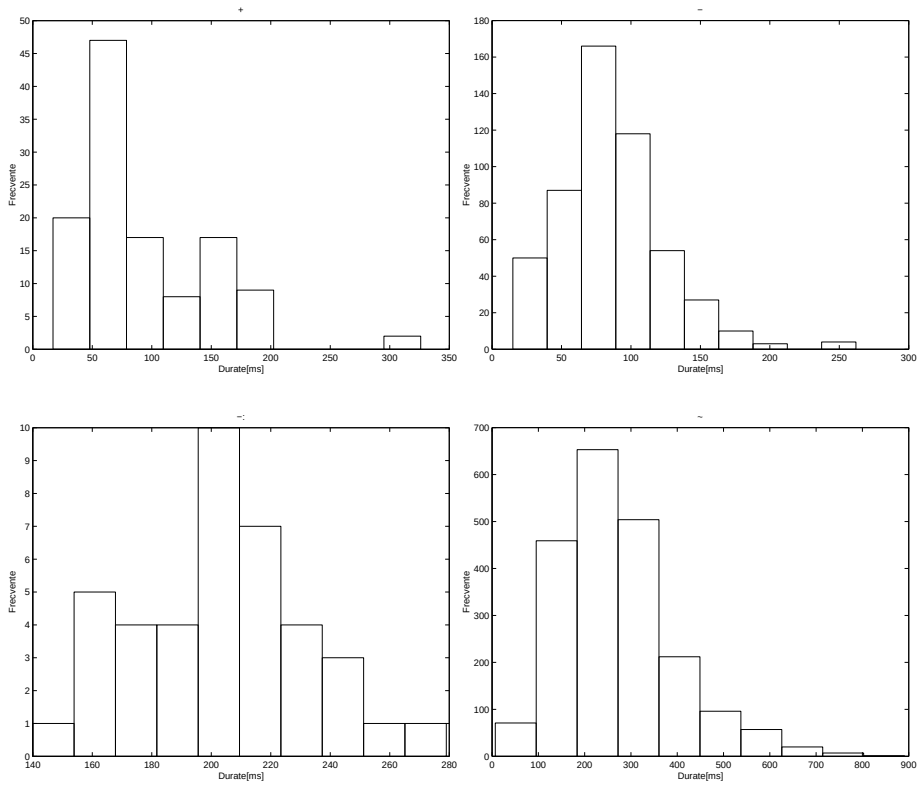


Figura A.8: Histograme ale duratelor fonemelor: /v/, /w/, /y/, /z/, /hz/

Anexa B

Duratele fonemelor din baza de date OASIS



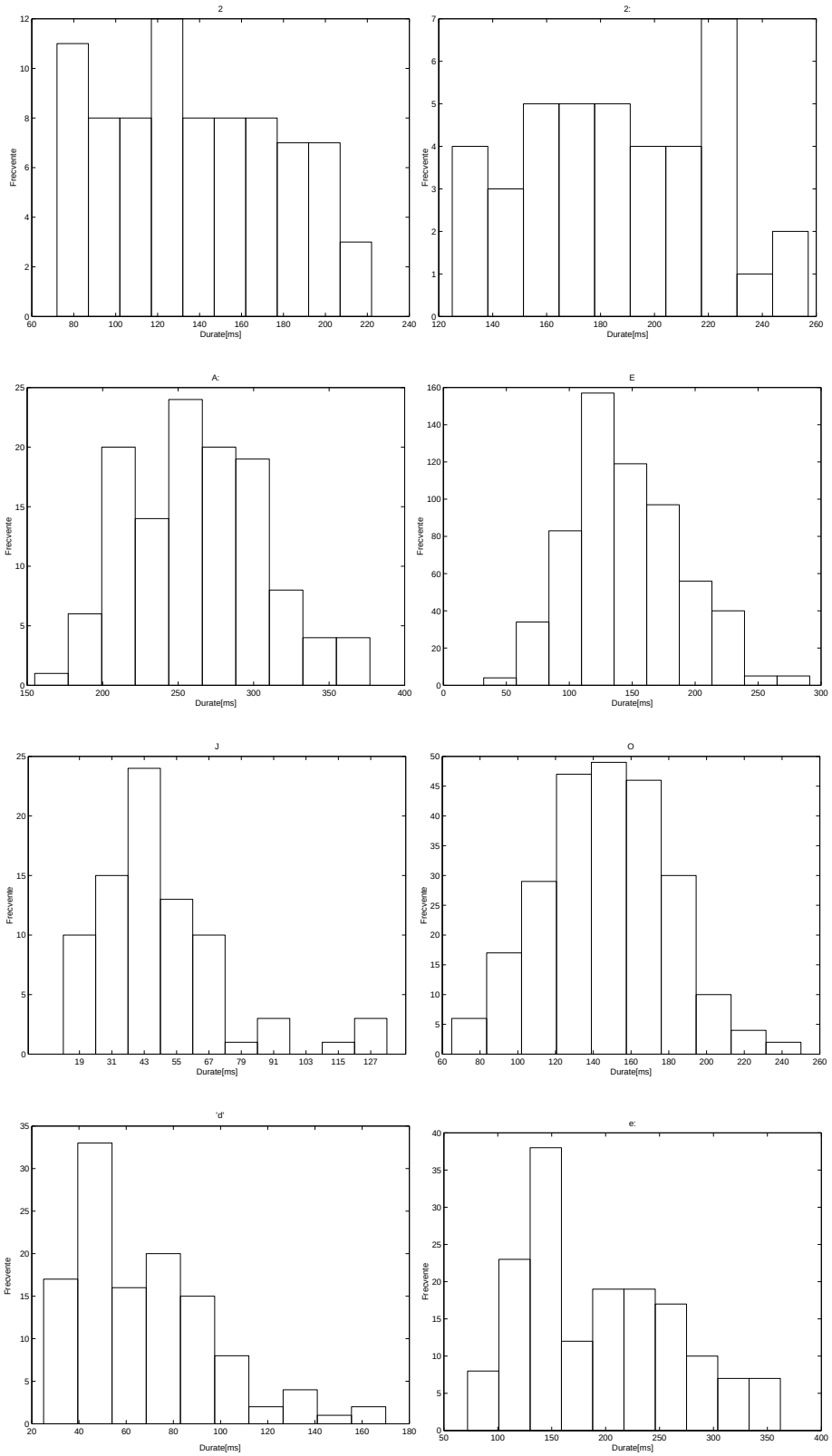


Figura B.1: Histograme ale duratelor fonemelor: /2/, /2:/, /A/, /E/, /J/, /O/, /'d'/, /e:/

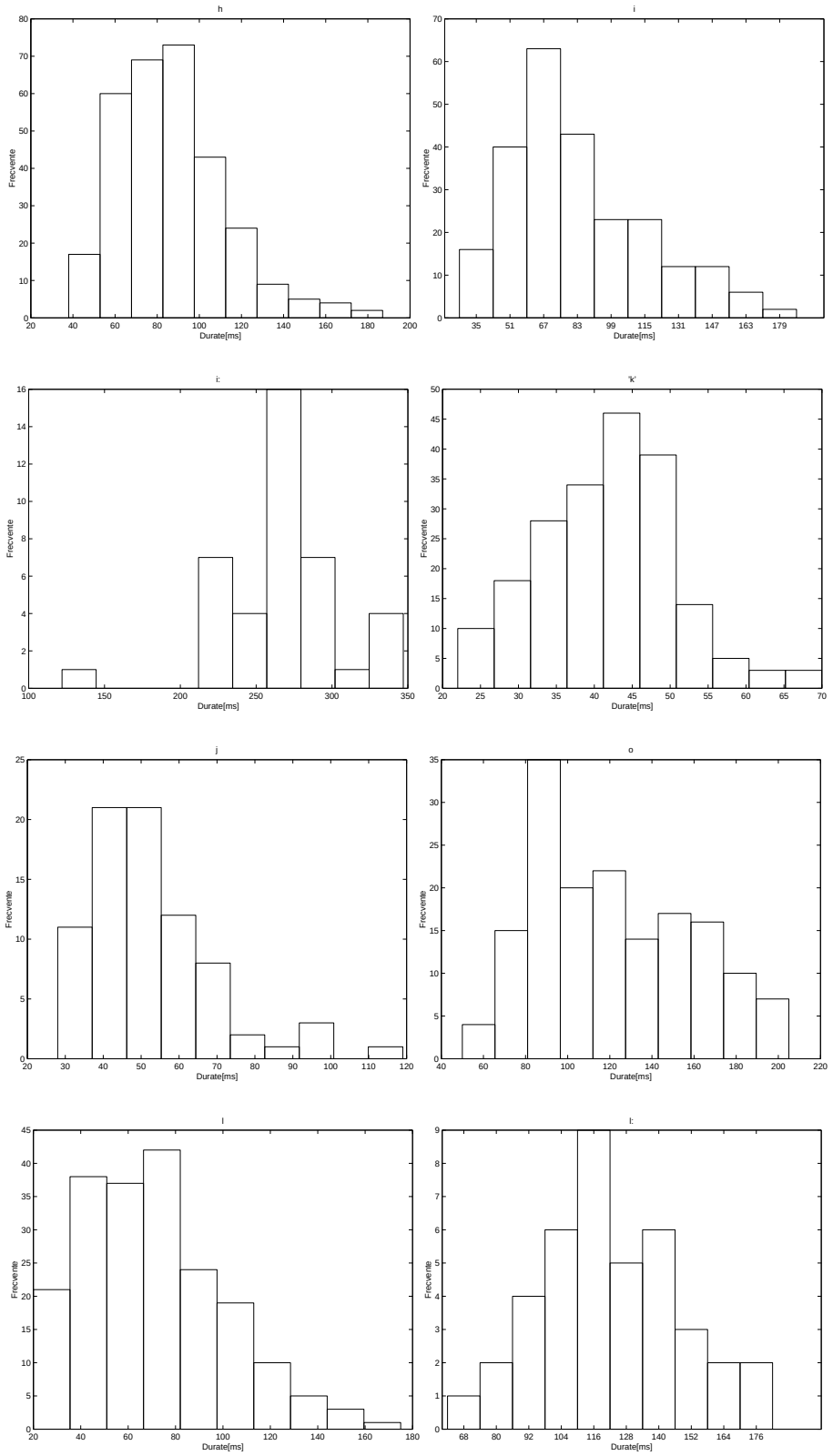


Figura B.2: Histograme ale duratelor fonemelor: /h/, /i/, /i:/, /'k/, /j/, /o/, /l/, /l:/

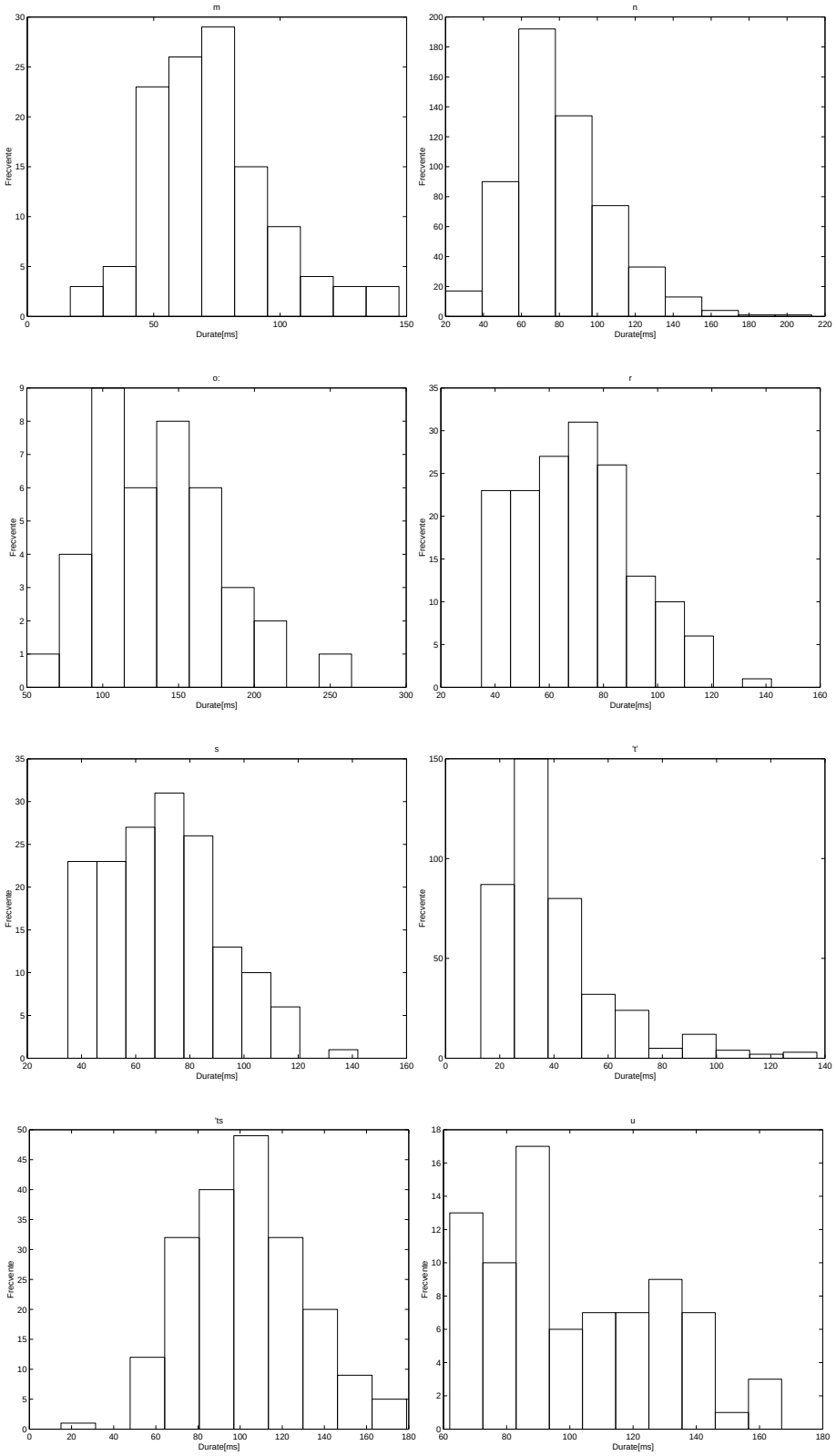


Figura B.3: Histograme ale duratelor fonemelor: /m/, /n/, /o:/, /r/, /s/, /t/, /'ts/, /u/

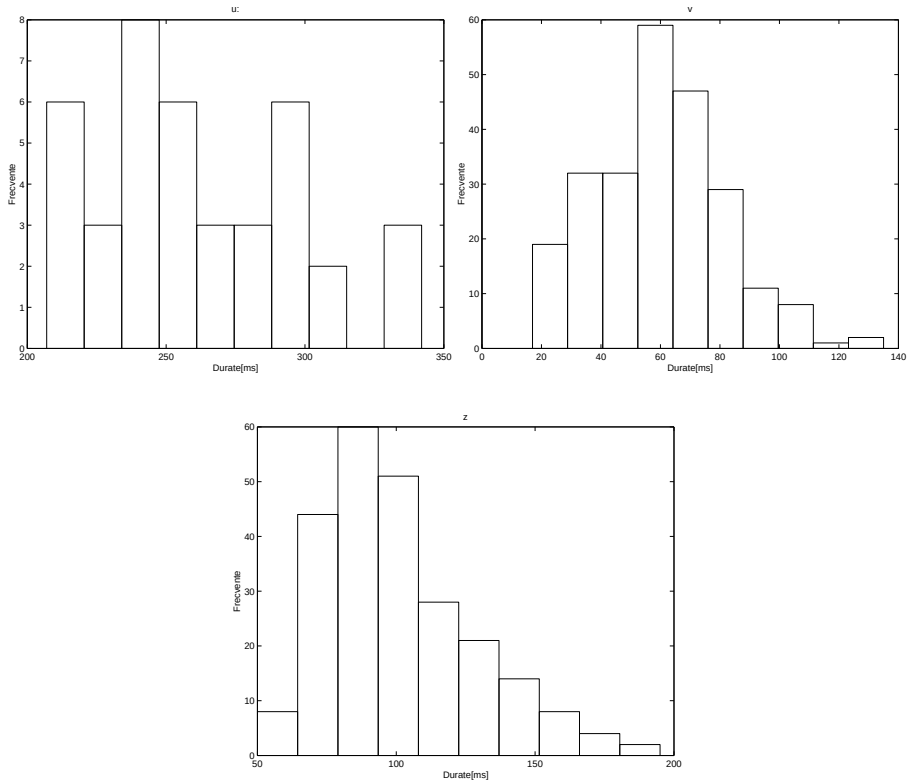


Figura B.4: Histograme ale duratelor fonemelor: /u:/, /v/, /z/

Anexa C

Foneme TIMIT

Pentru segmentarea bazei de date TIMIT s-au utilizat 61 de simboluri. Părţile de închidere ale plozivelor s-au notat separat, astfel simbolurile `bcl, dcl, gcl, pcl, tck, kcl` reprezintă părţile de închidere ale fonemelor `b, d, g, p, t, k`.

Simbolul `h#` s-a utilizat pentru marcarea părţilor de linişte de la începutul, respectiv sfârşitul rostirilor. Există încă două simboluri pentru linişte: `pau` a fost utilizat pentru marcarea liniştii dintre cuvinte, iar `epi` este un fel de linişte, care apare între pronunţarea unei fricative şi a unei semivocale.

Categorie	Simbol	Cuvânt	Transcriere fonetică
Plozive	b	bee	BCL B iy
	d	day	DCL D ey
	g	gay	GCL G ey
	p	pea	PCL P iy
	t	tea	TCL T iy
	k	key	KCL K iy
	dx	muddy, dirty	m ah DX iy, dcl d er DX iy
	q	bat	bcl b ae Q
Africate	jh	joke	DCL JH ow kcl k
	ch	choke	TCL CH ow kcl k
Fricative	s	sea	S iy
	sh	she	SH iy
	z	zone	Z ow n
	zh	azure	ae ZH er
	f	fin	F ih n
	th	thin	TH ih n
	v	van	V ae n
	dh	then	DH e n
Nazale	m	mom	M aa M
	n	noon	N uw N
	ng	sing	s ih NG
	em	bottom	b aa tcl t EM
	en	button	b ah q EN
	eng	washington	w aa sh ENG tcl t ax n
	nx	winner	w ih NX axr

Tabelul 29: Foneme TIMIT-1

Categorie	Simbol	Cuvânt	Transcriere fonetică
Semivocale	l	lay	L ey
	r	ray	R ey
	w	way	W ey
	y	yacht	Y aa tcl t
	hh	hay	HH ey
	hv	ahead	ax HV eh dcl d
	el	bottle	bcl b aa tcl t EL
Vocale	iy	beet	bcl b IY tcl t
	ih	bit	bcl b IH tcl t
	eh	bet	bcl b EH tcl t
	ey	bait	bcl b EY tcl t
	ae	bat	bcl b AE tcl t
	aa	bott	bcl b AA tcl t
	aw	bout	bcl b AW tcl t
	ay	bite	bcl b AY tcl t
	ah	but	bcl b AH tcl t
	ao	bought	bcl b AO tcl t
	oy	boy	bcl b OY
	ow	boat	bcl b OW tcl t
	uh	book	bcl b UH kcl k
	uw	boot	bcl b UW tcl t
	ux	toot	tcl t UX tcl t
	er	bird	bcl b ER dcl d
	ax	about	AX bcl b aw tcl t
	ix	debit	dcl d eh bcl b IX tcl t
	axr	butter	bcl b ah dx AXR
	ax-h	suspect	s AX-H s pcl p eh kcl k tcl t

Tabelul 30: Foneme TIMIT-2

Anexa D

Vocabular OASIS Numbers

Transcriere fonetică	Număr
E + 'd'	1
E z E r	1000
h A: r o m	3
A: r o m	3
h O r m i n - 'ts	30
O r m i n - 'ts	30
h O - 't	6
O - 't	6
h O - 't v O n	60
O - 't v O n	60
h e: - 't	7
e: - 't	7
h E - 't v E n	70
E - 't v E n	70
h u: s	20
u: s	20
h u s o n	20-
u s o n	20-
'k e: - 't E z E r	2000
'k e: - s A: z	200
'k E -: 't 2:	2
'k i l E n - 'ts	9
'k i l E n - 'ts v E n	90
m i l i o:	1000000
n e: + 'd'	4
n E + 'd' v E n	40
n u l: O	0
J j o l - 'ts	8
J j o l - 'ts v O n	80
2 - 't	5
2 - 't v E n	50
s A: z	100
't i: z	10
't i z E n	10-

Tabelul 31: Vocabular alofonic OASIS Numbers

Bibliografie

[1] Antal, M., Statistical methods in speaker recognition, Lucrările sesiunii naţionale de comunicări ştiinţifice, Univ. Petru Maior, Facultatea de Ştiinte Economice şi Administrative, vol. 2, pp. 446-451, 2003.

[2] Antal, M., Statistical Methods for Speaker Identification, Proceedings of the 6th International Conference on Applied Informatics, Jan. 27-31, Eger, Hungary, pp. 65-74, 2004.

[3] Antal, M., Soós, A., Unsupervised Classification for Designing Speaker Identification Systems, Studia Universitatis Babeş-Bolyai Mathematica, Vol. XLIX, No.1, pp. 3-14, 2004.

[4] Antal, M., A Comparison of Parametric Clustering Techniques used in Speaker Identification, IJSIT Lecture Notes of International Conference on Intelligent Knowledge Systems, Aug. 16-20, Assos, Turkey, Vol.1, No.1, pp. 19-25, 2004.

[5] Antal, M., Speaker Independent Phoneme Classification in Continuous Speech, Studia Univ. Babeş-Bolyai Informatica, Vol. XLIX, No. 2, pp. 55-64, 2004.

[6] Antal, M., Phonetic Aspects of Speaker Identification, 15th International Conference in Computer Science and Education SzamOkt-2005, Cluj-Napoca, Romania, Mar. 17-20, pp. 231-238, 2005.

[7] Antal, M., Toderean, G., Phoneme Classification Using Two-Level Classification Scheme, Proceedings of the 3rd Conference of Speech Technology and Human-Computer-Dialog, Cluj-Napoca, Romania, May 13-14, pp. 231-238, 2005.

[8] Antal, M., Toderean, G., Phonetic Analysis of GMM-based Speaker Models, Biometric Authentication Workshop, Cluj-Napoca, May 26-27, pp. 235-240, 2005.

[9] Antal, M., Toderean, G., Speaker Recognition and Broad Phonetic Groups, Proceedings of the 24th IASTED International Multi-Conference on Signal Processing, Pattern Recognition and Applications, Febr. 15-17, Innsbruck, Austria, pp. 155-158, 2006.

[10] Antal, M., Phoneme Recognition for ASR, Proc. of the International Conference Communications 2006, June 8-10, Bucharest, Romania, pp. 123-126, 2006.

[11] Antal, M., Toderean, G., Broad Phonetic Classes Expressing Speaker Individuality, Studia Univ. Babeş-Bolyai Informatica, Vol. LI, No. 1, pp. 49-58, 2006.

[12] Aubert, X. L., A Brief Overview of Decoding Techniques for Large Vocabulary Continuous Speech Recognition, ASR-2000, Paris, France, , Sept. 18-20, pp. 91-97, 2000.

[13] Baum, L., Petrie, T., Statistical Inference for Probabilistic Functions of Finite State Markov chains, Annals of Mathematical Statistics, vol. 37, pp. 1554-1563, 1966.

[14] Becchetti, L. P. Ricotti, Speech Recognition,Theory and C++ Implementation, John Wiley & Sons, 1999.

[15] BenZeghiba, M. F., Bourlard, H., On the Combination of Speech and Speaker Recognition, Proc. Eurospeech, Geneva, Switzerland, pp. 1361-1364, 2003.

[16] Bimbot, F., Bonastre, J-F, Fredouille, C., Gravier, G., Margin-Chagnolleau, I., Meignier, S., Merlin, T., Ortega-Gracia, J., Petrovska-Delacretaz, D., Reynolds, D. A., A Tutorial on Text-Independent Speaker Verification, EURASIP Journal on Applied Signal Processing 4, 430-451, 2004.

[17] Chagnolleau, I.M., Bonastre, J-F., Bimbot, F., Effect of utterance duration and phonetic content on speaker identification using second order statistical methods, Eurospeech, pp. 337-340, 1995.

[18] Bourlard, H., Morgan, N., Connectionist Speech Recognition: A hybrid approach, Kluwer Academic Publishers, 1994.

[19] Bourlard, H., Hermansky, H., Morgan., N., Towards Increasing Speech Recognition Error Rates, Speech Communication, Vol. 18., pp. 205-231, 1996.

[20] Burileanu, D., Sima, M., Negrescu, C., Croitoru, V., Robust Recognition of Small-Vocabulary Telephone Quality Speech, Speech Technology and Human-Computer Dialogue, Ed, Academiei Române, Bucureşti, pp. 145-154, 2003.

[21] Campbell, J.P., Speaker recognition: A tutorial, Proc. IEEE, vol. 85, no. 9., pp. 1437-1462, 1997.

[22] Chengalvarayan, R., Deng, L., Speech Trajectory Discrimination Using the Minimum Classification Error learning, IEEE Transaction on Speech and Audio Processing, Vol. 6, No. 6, pp. 505-515, 1998.

[23] Cormen, T. H., Leiserson, C. E., Rivest, R. R., Introducere în algoritmi, Agora, 2000.

[24] Cover, T. M., Thomas, J. A., Elements of information theory, Wiley New York, 1991.

[25] Das, S. K., Picheny, M. A., Issues in Practical Large Vocabulary Isolated Word Recognition: The IBM Tangora System, Automatic Speech and Speaker Recognition, Advanced Topics, ed. by Lee, C-H, Soong, F. K., Paliwal, K. K., Kluwer Academic Publisher, 1996.

[26] Deller, J.R., Hansen, J. H. L., Proakis, J. G., Discrete-Time Signal Processing of Speech Signals, John Wiley & Sons, 2000.

[27] Dempster, A. P., Laird, N. M., Rubin, D. B., Maximum Likelihood from Incomplete Data via the EM Algorithm, Journal of the Royal Statistical Society, vol. 39, pp. 1-88, 1977.

[28] Doddington, G., Speaker Recognition based on Idiolectal Differences between Speakers, Eurospeech, pp. 2521-2524, 2001.

[29] Duda, R. O. , Hart, P. E., Pattern Classification, John Wiley & Sons, 2000.

[30] Dumitrescu, D., Costin, H., Reţele neuronale, Teora, 1996.

[31] Dumitru, C. O., Gavat, I., Statistical, Neural and Hybrid Methods for Speech Recognition in Romanian Language, Communications, Proc. of the International Conference Communications, June 8-10, Bucharest, Romania, pp. 139-142, 2006.

[32] Eatock, J.P., Mason, J. S., A Quantitative Assessment of the Relative Speaker Discriminating Properties of Phonemes, International Conference on Acoustics, Speech, and Signal Processing, pp. 333-336, 1994.

[33] Falthauser, R., Ruske, G., Improving Speaker Recognition Performance Using Phonetically Structured Gaussian Mixture Models, Eurospeech, pp.751-754, 2001.

[34] Ferrer, L., Bratt, H., Gadde, V. R. R., Kajarekar, S. S., Shriberg, E., Sonmez, K., Stolcke, A., Venkataraman, A., Modeling duration patterns for speaker recognition, Eurospeech, pp. 20017-2020, 2002.

[35] Fukunaga, K., Introduction to Statistical Pattern Recognition, Academic Press, 1990.

[36] Furui, S., An Overview of Speaker Recognition Technology, Automatic Speech and Speaker Recognition, Advanced Topics, ed. by Lee, C-H, Soong, F. K., Paliwal, K. K., Kluwer Academic Publisher, pp. 31-56, 1996.

[37] Furui, S., Recent Advances in Speaker Identification, Pattern Recognition Letters, vol. 18, no. 9, pp. 859-872¸ 1997.

[38] Furui, S., Digital Speech Processing, Synthesis and Recognition, Marcel Dekker, New York, 2001.

[39] House, A. S., and K. N. Stevens, Analog studies of the nasalization of vowels, Journal of Speech and Hearing Disorders, vol. 21, pp. 218-232, 1956.

[40] Huang, X., Acero, A., Hon, H-W., Spoken Language Processing, Prentice Hall, 2001.

[41] Jain, A. K., Duin, R. P. W., Mao, J., Statistical Pattern Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.22, No. 1, pp. 4-37, 2000.

[42] Jang, J-S. R., Sun, C.-T., Mizutani, E., Neuro-Fuzzy and Soft Computing, Prentice-Hall NJ., 1997

[43] Jelinek, F. , Statistical Methods for Speech Recognition, MIT Press, 1997.

[44] Jurafsky, D., Martin, J. H., Speech and Language Processing, Prentice Hall, 2000.

[45] Kinnunen, T., Kilpelainen, T., Franti, O., Comparison of clustering algorithms in speaker identification, Proc. IASTED International Conference on Signal Processing and Communications, pp.222-227, 2000.

[46] Kinnunen, T., Franti, P., Speaker Discriminative Weighting Method for VQ-based Speaker Identification, Proc. 3rd International Conference on audio and video-band biometric person authentication, pp.150-156, Halmstad, Sweden, 2001.

[47] Kocsor, A., Kuba, A. Jr., Tóth, L., An Overview of the OASIS Speech Recognition Project, Proceedings of the 4th International Conference on Applied Informatics, August 30 - September 3, Eger-Noszvaj, Hungary, pp. 94-102, 1999.

[48] Kocsor, A., Tóth, L., Kuba, A., Kovács, K., Jelasity, M., Gyimóthy, T., Csirik J., A Comparative Study of Several Feature Transformation and Learning Methods, International Journal of Speech Technology, pp. Vol. 3, Nr 3/4, pp. 253-262, 2000.

[49] Kohonen, T., Self organizing Maps, Springer, 2001.

[50] Knuth, D. E., Arta programării calculatoarelor, Vol. 3., Sortare şi căutare, Teora, 2002.

[51] Lee, K-F., Hon, H-W., Speaker-independent Phone Recognition Using Hidden Markov Models, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 37, No. 11, 1989.

[52] Lee, C-H., Giachin, E., Rabiner, L. R., Pierraccini, R., Rosenberg, A. E., Improved Acoustic Modeling for Continuous Speech Recognition, Computer Speech and Language, vol. 6, no. 2, pp. 103-127, 1992.

[53] Lee, C-H., Soong, F. K., Paliwal, K. K. (editors), Automatic Speech and Speaker Recognition, Kluwer Academic Publishers, 1996.

[54] Levinson, S. E., Ljolje, A., Miller, L. G., Continuous Recognition from Phonetic Transcription, Proceedings of a workshop on Speech and Natural Language, Pennsylvania, U.S., pp. 190-199, 1990.

[55] Lin, Q., Jan, E-E., Che, C., Yuk, D-S, Flanagan, J. L., Selective use of the speech spectrum and a VQGMM method for speaker identification, Fourth

International Conference on Spoken Language Processing, pp. 1321-1324, 1996.

[56] Logan, B., Moreno, P., Factorial HMMs for acoustic modeling, ICASSP, vol. 2, pp. 813-816, 1998.

[57] Lovell,B.,C., Caelli, T., Hidden Markov Models for spatio-temporal pattern recognition, Handbook of Pattern Recognition and Computer Vision, Chen, C.H., Wang, P.S.P. editors, World Scientific, pp. 25-40, 2005.

[58] Manning, C., Schütze, H., Foundations of Statistical Natural Language Processing, MIT Press Cambridge, 1999.

[59] Mari, J. F., Fohr, D., Junqua, J-C., A second order HMM for high performance word and phoneme-based speech recognition, IEEE Transactions on Speech and Audio Processing, vol. 23, no. 2, pp. 435-438, 1996.

[60] Mason, M., Vogt, R., Baker, B., Shridharan, S., The QUT NIST 2004 Speaker Verification System: A fused acoustic and high-level approach, Proc. of the 10th Australian International Conference on Speech Science & Technology, pp. 398-403, 2004.

[61] Merwe, R., Variations on Statistical Phoneme Recognition - a hybrid approach, master thesis, 1997.

[62] Mohri, M., Riley, M., Hindle, D., Ljolje, A., Pereira, F., Full Expansion of Context-Dependent Networks in Large Vocabulary Speech Recognition, Proc. ICASSP, pp. 665-668, 1998.

[63] Nedic, B., Bourlard, H., Recent Developments in Speaker Verification at IDIAP, IDIAP Research Report 00-26, 2000.

[64] Ney, H., Aubert, X., Dynamic Programming Search Strategies: From Digit Strings to Large Vocabulary Word Graphs, Automatic Speech and Speaker Recognition, Advanced Topics, ed. by Lee, C-H, Soong, F. K., Paliwal, K. K., Kluwer Academic Publisher, pp. 385-411, 1996.

[65] Patane, G., Russo, M., Fully Automatic Clustering System, IEEE Transactions on Neural Networks, Vol. 13, No. 6, November, pp. 1285-1298, 2002.

[66] Pylkkonnen, J., Phone Duration Modeling Techniques in Continuous Speech Recognition, Master thesis Helsinki University of Technology, 2004.

[67] Rabiner, L., R., Juang, B.H., Fundamentals of Speech Recognition, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[68] Rabiner, L. R., A tutorial on hidden markov models and selected applications in speech recognition, Readings in speech recognition, pp. 267–296, 1990.

[69] Rabiner, L. R., Juang, B.-H., Lee, C. -H., An overview of Automatic Speech Recognition, Automatic Speech and Speaker Recognition, Advanced Topics, ed. by Lee, C-H, Soong, F. K., Paliwal, K. K., Kluwer Academic Publisher, pp. 1-30, 1996.

[70] Reynolds, D. A., Speaker identification and verification using Gaussian mixture speaker models, Speech Communications 17, pp. 91-108, 1995.

[71] Reynolds, D. A., Automatic Speaker Recognition using Gaussian Mixture Speaker Models, The Lincoln Laboratory Journal, vol. 8, nr. 2, pp. 173-192, 1995.

[72] Riley, M. D., Ljolje, A., Automatic Generation of Detailed Pronunciation Lexicons, Automatic Speech and Speaker Recognition, Advanced Topics, ed. by Lee, C-H, Soong, F. K., Paliwal, K. K., Kluwer Academic Publisher, pp. 285-301, 1996.

[73] Robinson, T., Fallside, F., A Recurrent Error Propagation Network Speech Recognition System, Computer Speech and Language, Vol. 5, No. 3, pp. 259-274, 1991.

[74] Stapert, R., Mason, J.S., Speaker Recognition and the Acoustic Speech Space, Odyssey Speaker Recognition Workshop, Crete, pp. 195-199, 2001.

[75] Stolojanu, G., Podaru, V., Cetină, F., Prelucrarea numerică a semnalului vocal, Editura Militară, 1984.

[76] Szarvas, M., Matsunaga, S., Improving Phoneme Classification Performance Using Observation Context-Dependent Segment Models, International Journal of Speech Technology, pp. Vol. 3, Nr 3/4, pp. 263-276, 2000.

[77] Szarvas, M., Fegyó, T., Mihajlik, P., Tatai, P., Automatic Recognition of Hungarian: Theory and Practice, International Journal of Speech Technology, pp. Vol. 3, Nr 3/4, pp. 237-251, 2000.

[78] Schwarz, P., Matejka, P., Cernocky, J., Recognition of Phoneme Strings using TRAP Technique, Eurospeech, Geneve, pp. 825-828, 2003.

[79] Toderean, G., Caruntu, A., Metode de recunoaşterea vorbirii, Risoprint, Cluj-Napoca, 2005.

[80] Tóth, L., Kocsor, A., Csirik, J., On naive Bayes in Speech Recognition, In. J. Appl. Math. Comput. Sci., Vol. 15, No. 2, pp. 287-294, 2005.