UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE ELECTRONICĂ, TELECOMUNICAȚII ȘI TEHNOLOGIA INFORMAȚIEI

SERVICIU WEB DE AUTENTIFICARE PRIN VOCE. SISTEME DE RECUNOAȘTERE DE CIFRE CONECTATE ȘI VERIFICARE DE VORBITOR

LUCRARE DE DIPLOMĂ

Prezentată ca cerinţă parţială pentru obţinerea
titlului de *Inginer*
în domeniul *Electronică, Telecomunicaţii şi Tehnologia Informaţiei*
programul de studii *Calculatoare și Tehnologia Informației*

Profesor coordonator:
Ș.l. Ing. Horia Cucu
Prof. Dr. Ing. Corneliu Burileanu

Student:
Ioana-Alina Bănică

București
2015

DECLARAŢIE DE ONESTITATE ACADEMICĂ

Prin prezenta declar că lucrarea cu titlul "Serviciu Web de autentificare prin voce. Sisteme de recunoaștere de cifre conectate și verificare de vorbitor", prezentată în cadrul Facultăţii de Electronică, Telecomunicaţii şi Tehnologia Informaţiei a Universităţii „Politehnica" din Bucureşti ca cerinţă parţială pentru obţinerea titlului de *Inginer* in domeniul Inginerie Electronică şi Telecomunicaţii, programul de studii *Calculatoare și Tehnologia Informației* este scrisă de mine şi nu a mai fost prezentată niciodată la o facultate sau instituţie de învăţământ superior din ţară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referinţe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi fac referinţă la sursă. Reformularea în cuvinte proprii a textelor scrise de către alţi autori face referinţă la sursă. Înţeleg că plagiatul constituie infracţiune şi se sancţionează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor şi măsurătorilor pe care le prezint ca fiind făcute de mine, precum şi metodele prin care au fost obţinute, sunt reale şi provin din respectivele simulări, experimente şi măsurători. Înţeleg că falsificarea datelor şi rezultatelor constituie fraudă şi se sancţionează conform regulamentelor în vigoare.

Bucureşti, *Data* Ioana-Alina Bănică

Cuprins

LISTĂ DE FIGURI

LISTĂ DE TABELE

LISTĂ DE ACRONIME

REST - Representational State Transfer

IPA - International Phonetic Alphabet

GMM - Gaussian Mixture Model

UBM - Universal Background Model

DCT - Discrete Cosine Transform

LPC - Linear Predictive Coefficients

PLP - Linear Predictive Coefficients

MFCC - Mel Frequency Cepstral Coefficients

HMM - Hidden Markov Model

SVM - Support Vector Machine

NN - Neural Networks

LIUM - Le Laboratoire d'Informatique de l'Université

EM - Expectation Maximization

MAP - Maximum-a-Posteriori

HTTP - Hypertext Transfer Protocol

FRR - False Rejection Rate

FAR - False Acceptance Rate

WER - Word Error Rate

SER - Sentence Error Rate

URL - Uniform Resource Locator

URI – Uniform Resource Identifier

RPC - Remote Procedure Call

SOAP – Simple Object Access Protocol

RAV – Recunoaștere Automată a Vorbirii

CORBA - Common Object Request Broker Architecture

INTRODUCERE

MOTIVAȚIA LUCRARII

Vorbirea este cel mai natural mod de comunicare, iar scopul fundamental este acela de a transmite informații. Comunicarea între oameni se face foarte rapid prin vorbire și s-a dorit o astfel de comunicare rapidă și între om și calculator, de aceea s-au dezvoltat mai multe sisteme de interacțiune om-mașină bazate pe voce, precum sisteme de recunoaștere automată de vorbire sau sisteme pentru sinteza vorbirii.

Procesarea vorbirii este un domeniu care a evoluat mult odată cu creșterea performanțelor sistemelor de calcul. Cu ajutorul tehnologiei care s-a dezvoltat în ultimii 50 de ani putem extrage foarte multe informații dintr-un semnal vorbit, chiar de durată scurtă, de cateva milisecunde. Prin analiza unui semnal vocal putem răspunde la intrebări precum: cine a fost, ce a spus, etc.

Tehnologia vorbirii este folosită tot mai des și în securizarea accesului la resurse critice din punct de vedere al securității. In felul acesta o persoană nu mai este nevoită sa memoreze o parolă sau să păstreze o cheie sau un card de acces care pot fi cu ușurință pierdute sau furate. O problemă importantă care se distinge aici este precizia recunoașterii si erorile care apar în procesul de captare a semnalului vocal. Se dorește construirea unui sistem cât mai robust, care

să asigure o potrivire perfectă la fiecare comparare a datelor biometrice curente cu datele de referință din baza de date. O potrivire perfectă este totuși lipsită de o utilitate practică, deoarece in realitate puțini ar putea folosi sistemul pentru că utilizatorii care doresc accesul la resursele securizate ar fi respinși tot timpul. Pentru a evita acest lucru se acceptă o variabilitate a datelor biometrice, aleasă in funcție de nivelul de securitate dorit.

Lucrarea de față este motivată de toate aceste lucruri menționate și își propune crearea unui sistem de autentificare în cont pe bază de amprentă vocală care să asigure un nivel ridicat de securitate.

APLICAȚII ALE RECUNOAȘTERII VORBIRII

Un sistem de recunoaștere automată de vorbire este un sistem care primește la intrare un semnal audio ce conține vorbire, iar la ieșire generază o succesiune de cuvinte pe baza semnalului de intare. Recunoașterea automată a vorbirii o găsim in multe domenii, avand o aplicabilitate foarte largă. Sunt trei categorii importante de aplicații: dictarea, indexarea audio si dialogul om-mașină.

Dictarea presupune transcrierea unui semnal vocal. În multe domenii sunt folosite astfel de aplicații de transcriere, cum ar fi: transcrierea discuțiilor în sălile de judecată, editarea unor documente, etc. Sarcina de recunoaștere a vorbirii poate fi dificilă, deoarece procesul de recunoaștere depinde de mai mulți factori, precum: vorbitorul (care ar putea fi cunoscut sau necunoscut), mediul acustic, specificitatea limbii, numarul de cuvinte ce pot fi recunoscute, stilul de vorbirii, etc. Dimensiunea vocabularului este un alt factor important în sarcinile de recunoaștere de vorbire, deoarece pentru a avea un sistem robust este necesară o bază de date mare, in care să avem achiziționate multe resurse acustice.

Aplicațiile de recunoaștere de vorbire poate fi clasificate și in funcție de dificultatea de recunoaștere. Astfel avem o dificultate mai mare atunci când se dorește o recunoaștere de vorbire spontană, față de recunoașterea vorbirii citate sau de recunoașterea de cuvinte izolate. Pentru a măsura nivelul de performanță a unui sistem de recunoaștere se calculează rata de cuvinte eronate care reprezintă un criteriu de performanță standard. Rezultatele obținute pentru diferite sarcini de recunoaștere de vorbire sunt prezentate in Tabelul 1.

Sarcina de RAV	Dimensiunea vocabularului	WER [%]
TI Digits	11 cuvinte	0.55
Wall Street Journal read speech	5000 cuvinte	3.0
Wall Street Journal read speech	20000 cuvinte	<6.6
Broadcast News	64000+ cuvinte	9.9
Conversational Telephone Speech	64000+ cuvinte	20.7

Tabelul 1 Rezultatele WER obținute pentru diverse sarcini RAV

Indexarea audio este o altă categorie de aplicație a recunoașterii vorbirii. Indexarea audio presupune indexarea materialelor audio înregistrate, iar odată creat, textul transcris este stocat într-o bază de date si poate fi accesat într-un mod similar celui căutarii în paginile web bazate pe text.

Dialogul om-mașină reprezintă o alt categorie de aplicație pentru recunoașterea vorbirii. Aplicațiile de tipul acesta presupun un nivel scăzut de dificultate în recunoaștere, de cele mai multe ori recunoaștere constă doar în recunoașterea unor cuvinte izolate, de exemplu

parcurgerea unui meniu, modificarea temperaturii ambientului, modificarea luminării camerei, etc.

APLICAȚII ALE RECUNOAȘTERII DE VORBITOR

Pentru recunoaşterea de vorbitor avem două sarcini fundamentale: identificarea si verificarea vorbitorului. Identificarea unei persoane este folosită pentru a determina cine este un individ dintr-un grup de vorbitori cunoscuți sau pentru a se verifica daca vorbitorul aparține unui anumit grup. Identificarea unui vorbitor presupune identificarea unei voci necunoscute dintr-un set predefinit de vorbitori. Aplicațiile de identificare de vorbitor sunt limitate, deoarece pentru ca un vorbitor să fie identificat acesta trebuie să fie cunoscut de sistem. Există si aplicații adaptive de identificare, în care o voce necunoscută este atribuită vorbitorului pentru care se aseamănă acustic cel mai bine.

În identificarea persoanelor se disting două faze: faza de antrenare si faza de autentificare. În faza de antrenare se va folosi semnalul vocal de la fiecare vorbitor si se vor crea modele pentru fiecare dintre acesta. În etapa de decodare se va compara semnalul vocal primit cu toate modelele existente in sistem, iar vorbitorul va fi identificat ca fiind vorbitorul al cărui model a obținut scorul cel mai bun. Performanța unui astfel de sistem este măsurată de rata de identificare.

În cazul verificării, semnalul vocal al unui individ este folosit pentru a verifica identitatea pretinsă de acesta. Pentru fiecare vorbitor care dorește înrolarea în sistem se creează un model. Modelul va fi folosit în etapa de decodare, unde semnalul vocal al unui individ va fi comparat cu modelul vorbitorului pretins. În cazul verificării persoanelor există două metode de măsurare a nivelului de performanță: eroarea de falsă rejecție si eroarea de falsă acceptare. Eroarea de falsă rejecție reprezintă numarul de ori în care vorbitorul adevarat (vorbitorul este chiar cel care se pretinde) este rejectat de către sistem. Eroarea de falsă acceptare reprezintă raportul dintre numărul de acceptări false (un vorbitor a fost acceptat, deși este un impostor) si numărul de încercari de identificare. La proiectarea unui sistem de verificare de vorbitor trebuie să se țină cont de aceste erori, în funcție de nivelul de securitate dorit. Dacă se dorește un sistem cu nivel înalt de securitate, atunci rata de falsă acceptare se dorește a fi mică, chiar nulă, dar asta presupune o rată de falsă rejecție mare ce poate fi supăratoare. Proiectantul unui sistem de recunoaștere de vorbitor va balansa cele două rate de eroare astfel încât să obțină un sistem ce răspunde nevoilor dorite.

Aplicațiile de recunoaștere de vorbitor au si ele o largă aplicabilitate și pot fi întâlnite în diverse domenii. Una dintre cele mai întalnite aplicații ale recunoașterii de vorbitor este aceea de autentificare, pe bază de amprentă vocală, într-un sistem ce partajează resurse securizate. Sistemele de recunoaștere de vorbitor sunt întâlnite în domenii precum: domeniul bancar și telecomunicații, unde se dorește identificarea clienților în urma unei conversații uzuale, domeniul judiciar, unde se caută anumite caracteristici ale semnalului vocal al unui anumit suspect, etc.

OBIECTIVELE LUCRARII

Lucrarea de față își propune realizarea/descrierea unui sistem robust de autentificare pe bază de voce care să asigure un nivel înalt de securitate. Sistemul este alcătuit din două module, un modul de recunoaștere de cifre si un modul de recunoaștere de vorbitor. Ambele module au fost proiectate astfel încât să asigure un nivel ridicat de securitate. Pentru proiectarea unor astfel de module care să respecte aceste cerințe s-au realizat mai multe experimente prin care s-au creat numeroase modele. Dintre toate modelele create s-au păstrat acele modele pentru

care am obținut rezulatele cele mai bune, asta însemnând pentru sistemul de recunoaștere de vorbire o rată de cuvinte eronată mică, iar pentru sistemul de recunoaștere de vorbitor s-au creat modele pentru fiecare vorbitor după modelul acelora care au obținut o rată de falsă acceptare si o rată de falsă rejecție cât mai mică.

În capitolul 2 se va prezenta imaginea de ansamblu a sistemului de autentificare pe bază de amprentă vocală, cât si arhitectura acestuia. Capitolul va descrie modul de funcționare a întregului sistem, dar și a fiecărui modul în parte. Astfel în acest capitol vor fi introduse serviciul de recunoaștere de vorbitor și serviciul de recunoaștere de cifre conectate.

Capitolul 3 este dedicat tehnologiilor software folosite pentru dezvoltarea proiectului. Se vor prezenta atât avantajele cât si dezavantajele folosirii serviciului REST și de asemenea tot în acest capitol se va detalia si modul în care sistemul de recunoaștere de cifre, cât si sistemul de recunoaștere de vorbitor au fost expuse ca și servicii REST.

În capitolele 4 și 5 se va descrie în detaliu serviciul de recunoaștere de cifre conectate, respectiv serviciul de recunoaștere de vorbitor. Pentru fiecare serviciu vor fi prezentate arhitectura, modul de implementare, iar la sfârșit se vor prezenta rezultatele obținute în urma evaluării sistemului.

Ultimul capitol este dedicat concluziilor și contribuțiilor personale.

CAPITOLUL 1

RECUNOAȘTEREA LIMBAJULUI VORBIT ȘI RECUNOAȘTEREA VORBITORULUI

1.1 ARHITECTURA GENERALĂ A UNUI SISTEM DE RECUNOAȘTERE A VORBIRII

Sarcina de recunoaștere de vorbire poate avea diverse grade de dificulate, în funcție de tipul de vorbire care se dorește a fi recunoscut. Astfel sarcina de recunoaștere a vorbirii spontane va fi mai grea decât cea a recunoașterii vorbirii continue sau a recunoașterii de cuvinte izolate. Arhitectura unor astfel de sisteme, de recunoaștere de vorbire continuă sau spontană, este prezentată în Fig. 1.1.

Procesul de recunoaștere a limbajului vorbit, deci de transformare a vorbirii în text, se realizează pe baza tehnicilor de modelare statistică care presupune crearea modelelor pe baza unor cantități mari de date. Transformarea limbajului vorbit în text se va reduce la determinarea celei mai probabile secvente de cuvinte W* din limba L, dat fiind mesajul X și se poate reprezenta astfel:

$$W^* = \underset{w}{argmax}\, p(W|X) \;=\; \underset{w}{argmax}\, \frac{p(X|W)\cdot p(W)}{p(X)} = \underset{w}{argmax}\, p(X|W) \;\cdot p(W)$$

Estimarea celei mai probabile secvențe de cuvinte, dat fiind un mesaj vorbit X se realizează datorită modelelor de limbă, respectiv a modelului acustic. Asfel probabilitatea apriori a secvenței de cuvinte este estimată pe baza modelului de limbă, iar estimarea probabilității dată fiind secventa de cuvinte p(X/W) se face pe baza modelului acustic. Așa cum se observă din Fig. 1.1 cele doua modele, modelul acustic și modelul de limbă, sunt conectate prin intremediul modelului fonetic.

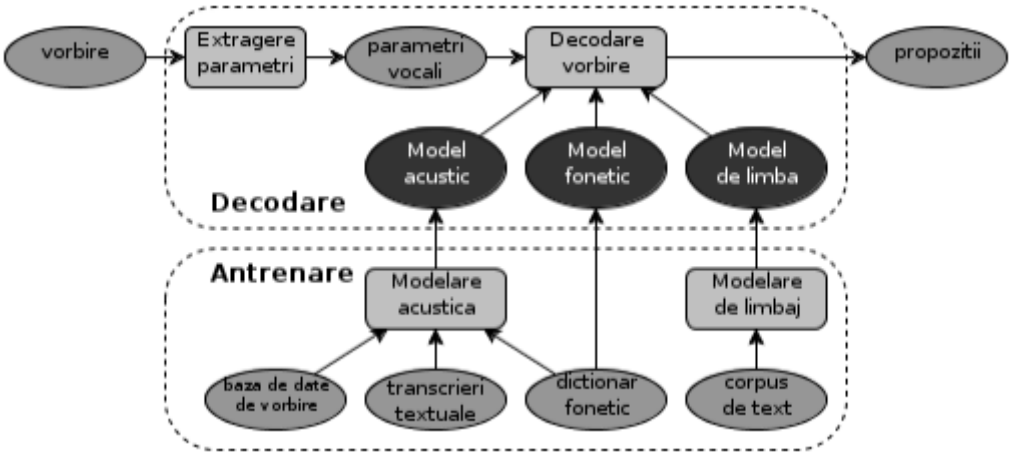

Fig. 1.1 Arhitectura generală a unui sistem de recunoaștere de vorbire

Modelul de limbă este cel care se asigură că secvența de cuvinte rostită este o secvență cu sens. În cazul unor fraze asemanătoare acustic modelul de limbă va atribui o probabilitate mai mare unei secvențe cu sens, decât uneia irelevante, ajutând sistemul de recunoaștere în luarea deciziei corecte, deci de a considera că fraza cu sens a fost cea rostită. Acest model este construit cu ajutorul unui corpus de text de dimensiuni mari care să permită extragerea unor statistici caracteristice limbii.

Modelul acustic are rolul de a estima probabilitatea să fi fost rostit un mesaj vorbit, dată fiind o succesiune de cuvinte. Acesta se construiește pe baza unui set de înregistrari audio, fiecare înregistrare având asociată și o transcriere textuală și de asemenea necesită existența unui dicționar care să indice modul de pronunție al acestora. În sistemele de recunoaștere de vorbire nu se folosesc modele acustice care au ca unități fonetice de bază cuvinte, ci se folosesc modele create pe baza unor unitați sub-lexicale numite fenome sau a unor unități sub-fonetice numite senone.

Fonemele reprezintă unitați sublexicale și reprezintă cea mai mică unitate fonetică cu sens prezentă în vorbire, fiind similară cu litera în limbajul scris. În Tabelul 1.1 sunt prezentate fonemele limbii. De aici putem trage concluzia că literele se pronunță diferit în funcție de literele vecine, de exemplu litera *o* din cuvântul *loc* se va pronunța diferit de aceeși literă din cuvântul *oase*. Astfel litera *o* este reprezentată de două simboluri IPA pentru foneme, în funcție de pronunție.

Modelul fonetic este cel care conectează modelul acustic de modelul de limbă și este de cele mai multe ori un dicționar care specifică modul de pronunție al cuvintelor, mai exact asociază fiecărui cuvânt din vocabular una sau mai multe secvențe de foneme adecvate.

Fonemul			Exemple de cuvinte	
Tip	Simbol IPA	Simbol intern	Forma scrisă	Forma fonetică
vocale	a	a	sat	s a t
	e	e	mare	m a r e
	i	i	lift	l i f t
	o	o	loc	l o c
	u	u	şut	s1 u t
	ə	a1	gură	g u r a1
	ɨ	i2	între	i2 n t r e
vocale împrumutate	y	y	ecru	e c r y
	ø	o2	bleu	b l o2
semivocale	ę	e1	deal	d e1 a l
	j	i3	fiară	f i3 a r a1
	ǫ	o1	oase	o1 a s e
	w	w	sau	s a w
consoane	c	k2	chem	k2 e m
	b	b	bar	b a r
	p	p	par	p a r
	k	k	acum	a k u m
	ʧ	k1	cenuşă	k1 e n u s1 a1
	g	g	galben	g a l b e n
	ʤ	g1	girafă	g1 i r a f a1
	ɹ	g2	unghi	u n g2
	d	d	dar	d a r
	t	t	tot	t o t
	f	f	faţa	f a t1 a
	v	v	vapor	v a p o r
	h	h	harta	h a r t a
	ʒ	j	ajutor	a j u t o r
	ʃ	s1	coş	k o s1
	l	l	lac	l a c
	m	m	măr	m a1 r
	n	n	nas	n a s
	s	s	sare	s a r e
	z	z	zar	z a r
	r	r	risc	r i s k
	ʦ	t1	ţăran	t1 a1 r a n
consoană palatalizată	ʲ	i1	tari	t a r i1

Tabelul 1.1 Fonemele limbii române

1.2 ARHITECTURA GENERALA A UNUI SISTEM DE RECUNOAŞTERE DE VORBITOR

Sistemele de recunoaştere de vorbitor pot fi sisteme de identificare sau sisteme de verificare de vorbitor. Există două tipuri de sisteme de identificare: sisteme cu set deschis de identificare, unde vorbitorul poate fi o persoană arbitară, deci poate fi şi un vorbitor necunoscut sistemului. În cazul acesta se va verifica dacă vorbitorul aparţine sau nu grupului de vorbitori cunoscuţi de sistem, iar dacă acesta aparţine grupului atunci se va determina cine este individul din grupul respeciv. În sistemele cu set închis de identificare vorbitorul trebuie să fie o persoană cunoscută sistemului. În cazul acesta recunoaşterea se rezumă la identificarea individului din respectivul grup. O problemă mare la sisteme de identificare de vorbitor este că semnalul vocal de la un vorbitor, fie el şi necunoscut sistemului, va fi comparat cu modelul fiecărui vorbitor din grup, iar modelul care va obţine cel mai mare scor va fi desemnat câştigător, deci vorbitorul va fi considerat ca fiind individul al cărui model a obţinut scorul cel mai bun. Arhitectura unui sistem de identificare de vorbitor este prezentată in Fig. 1.2.

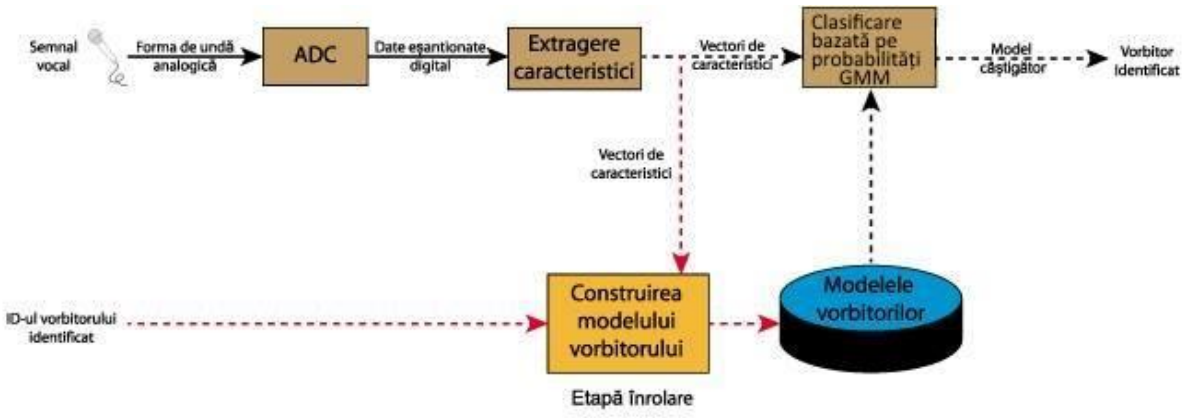

Fig. 1.2 Arhitectura generală a unui sistem de identificare de vorbitor

Sistemele de verificare de vorbitor vor compara carateristicile semnalului vocal al unui vorbitor cu modelul individului care se pretinde. Arhitectura unui sistem de verificare de vorbitor este prezentată in Figura 2.3. Ca şi la sistemele de identificare de vorbitor, configurarea sistemului se realizează în etapa de antrenare sau la înrolarea în sistem a unui vorbitor care va trebui să furnizeze probe de vorbire care vor fi folosite pentru a antrena modelul pentru acel vorbitor. În sistemele de verificare, semnalul vocal de la un vorbitor necunoscut este comparat cu modelul vorbitorului pretins şi cu un model general de vorbire ce modeleză vorbirea umană.

De obicei aceste sisteme de verificare de vorbitor sunt folosite impreună cu un sistem de recunoaştere de vorbire. Astfel vor exista două tipuri de sisteme de verificare de vorbitor: sisteme dependente de context şi sisteme independente de context. Sistemele dependente de context au performanţe mai bune decât cele independente de context, dar arhitectura lor este mai complexă, deoarece aceste sisteme conţin şi un sistem de recunoaştere a limbajului vorbit. Cu toate acestea se observă un interes mai mare în sistemele independente de context care recunosc un individ fără constrângeri legate de ceea ce spune acesta.

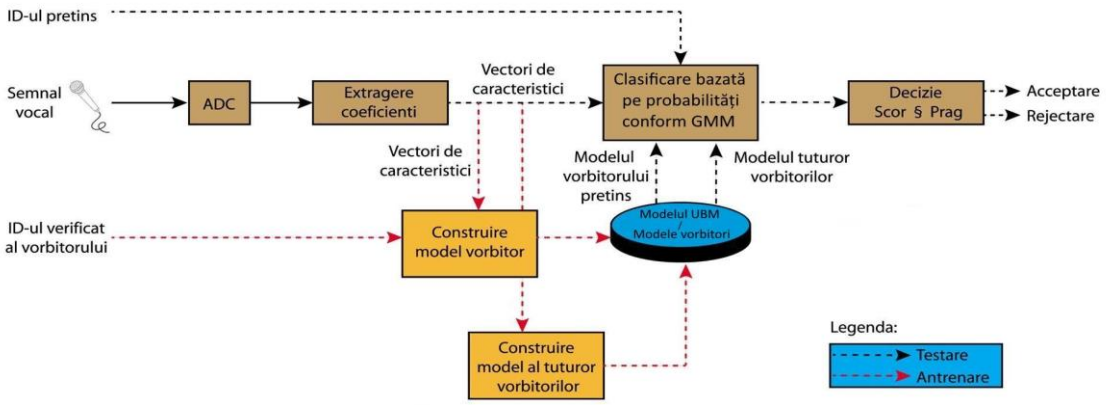

Fig. 1.3 Arhitecura unui sistem de verificare de vorbitor

Se observă din cele două tipuri de arhitecturi, prezentate în Figura 2.2 si Figura 2.3, că cele două tipuri de sisteme de recunoaștere de vorbitor au câteva puncte comune. cum ar fi extragerea caracteristicilor (despre care se vorbește mai în detaliu în secțiunea //scrie sectiunea) care se realizează în orice situație și crearea modelului specific fiecărui vorbitor. Se observă că în cazul sistemelor de verificare de vorbitor se creează în etapa de antrenare și un model pentru vorbirea umană generală, ceea ce nu este necesar pentru sistemele de identificare din moment ce ele vor compara semnalul vocal de la un individ cu toate modelele vorbitorilor existenți în sistem.

Modul de testare, după cum se observă din cele două arhitecturi, diferă semnificativ. Dacă în cazul sistemelor de identificare caracteristicile semnalului vocal al unui vorbitor sunt comparate cu fiecare model existent în baza de date, în cazul sistemelor de verificare se vor folosi două modele, deci pe baza caracteristicilor semnalului vocal al unui vorbitor se vor obține două scoruri: unul pentru modelul vorbitorului pretins și unul pentru modelul general de vorbire.

O altă concluzie ce poate fi trasă privind cele două tipuri arhitecturi este aceea că performanțele unui sistem de verificare nu scad odată cu creșterea numărului de vorbitori existenți în sistem, ceea ce nu putem afirma în cazul sistemelor de identificare.

1.3 PRINCIPIILE DE BAZĂ ALE MODELĂRII ACUSTICE

Atât sistemele de recunoaștere cât și sistemul de recunoaștere de vorbitor funcționează pe baza unor modele acustice. În cazul sistemelor de recunoaștere a limbajului vorbit modelul acustic este format dintr-un set de modele pentru foneme care se conectează în timpul decodării și formează modele pentru cuvinte, urmând ca mai apoi să formeze modele pentru succesiunui de cuvinte. În cazul sistemelor de recunoaștere de vorbitor modelul acustic al unui vorbitor se creează pe baza caracteristicilor semnalului vocal furnizat de acesta în etapa de antrenare. Voi descrie mai în detaliu procesul de creare al modelelor acustice în sectiuniile viitoare.

1.3.1 *Caracteristicile semnalului vocal*

Deoarece semalul vocal este un semnal nestaționar, deci variază permanent și își modifică caracteristicile pentru codificarea succesiunilor de foneme, analiza spectrală se va face pe segmente scurte de câteva zeci de secunde (20 - 30 ms), în care semnalul va fi considerat staționar. Semnalul vocal este transformat prin ferestruire intr-o secvență de ferestre. Din fiecare fereastră se vor extrage mai mulți parametri acustici. Pentru ferestruirea semnalului vocal se pot folosi mai multe tipuri de ferestre. În Fig. 1.4 sunt prezentate cele mai uzuale ferestre folosite pentru extragerea caracteristicilor vocale.

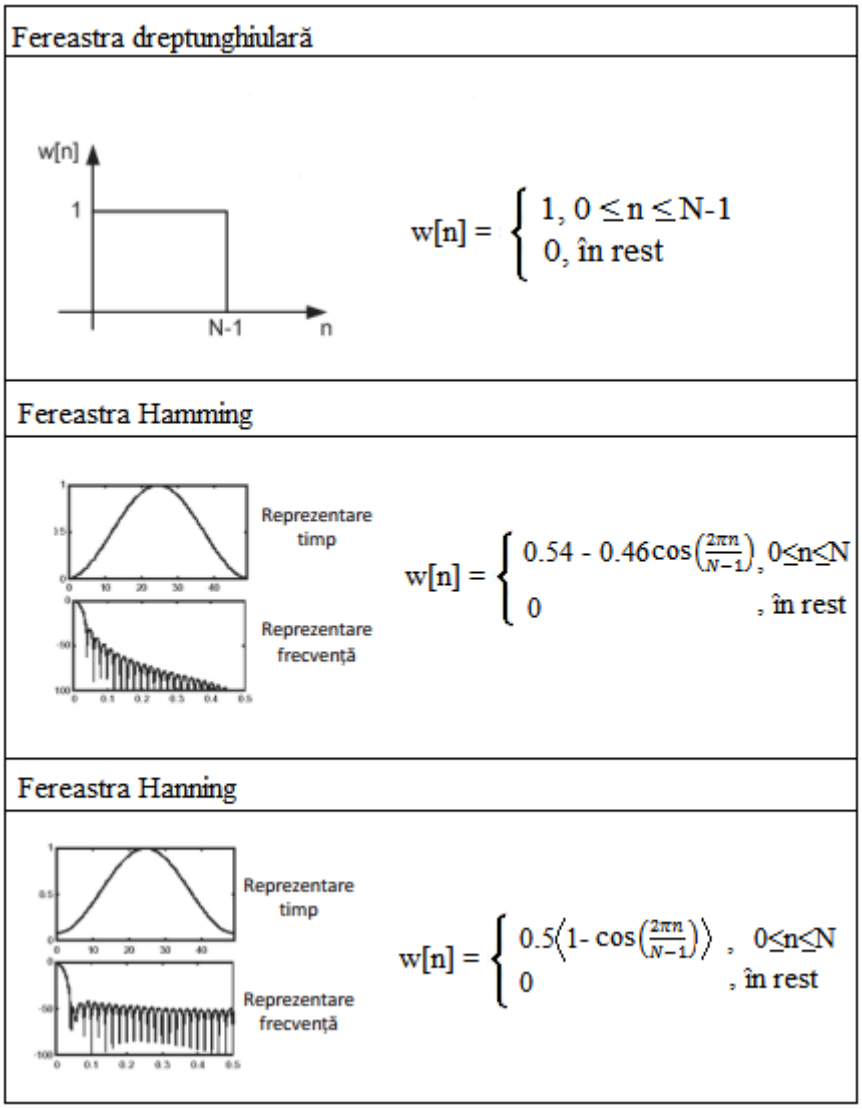

Fig. 1.4 Ferestre uzuale folosite pentru extragerea caracteristicilor

Analiza în timp a semnalului vocal se face pentru determinarea parametrilor globali, valabili pe termen scurt. Parametrii de timp scurt sunt: energia de timp scurt, numărul trecerilor prin zero, funcția de autocorelație.

Energia de timp scurt este utilizată în detecția perioadelor de liniște și pentru decizia vocalizat/nevocalizat. Decizia se ia în sensul că pentru o energie măsurată En se verifică dacă este mai mare sau mai mica decât un prag impus. În cazul în care energia este mai mare decât pragul ales se consideră că segmentul analizat este sonor, în caz contrar segmentul este considerat zgomot sau impuls.

Numărul trecerilor prin zero este un parametru important al semenalului vocal, fiind folosit împreună cu energia semnalului vocal. Acesta ajută de altfel pentru detecția vorbire-liniște, respective detecția sonor-nesonor.

O altă caracteristică importantă a semanlului vocal este funcția de autocorelație. Aceasta oferă informații despre periodicitate în cazul segmentelor vocalizate. Se obțin rezultate mai bune atunci când este folosită o fereastră dreptunghiulară. Funcția de autocorelație ajută în determinarea tonului fundamental ce reprezintă intonația vocii. Uneori funcția de

autocorelație poate conține o cantitate mare de informație, dintre care mare parte să fie irelevantă. Probleme pot apărea în cazul semnalelor zgomotoase.

1.3.2 *Parametri de voce*

Parametrii de voce utilizați atât în domeniul recunoașterii limbajului vorbit, cât și în domeniul recunoașterii de vorbitor pot fi: coeficienții de predicție liniară LPC (Linear Predictive Coefficients), coeficienții cepstrali din scara Mel MFCC (Mel Frequency Cepstral Coefficients), coeficienții perceptuali de predicție liniară PLP (Perceptual Linear Prediction), etc. Acești parametri de voce se vor folosi atât în etapa de antrenare, pe baza cărora se creează modelele acustice, cât și în etapa de recunoaștere. Parametrii de voce vor fi calculați la fiecare fereastră de timp și vor forma vectorul de obsevații.

Atât în cazul sistemului de recunoaștere de vorbitor, cât și în cazul sistemului de recunoaștere al limbajului vorbit experimentele au fost făcute folosind parametrii MFCC, de accea în cele ce urmează vor fi prezentați mai în detaliu, descriind doar în principiu parametrii LPC și PLP.

1.3.2.1 Obținerea parametrilor MFCC

Semnalul vocal este obținut prin convoluția în timp a semnalului de excitație produs de vibrația corzilor vocale și răspunsul în timp al filtrului reprezentat de traiectul vocal. Cele două semanle nu se pot separa în domeniul timp, dar se pot separa în domeniul frecvență. Pentru a separa cele două semnale se va aplica transformata Fourier, se va logaritma, urmând să se aplice transformata Fourier inversă pentru determinarea cepstrului în domeniul timp.

Pentru a determina coeficienţii cepstrali se va aplica mai întâi transformata Fourier semnalului vocal, iar spectrul rezultat urmând să fie netezit printr-un banc de filtre triunghiulare. Acestea calculează spectrul mediu în jurul frecvenței centrale a fiecărei benzi, iar fiecare filtru va ocupa o bandă mai largă în funcție de ordinea din filtru, așa cum este prezentat și în Fig. 1.5.

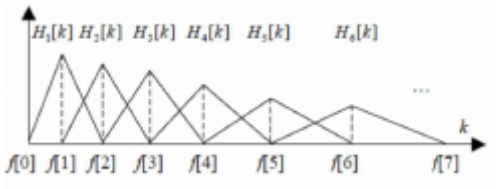

Fig. 1.5 Banc de filtre

Formula unui filtru triunghiular:

$$H_m[k] = \begin{cases} 0 & k < f[m-1] \\ \dfrac{2(k - f[m-1])}{(f[m+1] - f[m-1])(f[m] - f[m-1])} & f[m-1] \le k \le f[m] \\ \dfrac{2(f[m+1] - k)}{(f[m+1] - f[m-1])(f[m+1] - f[m])} & f[m] \le k \le f[m+1] \\ 0 & k > f[m+1] \end{cases}$$

Numărul de filtre dintr-un banc este configurabil în sistemele de recunoaştere, variind între 24 – 40 de filtre.

Se calculează energia logaritmică la ieşirea fiecărui filtru, iar apoi se vor calcula coeficienţii cepstrali prin tranformata cosinus discretă (DCT – Discrete Cosinus Transform). Transformata cosinus discretă are capacitatea de a concentra informaţia spectrală într-un număr mai mic de parametrii şi de a decolera aceste valori.

1.3.2.2 Parametri de voce LPC

Etapele pentru calculul coeficienţiilor LPC sunt prezentate în Fig. 1.6.

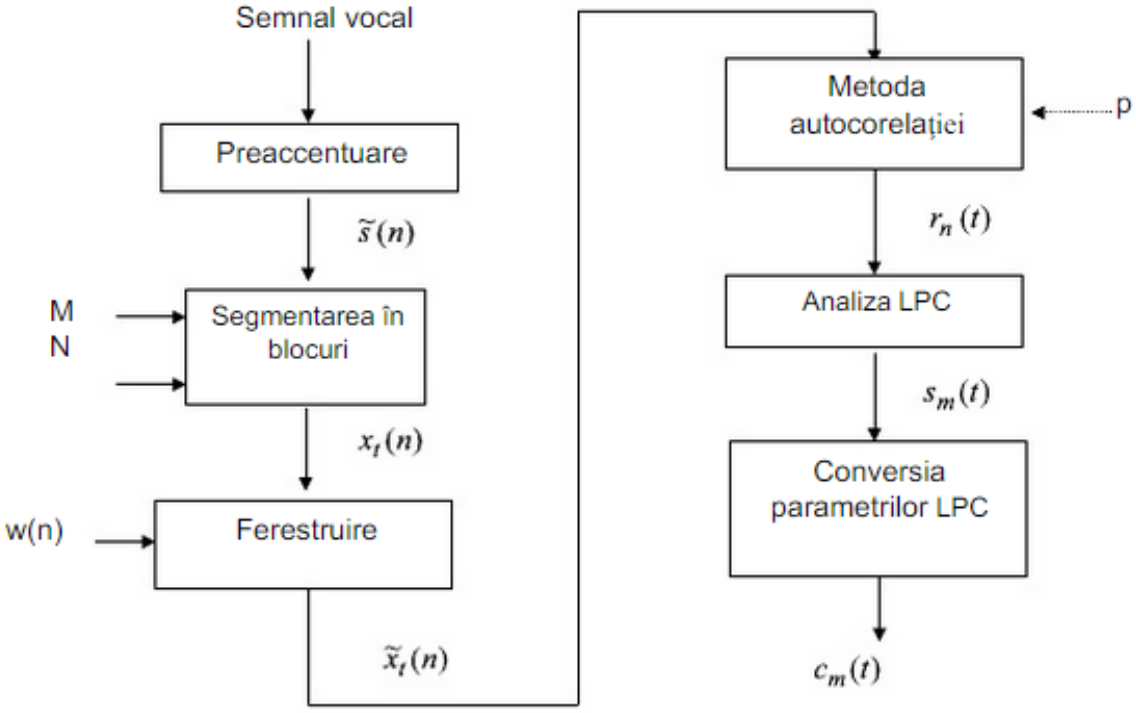

Fig. 1.6 Etapele calculului coeficienţiilor LPC

Etapa de preaccentuare se foloseşte pentru egalizarea tăriei sonore. În etapa de segmentare în blocuri, semnalul preamplificat este împărţit în cadre de N eşantioane, iar cadrele adiacente vor fi separate de M eşantioane. Dacă M ≤ N, atunci cadrele alăturate se vor suprapune, iar estimaţii spectrali LPC rezultaţi vor fi corelaţi din cadru în cadru daca M << N. În cazul acesta estimaţii spectrali vor varia foarte puţin. În cazul în care M>N nu va apărea suprapunearea între cadre, dar o parte din semnal va fi pierdut şi corelaţia între estimaţii spectrali LPC din cadre alăturate va conţine o componentă de zgomot care creşte odată cu M. Următorul pas constă în ferestruirea fiecărui cadru. Ferestruirea va minimiza discontinuităţile semnalului la începutul şi sfârşitul fiecărui cadru.

Există două metode mai cunoscute de a determina coeficienţii LPC: metoda covarianţei şi metoda corelaţiei. După aplicare uneia dintre metode de soluţionare a unui sistem se va face analiza LPC, ultima etpă fiind de conversie a coeficienţiilor LPC.

1.3.2.3 Parametri de voce PLP

Etapele calculului coeficienţiilor PLP sunt prezentate în Fig. 1.7. Semnalul vocal este iniţial supus unei analize spectrale, folosind segmente vocale de 20 ms lungime şi o fereastră de tip Hamming.

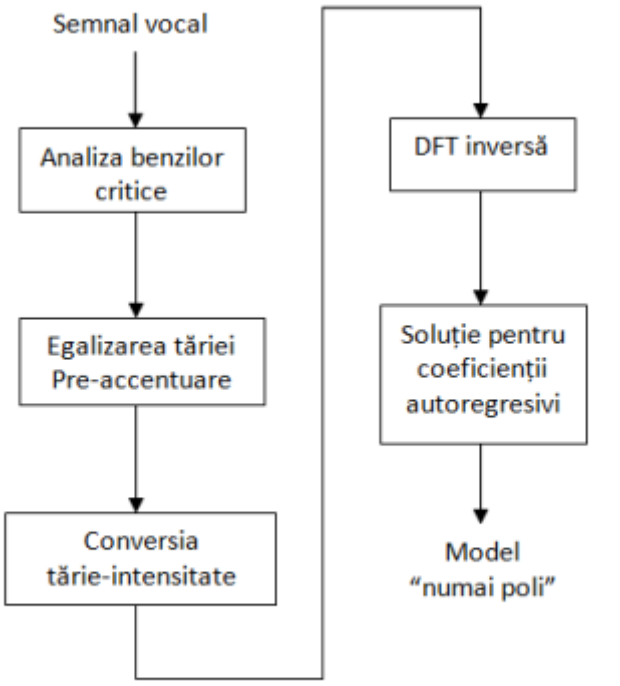

Fig. 1.7 Etapele calculului coeficienţiilor PLP

În tehnica PLP proprietăţiile semnalului auditiv sunt simulate prin diferite apoximări. Spectrul semnalului rezultat va fi apoximat cu un model numit "numai poli" autoregresiv.

1.3.3 Modelul acustic construit pe HMM

Un HMM (Hidden Markov Model) este un instrument ce reprezintă distribuţia de probabilitate a unei secvenţe de observaţii. Acesta este un automat cu stări finite, având un set de stări conectate cu arce ce reprezintă tranziţii. Secvenţa de stări este ascunsă, nu îi este direct disponibilă observatorului. Fiecare stare are ataşată o funcţie de densitate de probabilitate. Un HMM modelează foarte bine producerea vorbirii, de aceea platforma HMM este foarte răspândită în sistemele de recunoaştere a limbajului vorbit, dar şi în numeroase aplicaţii din domeniul biologiei, în compresia datelor, aplicaţii ce vizează inteligenţa artificială şi recunoaşterea formelor.

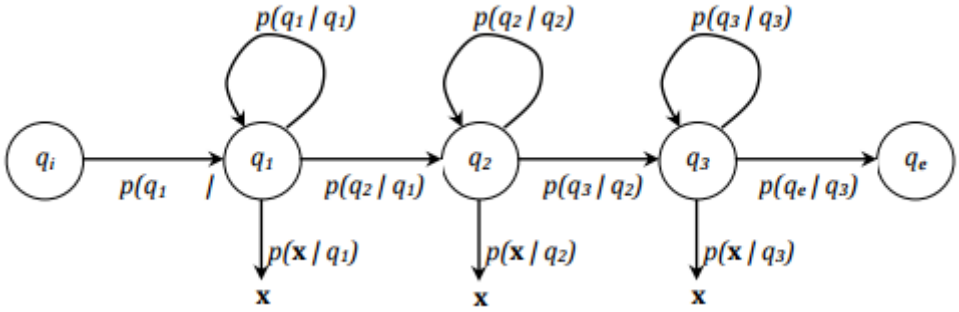

Fig. 1.8 Reprezentare detaliată unui HMM

Un HMM este caracterizat de un set de stări $Q = q_1q_2...q_N$, un set de probabilități ce reprezintă probabilitățile de tranziție între stări și un set de observații probabilistice $B = b_i(x_t) = p(x_t|q_i)$ care exprimă probabiliatea ca observația x_t să fie generată de starea i.

În sistemele de recunoaștere de vorbire tranzițiile nu sunt arbitrare, deci tranzițiile nu se pot duce în orice stare, pentru a respecta caracterul secvențial al producerii vorbirii. Folosirea acestor modele în sistemele de recunoaștere de vorbire este posibilă datorită a două presupuneri, și anume: probabilitatea tranziției următoare depinde numai de starea curentă și observațiile sunt independente, ceea ce înseamnă că probabilitatea ca o stare să genereze un vector de parametri este independentă de vectorii de parametri generați anterior.

În cazul general carcateristicile acustice care reprezintă ieșirea unui HMM pot avea o gamă largă de valori reale. Din această cauză probabilitățiile observațiilor sunt funcții discrete doar într-o abordare simplificată, dar în general acestea sunt funcții de densitate de probabilitate. În general funcția densitate de probabilitate pentru o stare q_i este o Gaussiană n dimensională, caracterizată de vectorul medie μ_i și matricea de covarianță Σ_i.

1.3.3.1 Definiția GMM

Un GMM (Gaussian Mixture Model) este o suma ponderata de Gaussiene. GMM-urile sunt folosite pentru modelarea probabilitatii de distributie a caracteristicilor (features) intr-un sistem biometric, cum ar fi caracteristicile spectrale intr-un sistem de recunoaștere de vorbire. În Fig. 1.9 este reprezentat un GMM, mai exact o suma ponderată de Gaussiene de diferite medii și dispesii.

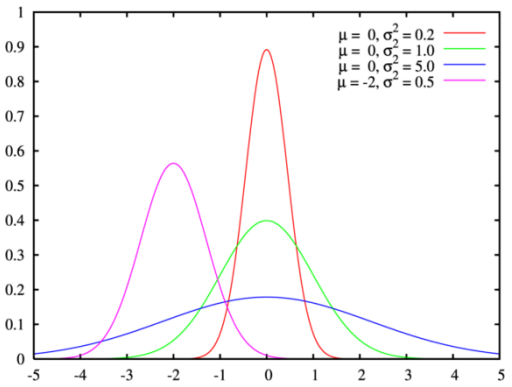

Fig. 1.9 Reprezentarea unui GMM

În sistemele de recunoaștere a limbajului vorbit, cât și în sistemele de recunoaștere de vorbitor ca instrument de clasificare, în locul GMM-urilor, se pot folosi și SVM-uri (Support Vector Machine) sau mai nou rețele neuronale (NN - Neural Networks).

În cazul sistemelor de recunoaștere de vorbitor clasificarea pe bază de GMM-uri are rezultate foarte bune. O cauză bună a rezultatelor bune este datorată algoritmului expectation-maximization (EM). Caracteristica cheie a acestui algoritm este că poate garanta convergența setului de parametri optimi în doar câteva iterații.

Cu toate acestea putem observa și câteva dezavantaje ale folosirii GMM-urilor. Un dezavantaj important este acela că pentru crearea unui model puternic este nevoie de o un set mare de antrenare. Acestă problemă poate fi însă rezolvată folosind matricea diagonală de covarianță, în favoarea celei normale. O a doua problemă ar fi ca datele necunoscute de sistem, deci care nu au fost folosite la crearea modelelor, dar care apar în setul de testare vor produce un scor mic pentru acele date, iar performanța generală a sistemului se va degrada. De exemplu, pentru un anumit vorbitor anumite caracteristici ale vocii sale nu se regăsesc în setul de antrenare, deci nu se vor găsi nici în modelul său, dar când aceste caracteristici vor aparea în setul de test vor obține un scor mic. Soluția ar putea suna simplu, setul de antrenare să fie cât mai mare și mai variat, dar în practică, în sistemele de recunoaștere de vorbitor, problema este un pic mai greu de combătut.

1.3.3.2 Definiția UBM

În cazul sistemelor de recunoaștere de vorbitor, mai exact în cazul sistemele de verificare apare noțiunea de model universal ce modelează vorbirea umană în general (UBM – Universal Background Model).

În recunoașterea de vorbitor identitatea pretinsă a unui vorbitor este verificată comparând două scoruri: scorul obținut pentru modelul vorbitorului pretins și scorul obținut pentru modelul impostorului. Modelul impostorului este de fapt un GMM care modelează vorbirea umană, este un model al tuturor vorbitorilor și este menționat ca și UBM.

Modelele statistice, cum ar fi GMM-urile, nu doar că sunt capabile să estimeze direct folosind tehnici precum algoritmul EM, dar cu cantități mici de date parametrii pot fi adaptați în continuare la date noi folosind adaptarea Maximum A-Posteriori (MAP). Astfel se poate antrena un UBM, iar mai apoi prin adaptarea acestuia la datele unui vorbitor să se creeze GMM-ul specific acelui vorbitor. Etapa de antrenare și testare folosind platforma GMM-UBM este reprezentată în Fig. 1.10.

Se observă din figură ca adaptarea MAP poate fi aplicată doar după ce algoritmul EM prodce UBM-ul de la toți cei N vorbitori.

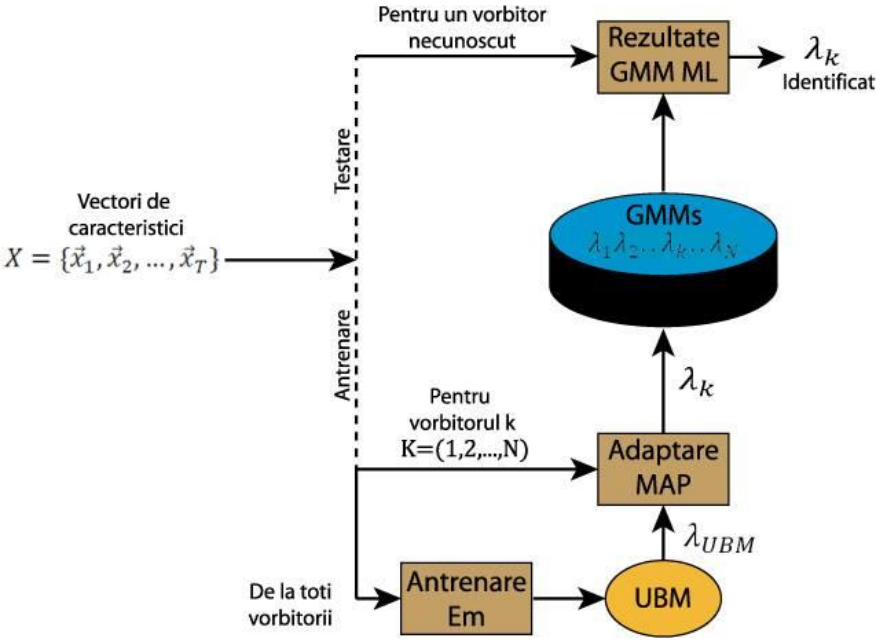

Fig. 1.10 Platforma GMM-UBM în etapa de antrenare şi testare

1.4 EVALUAREA SISTEMELOR

Evaluarea sistemelor de recunoaştere este o etapă importantă în dezvoltarea şi optimizarea acestora. În acestă etapă se determină dacă sistemul construit corespunde cerinţelor proiectantului. Această etapă nu poate fi realizată fără existenţa unei baze de date ce conţine înregistrări audio etichetate. O bază de date de evaluare pentru un sistem de recunoaştere de vorbire va avea înregistrări audio etichetate cu trascrierea acestor înregistrări, deci cu ceea ce este pronunţat în fiecare fişier audio, iar în cazul sistemelor de recunoaştere de vorbitor etichetele indică vorbitorul al cărui semnal audio se regăseşte în fiecare înregistrare.

1.4.1 *Evaluarea sistemelor de recunoaştere a limbajului vorbit*

Performanţele unui sistem de recunoaştere a limbajului vorbit sunt evaluate cu ajutorul a două măsurători standard: rata cuvintelor eronate (WER – Word Error Rate) şi rata propoziţiilor eronate (SER – Sequance Error Rate). Calculul acestor erori implică compararea secvenţei de cuvinte ce este returnată de sistem cu o secvenţă de cuvinte de referinţă, secvenţa corectă. Pot apărea trei tipuri de erori: inserţii, ştergeri şi substituţii. Rata de cuvinte eronate este calculată pe baza formulei:

$$\text{WER}[\%] = \frac{\#\text{inserţii}+\#\text{ştergeri}+\#\text{substituţii}}{\#\text{cuvinte ale propoziţiei corecte}} \times 100 \qquad (2.1)$$

Rata propoziţiilor eronate se va calcula după formula:

$$\text{SER}[\%] = \frac{\#\ propoziţii\ care\ conţin\ cel\ puţin\ un\ cuvânt\ eronat}{\#\ propoziţii\ din\ transcrierea\ de\ referinţă} \times 100 \qquad (2.2)$$

1.4.2 Evaluarea sistemelor de recunoaștere a vorbitorului

În sistemele de recunoaștere de vorbitor două măsurători mai populare sunt folosie pentru evaluarea performanțelor: rata de falsă acceptare (FAR – False Acceptance Rate) și rata de falsă rejecție (FRR – False Rejection Rate).

Rata de falsă acceptare reprezintă probabilitatea ca sistemul să accepte eronat accesul unui vorbitor neautorizat, deci a unui impostor. Rata de falsă acceptare pentru un vorbitor este dată de formula:

$$FAR[\%] = \frac{\#acceptări\ eronate\ ale\ \ unui\ vorbitor\ neautorizat}{\#incercări\ de\ autentificare\ ale\ vorbitorului\ neautorizat} \times 100 \quad (2.3)$$

Rata de falsă rejecție penru un vorbitor este calculată după formula:

$$FRR[\%] = \frac{\#rejecții\ eronate\ ale\ \ vorbitorului\ autorizat}{\#incercări\ de\ autentificare\ ale\ vorbitorului} \times 100 \quad (2.4)$$

În cazul sistemelor de recunoaștere de vorbitor, vocea unui vorbitor este comparată atât cu modelul vorbitorului pretins, cât și cu modelul general pentru vorbire, modelul tuturor vorbitorilor din sistem. Scorurile obținute vor fi comparate cu un prag, dacă scorul obținut este peste un anumit prag atunci vorbitorul este acceptat, în caz contrar acesta va fi respins. Dacă variem acest prag de decizie, FAR și FRR vor varia în direcții opuse. De exemplu, dacă creștem pragul, FAR se va micșora fiind mai puține acceptări eronate datorită pragului mai ridicat, dar FRR va crește, iar în cazul în care scădem pragul FRR se va micșora, iar FAR va crește, sistemul acceptând mai mulți impostori. Punctul tipic de alegere al pragului este atunci când FAR = FRR, și mai este numit și condiția de rată de eroare egală (ERR – Equal Error Rate). În Fig. 1.11 sunt reprezentate FAR și FRR în funcție de pregul de decizie ales.

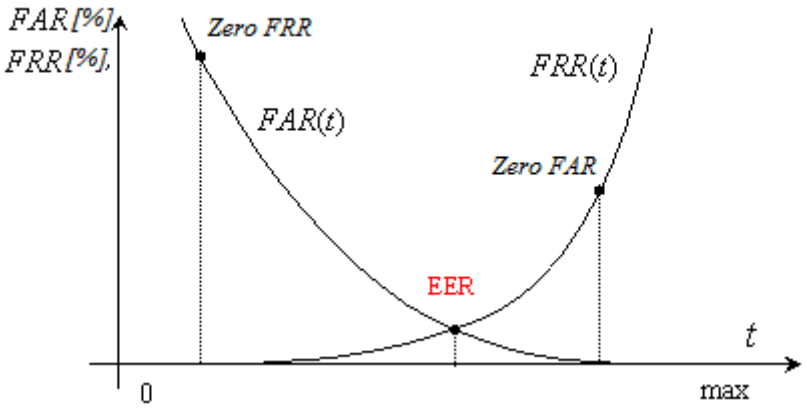

Fig. 1.11 Variația FAR și FRR în funcție de pragul de decizie ales

CAPITOLUL 2

O PRIVIRE DE ANSAMBLU ASUPRA SISTEMULUI

2.1 DESCRIERE GENERALĂ

Serviciul web de autentificare prin voce este un sistem de logare cu nivel înalt de securitate care permite accesul în cont pe baza amprentei vocale a utilizatorilor. Autentificarea utilizatorilor în conturi se va face atât pe baza clasicei parole, cât și pe baza semnalului vocal.

Acesta este format din trei module principale: un modul de recunoaștere de vorbitor, un modul de recunoaștere de cifre și un modul de autentificare (care nu este subiectul acestei lucrari) ce permite comunicarea între module. Cele trei module sunt independente și sunt expuse ca și servicii REST (Representational State Transfer). Sistemele de recunoaștere sunt sisteme robuste ce au fost evaluate atât în condiții reale de utilizare, cât și în condiții de laborator, iar rezulatele obținute ne permit să afirmăm că sistemul este un sistem de logare cu nivel înalt de securitate. Fiecare dintre aceste module are o sarcină bine determinată, iar folosite împreună oferă un serviciu sigur de autentificare. Schema bloc a serviciului este prezentată în Fig. 2.1.

Pentru a putea folosi acest serviciu și a accesa datele securizate un utilizaror trebuie să parcurgă două etape: etapa de înregistrare și etapa de autentificare. Etapa de înregistrare presupune completarea unui formular și realizarea unui set de înregistrări. Acest set de înregistrări audio vor fi trimise mai întâi la modulul de autentificare care le va trimite mai departe modulului de recunoaștere de cifre conectate, modulului de recunoaștere de vorbitor și bazei de date, unde vor fi stocate. Modulul de recunoaștere de cifre va realiza transcrierile setului de înregistrări pe care le va trimite modulului de autentificare. Modulul de recunoaștere de vorbitor pe baza setului de înregistrări va crea un model specific utizatorului. Pe baza acestui model se va face mai departe autentificarea.

În etapa de autentificare utilizatorul va trebui sa își introducă adresa de poștă electronică, parola și să înregistreze un fișier audio în care va rosti o secvență de 12 cifre generată aleator. Se vor face în paralel trei tipuri de verificări. Se va verifica dacă parola introdusă este cea corectă, dacă vorbitorul care dorește să se autentifice a rostit în fișierul audio înregistrat secvența de 12 cifre dată și dacă acestă înregistrare conține semnalul vocal al utilizatorului pretins.

Modulul de recunoaștere de vorbire va verifica dacă ceea ce este rostit în fișierul audio trimis corespunde cu secvența de cifre generată și este folosit pentru a împiedica autentificarea unui impostor pe baza unei clip audio preînregistrat care să conțină vocea utilizatorului autorizat. Aici se va accepta și o mică variabiliate, acceptând un WER nu mai mare de 8%. Acest WER acceptat diferă de la utilizator la utilizator, și este stabilit pentru fiecare utilizator în faza de autentificare pe baza setului de înregistări și a transcrierilor textuale de către modulul de autentificare (care nu este parte a acestei lucrări).

Modulul de recunoaștere de vorbitor are rolul de a decide dacă vorbitorul care încercă să se autentifice este într-adevăr utilizatorul care se pretinde sau este un impostor. Semnalul vocal este comparat atât cu modelul utilizatorului pretins, cât și cu un model de vorbire generală. Fiecare model va primi un scor, iar cel ce are scorul cel mai bun este desemnat câștigător. Astfel dacă obținem un scor mai bun pentru modelul utilizatorul pretins se va lua decizia că vorbitorul care încercă să se autentifice este chiar utilizatorul autorizat, iar în cazul contrar în care scorul este mai bun pentru modelul de vorbire generală se va considera că vorbitorul ce încercă să se autentifice este un impostor.

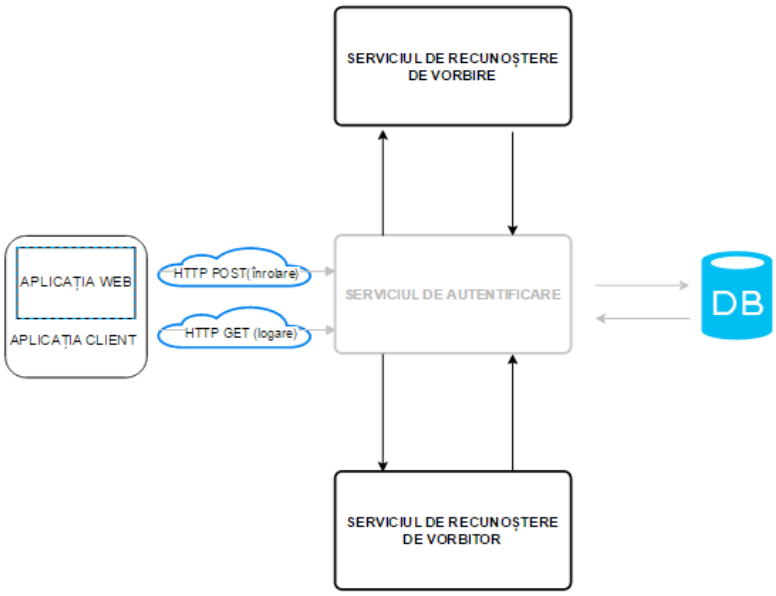

Fig. 2.1 Schema bloc a serviciului de autentificare prin voce

2.2 FUNCȚIONALITATEA SISTEMULUI

2.2.1 *Functionalitatea sistemului din perspectiva utilizatorului*

Așa cum am menționat anterior, un utilizator va parcurge două etape pentru a folosi serviciul de autentificare prin voce. Prima etapă este etapa de înrolare în sistem. În acestă etapă un utilizator va trebui să completeze formularul reprezentat în Fig. 2.2. După ce acesta

completează toate câmpurile va trebui să înregistreze și un set de clipuri audio. În fiecare clip audio utilizatorul va trebui să rostească secvența de 12 cifre afișată de aplicație.

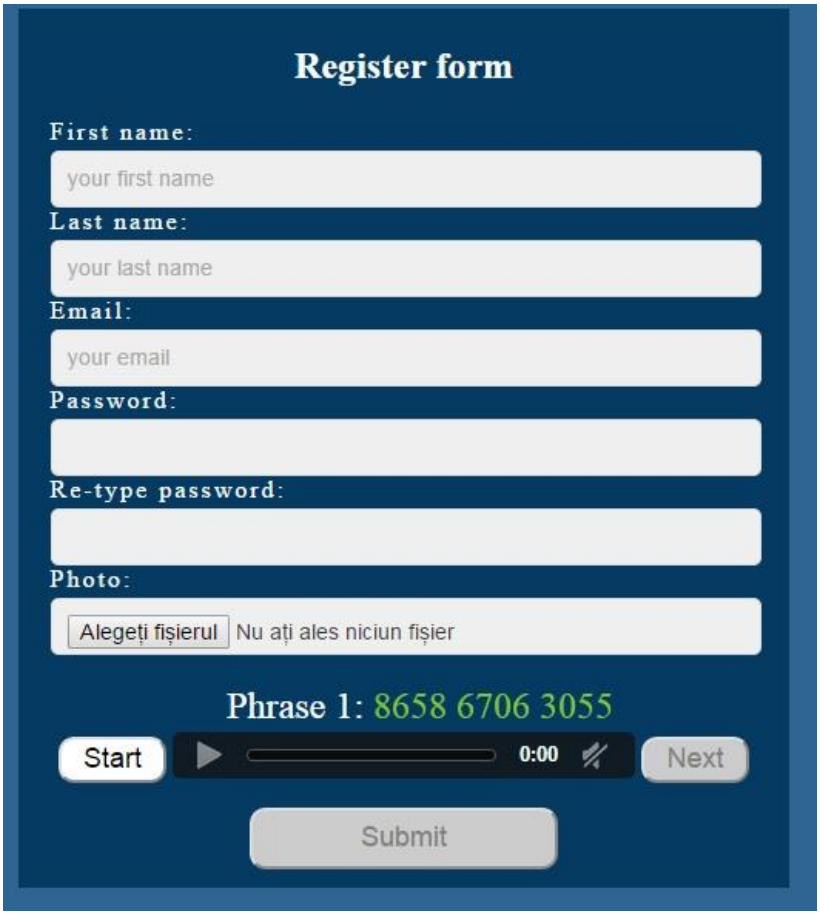

Fig. 2.2 Formular de înregistrare

Utilizatorul are posibiliatea să reasculte fiecare înregistrare audio, imediat ce acesta a realizat-o și o poate reface dacă consideră că este necesar. Nu se poate trimite formularul dacă nu s-au realizat cel puțin 10 clipuri audio sau dacă unul din câmpuri a fost omis. Utilizatorul poate să înregisteze oricât de multe clipuri audio dorește, neavând o limită superioară. Se recomandă înregistrarea a cât mai multor clipuri audio, deoarece acestea vor fi folosite în crearea modelului utilizatorului de către serviciul de recunoaștere de vorbitor. După ce utilizatorul s-a asigurat că a completat corespunzător toate câmpurile și a realizat clipurile audio îi este permis apăsarea butonului *Submit* ce va trimite formularul serviciului web de autentificare. Utilizatorul va primi un mesaj de informare asupra statusului înregistrării. Dacă înregistarea a avut loc cu succes, utilizatorul va fi redirecționat către un formular de autentificare. În caz contrar acesta este rugat să completeze încă o dată fișierul de înregistrare.

În etapa de autentificare utilizatorul va trebui să completeze formularul ilustrat în Fig. 2.3.

Fig. 2.3 Formular de autentificare

În acestă etapă utilizatorul va trebui să completeze câmpurile cerute și să înregistreze un clip audio în care va rosti secvența de cifre afisată. După completarea formularului și înregistrarea clipului audio utilizatorul poate apăsa butonul *Submit* care va trimite formularul către modulul de autentificare ce va face cele trei verificări: verificare dacă adresa oferită este o adresă existentă în baza de date și dacă parola este una validă, verificarea dacă transcriera fișierului audio corespunde cu secvența de cifre afișată, verificarea dacă vorbitorul este chiar utilizatorul care pretinde. După trimitearea formularului utilizatorul poate primi unul din mesajele: "unrecognized voice", în cazul în care vocea nu corespunde cu cea a utilizatorului pretins, "your audio file has big word error rate" dacă se obține un WER prea mare pentru transcrierea fișierului sau un mesaj de eroare în cazul în care intervin probleme interne la server. Dacă autentificare se realizează cu succes atunci utilizatorul este redirecționat către contul său care este ilustrat ca în Fig. 2.4.

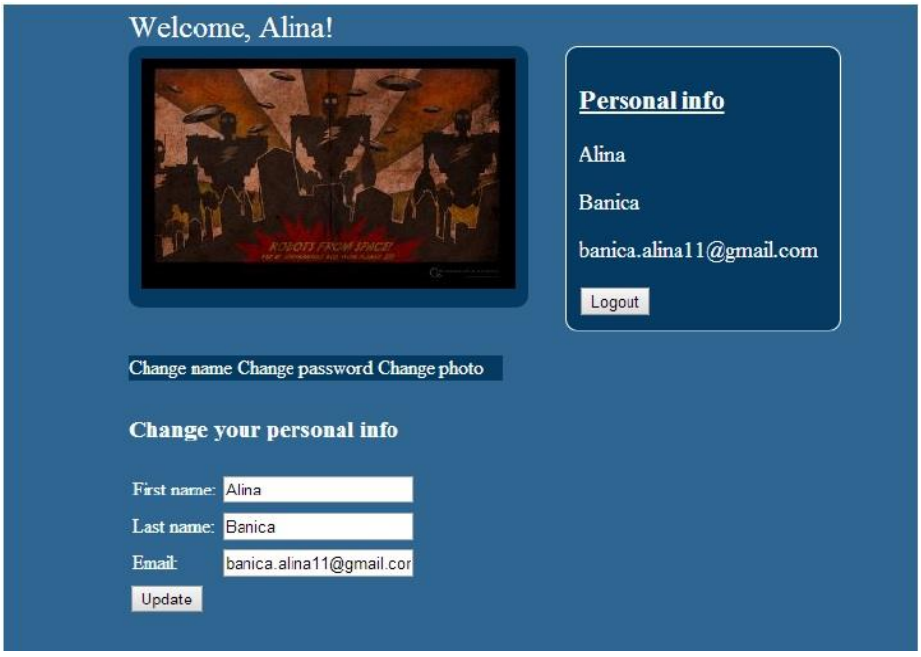

Fig. 2.4 Contul utilizatorului

Contul este realizat doar cu scop demonstrativ, neavând în momentul de față o funcționalitate practică. Utilizatorul aici își poate schimba anumite date personale, iar la sfârșit se poate deloga din cont apăsând butonul *Logout*.

Modul de realizare al formularelor și modul de comunicare între aplicația client și modulul de autentificare nu fac parte din acestă lucrare.

2.2.2 *Scenarii de utilizare*

Următoarele scenarii au rolul de a evidenția utilitatea fiecărui modul care intra în componența serviciului de autentificare pe bază de amprentă vocală.

2.2.2.1 Scenariu 1 – utilizator autorizat care dorește să se autentifice

Un utilizator autorizat care dorește să se autentifice este un un utilizator al cărui model este existent în baza de date și dorește să se autentifice pe baza acestui model. Acesta își va introduce adresa de poștă electronică, parola și va înregistra un clip audio în care va rosti secvența de cifre afișată. După apăsarea butonului *Submit* formularul va fi trimis la modulul de autentificare care va trimite mai departe clipul audio către serviciul de recunoaștere de cifre și serviciul de recunoaștere de vorbitor. Serviciul de recunoaștere de vorbitor va verifica dacă transcrierea corespunde cu secvența de cifre afișată (cu o mică variabilitate), în caz contrar acesta nu va fi lăsat să acceseze contul și va fi rugat să repete autentificarea.

Serviciul de verificare de vorbitor va folosi clipul audio înregistrat pentru a verifica daca într-adevăr utilizatorul care dorește să se autentifice este utilizatorul pretins. Astfel clipul audio va fi comparat cu modelul utilizatorului pretins și cu modelul vorbirii generale. Modelul utilizatorului pretins va obține un scor mai mare, față de modelul vorbirii generale, clipul audio aparținând într-adevăr utilizatorului.

Utilizatorul va fi redirecționat către contul său, deoarece toate verificările ar trebui să se realizeze cu succes.

2.2.2.2 Scenariu 2 – utilizator neautorizat care dorește autentificare într-un alt cont cunoscând parola

Un utilizator neautorizat este un utilizator care poate avea sau nu un model în baza de date, dar dorește autentificarea în contul unui alt utilizator. Acesta are statusul de impostor, deoarece încearcă intrarea frauduloasă în cont.

Presupunem că un utilizatorul neautorizat care cunoște adresa de poștă electronică și parola utilizatorului pretins dorște să realizeze autentificarea. Astfel el va completa câmpurile cerute ale formularului de autentificare și va înregistra clipul audio cerut în care va rosti secvența de cifre afișată. Se vor face și în cazul acesta cele trei verificări care se realizază în paralel. Verificarea parolei se încheie cu succes, parola fiind cea corectă. Clipul audio înregistrat conține într-adevăr rostirea secvenței de cifre afișate, deci transcriera va corespunde cu această secvență (cu anumită variabiliate) și verificarea realizată de serviciul de recunoaștere de cifre se va încheia și ea cu succes. Odată ajuns la serviciul de recunoaștere de vorbitor clipul audio va fi decodat. Se va compara clipul audio obținut cu modelul utilizatorului pretins

(modelul utilizatorului al cărui cont impostorul dorște să îl acceseze) și cu modelul ce modelează vorbirea general. Modelul general de vorbire va obține de data acesta un scor mai mare, deoarece caracteristicile utilizatorului neautorizat nu vor corespunde cu caracteristicile utilizatorului pretins. Astfel serviciul de recunoaștere de vorbitor va informa modulul de autentificare că cel ce dorește să se autentifice este un utilizator neautorizat care dorește autentificarea fraudulosă în cont. Utilizatorul nu va fi lăsat să acceseze contul și va primi mesajul "unrecognized voice".

2.2.2.3 Scenariu 3 – utilizator neautorizat care dorește autentificarea pe baza unui fișier preînregistrat ce conține vocea utilizatorului pretins

Prespunem că un utilizator neautorizat cunoaște parola și deține un fișier audio ce conține vocea utilizatorului al cărui cont dorește să îl acceseze. Acesta va completa și câmpurile cerute cu adresa poștei electronice și parola, iar pentru înregistrarea clipului audio va folosi clipul audio preînregistrat ce conține vocea utilizatorului pretins. Se vor face și aici cele trei verificări. Verificarea pentru parolă se va finaliza cu succes, deoarece utilizatorul cunoște acestă parolă. Clipul audio va ajunge pentru verificare și la serviciul de verificare de vorbitor. Și în cazul scenariului de față clipul audio va fi comparat cu modelul utilizatorului pretins și cu modelul general de vorbire. Scorul mai mare va fi obținut de către modelul utilizatorului pretins, deoarece clipul folosit la autentificare conține într-adevăr vocea acestuia. Verificarea din partea serviciului de recunoaștere de vorbitor se va finaliza și ea cu succes. Odată ajuns însă clipul audio la serviciul de recunoaștere se va realiza decodarea acestuia. Transcrierea obținută nu va corespunde cu secvența de cifre afișată la momentul autentificării, deoarece acestă secvență de 12 cifre este generată aleator la fiecare autentificare și este puțin probabil (există 10^{12} combinații posibile pentru această secvență) ca impostorul să îl fi surprins pe utilizatorul autorizat rostind acceeași secvență de cifre ca cea afișată (cu o anumită variabilitate).

Utilizatorul neautorizat, va trece de cele două verificări: verificarea parolei și verificarea dacă semnalul vocal aparține sau nu utilizatorului pretins, dar nu va trece de verificarea serviciului de recunoaștere de cifre care verifică dacă ceea ce se aude rostit în clipul audio corespunde cu secvența de cifre afișată în timpul autentificării. În cazul acesta impostorul nu va fi lăsat să acceseze contul și va primi mesajul "your audio file has big word error rate".

Cele trei scenarii pun în evidență funcționaliatea fiecărui serviciu din componența serviciului de autentificare prin voce. Astfel primul scenariu este cel în care un utilizator autorizat dorește autentificarea în contul său. Deoarece acesta îndeplinește toate criteriile de autentificare, trecând toate verificările cu succes, va fi lăsat să își acceseze contul. Al doilea scenariu este cel în care se evidențiază utilitatea serviciului de recunoaștere de vorbitor. Deși impostorul cunoaște parola și rostește secvența de 12 cifre, el nu va fi lăsat să își acceseze contul, deoarece semnalul vocal nu corespunde cu cel al utilizatorului pretins. Al treilea scenariu pune în evidență utilitatea serviciului de recunoaștere de cifre. Utilizatorul neautorizat care cunoaște parola și deține un fișier audio ce conține vocea utilizatorului pretins nu va fi lăsat să acceseze contul, deoarece este puțin probabil ca clipul audio înregistrat să conțină secvența de cifre afișată în formularul de autentificare.

CAPITOLUL 3

TEHNOLOGII SOFTWARE UTILIZATE

3.1 LIUM

LIUM este un utilitar folosit în tehnologia vorbirii dezvoltat de Laboratorul de Informatică al Universității Maine. Acesta este specializat în procesul cunoscut în limba engleză ca și diarization, dar nu numai, fiind folosit în aplicații precum recunoaștere de vorbitor, recunoaștere a genului, recunoaștere de vorbire, detecția porțiunilor de liniște/vorbire, indexarea documentelor multimedia. Utilitarul este format din mai multe pachete, fiind toate scrise în limbajul de programare Java.

LIUM oferă mai multe unelte ce ajută în construirea aplicațiilor din domeniul tehnologiei vorbirii, precum: unelte de extragerea a caracteristicilor semnalului vocal, unelte de segmentare a fișierelor audio, unelte pentru clusterizarea înregistrărilor, etc. Segmentarea se referă la împărţirea fişierelor audio în diverse părţi omogene şi flexibile, iar procesul de diarizare reprezintă identificarea unică a vorbitorilor într-un fişier şi poate îmbunătăți

sistemele de recunoaștere de vorbire. Acest utilitar poate fi folosit atât în sarcini de identificare, cât și în sarcini de verificare de vorbitor.

Atât în etapa de decodare, cât și în etapa de antrenare LIUM nu folosește direct fișiere audio. Pe baza fișierelor audio se generează fișiere de tip ".mfcc" și ".seg". Fișierele de tip ".mfcc" sunt fișierele ce conțin vectorii de caracteristici ale semalului vocal, mai precis coeficienții MFCC extrași din clipurile audio. Fișierele ".seg" sunt formate dintr-o singură linie și sunt de forma "[id_fișier] [canal] [start_cadru] [sfârșit_cadru] [gen] [largime_banda] [mediu] [id_vorbitor]", unde descrierea pentru fiecare element este următoarea:

- [id_fișier] reprezintă eticheta fiecărui fișier. Fișierele sunt etichetate atât cu numărul de identificare al vorbitorului, cât și cu numărul înregistrării din setul de înregistrări al respectivului vorbitor. Exemplu de etichetă pentru înregistrarea numărul 11 din setul de înregistrări ale vorbitorului cu numărul de identificare 027: 027/027_10_0011.
- [canal] – reprezintă numărul canalului de comunicație.
- [start_cadru] – reprezintă numărul ferestrei de început care de obicei este reprezentată cu 0.
- [sfârșit_cadru] – reprezintă numărul ultimei ferestre.
- [gen] – indică genul vorbitorului fiecărei înregistrări. Astfel va fi reprezentat cu M genul masculin, cu F genul feminin, iar în cazul în care nu este cunoscut genul vorbitorului se va reprezenta cu U.
- [largime_banda] – reprezintă lărgimea de bandă a fișierului audio.
- [mediu] – reprezintă mediul în care a fost înregistrat fișierul audio. Mediul poate fi unul de liniște, zogomotos sau necunoscut.
- [id_vorbitor] – reprezintă numărul de identificare al unui vorbitor din setul total de vorbitori.

În lucrarea de față s-a folosit acest utilitar pentru a construi sistemul de recunoaștere de vorbitor. Cu ajutorul lui s-au extras caraterisiticile semnalului, mai exact coeficienții MFCC, s-au antrenat modele pentru vorbitori bazate pe GMM-uri, s-a creat UBM-ul pentru toți vorbitorii, s-a evaluat sistemul, etc. Pentru extragerea coeficienților MFCC din semnalul vocal s-a folosit o fereastră de tip cosinus ridicat, iar din fiecare fereastră s-au extras 13 coeficienți. Modelele create pentru fiecare vorbitor sunt bazate pe GMM-uri. Utilitarul LIUM folosește implicit un număr de 8 mixturi, dar se poate varia numărul acesta, ajungând la 512 mixturi. Pentru antrenarea modelelor, LIUM pune la dispoziție un număr mare de resurse. Printre cele mai importante clase folosite pentru antrenare fac parte *ClusterSet* și *FeatureSet*. Resursele acestor clase vor fi folosite și în faza de testare, unde se evaluează performanța sistemului de recunoaștere de vorbitor.

Pentru crearea modelelor se parcurg mai multe etape. În prima etapă se inițializază UBM-ul, modelul general de vorbire, folosind funcția *initEM*. Se pornește de la o singură Gaussiană care este ulterior antrenată ajungând la numărul de mixturi dorit. Următoarea etapă constă în antrenarea modelului obținut în etapa anterioară prin alogoritmul EM (Estimation Maximization). Se realizează mai multe iterații ale algoritmului (între una și 20), algoritmul se va opri dacă nu s-au produs modificări semnificative între două iterații succesive. Următoarea etapă este de a initializa modelul propriu pentru fiecare vorbitor folosind funcția *initMAP*. Modelul fiecărui vorbitor se va crea pe baza modelului obținut în pasul anterior, pe baza modelului general de vorbire. Ultima etapă este cea în care se adeaptează modelul general prin metoda Maximum-a-Posteriori.

Sistemul de recunoaştere de vorbitor poate face atât identificare cât şi verificare. În cazul identificării nu mai este necesară obținerea unui scor pentru UBM, însă pentru verificarea de vorbitor vom avea nevoie de acest scor. Pentru a obține scorul pentru UBM am modificat clasa de decodare astfel încât sistemul să recunoască modelul ca fiind al unui vorbitor, identificat prin numele spk_UMB. Scorurile atât pentru toți vorbitorii sistemului cât și pentru UBM vor fi salvate într-un fișier de tip ".csv", fiind ulterior interpretate rezultatele obținute.

Pentru testarea sistemului s-au realizat numeroase experimente, variind atât numărul de fișiere ale fiecărui vorbitor folosite în antrenare, cât și numărul de mixturi ale fiecărui model.

Utilitarul LIUM se folosește, pentru a crea modele, de coeficienții MFCC, iar modele sunt reprezentate de GMM-uri. Acesta nu permite folosirea unor alte tipuri de parametri de voce, cum ar fi parametrii LPC sau PLP. În afara GMM-urilor există și alte metode de a realiza recunoașterea de vobitor, cum ar fi recunoașterea bazată pe rețele neuronale sau i-vectors.

Un alt utilitar folosit în sistemele de recunoaștere de vorbitor este utilitarul Alize care la fel ca și LIUM este un utilitar pus la dispoziție gratuit. Alize pune la dispoziție mai multe metode de recunoaștere de vorbitor, cum ar fi: recunoaștere bazată pe i-vectors, pe SVM (Support Vector Machine), GMM/UBM, etc.

3.2 PROTOCOLUL HTTP

Protocolul HTTP (Hypertext Transfer Protocol) este un protocol de comunicare folosit în aplicațiile distribuite. Acesta reprezintă un set de reguli pentru transmiterea de fișiere (text, imagini grafice, fișiere audio, fișiere video sau alte tipuri de fișiere multimedia). Protocolul HTTP funcționează ca un protocol cerere-răspuns într-un sistem de tipul client-server, așa cum se poate observa în **Error! Reference source not found.**. Un browser web poate fi clientul, iar aplicația care rulează pe un calculator să fie serverul. Acest server returnează răspunsuri către client. Răspunsul poate conține informații de stare a cererii sau poate conține obiectele cerute de către client.

Comunicarea are loc, de obicei, prin protocolul TCP/IP, dar pot fi folosite și alte tipuri de protocoale asemănatoare. Portul folosit implicit este 80, dar pot fi folosite de asemenea alte porturi.

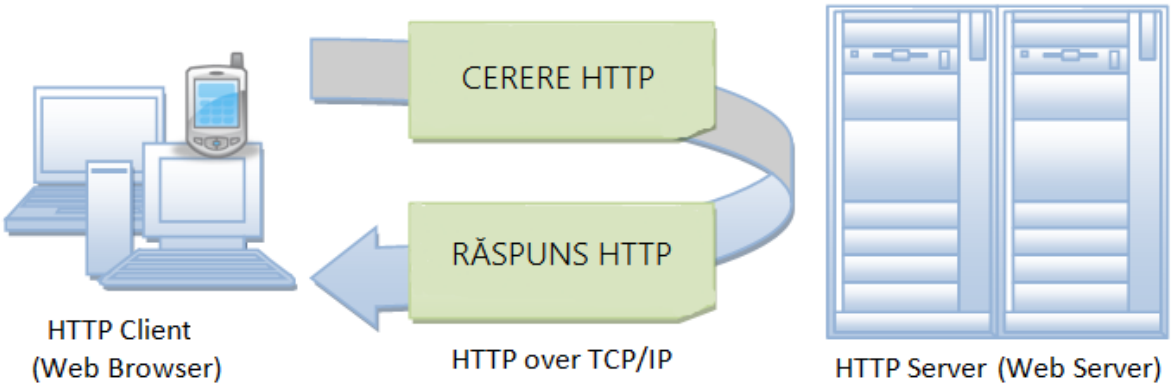

Fig. 3.1 Comunicare client-server

În cazul HTTP conexiunea dintre client și server este menținută doar pe timpul cererii curente, după acesta conexiunea este închisă. După ce clientul realizează o conexiune TCP cu serverul

şi trimite comanda de cerere, serverul trimite înapoi răspunsul său şi închide conexiunea. Prima versiune HTTP cauza multe probleme de supraîncărcare a reţelei, deoarece de fiecare dată când un fişier grafic de pe o pagină era cerut, o nouă conexiune se realiza între browse şi server. În ultima versiune HTTP 1.1 acest lucru nu se mai întâmplă, fişiere multiple se pot descărca prin aceeaşi conexiune.

Baza comunicării între clienţi şi server constă în mesajul de cerere ce este transmis printr-un URL (Uniform Resource Locators). Un URL are o structură simplă şi constă din următoarele componente:

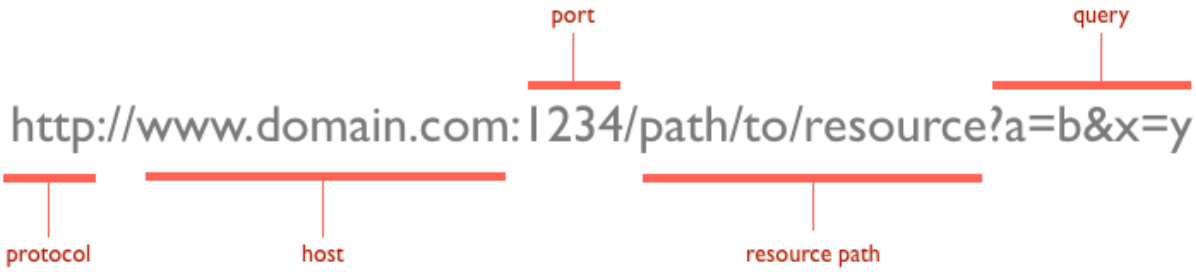

Fig. 3.2 Structură URL

Protocolul este de obicei http, dar poate fi şi https pentru comnuicaţiile sigure. Portul implicit este 80, dar poate fi ales şi un alt port explicit. Calea către resursa dorită este o cale locală a resursei aflate pe server.

Acţiunea care se realizează asupra gazdei este specificată prin verbele HTTP. Aceste verbe sunt:

- GET aduce din server orice resursă existentă. URL-ul conţine toate toată informaţia necesară serverului pentru a localiza şi a returna resursa.

- POST creează o nouă resursă. Când se foloseşte POST trebuie să se specifice datele perntru noua resursă.

- PUT updatează o resursă existentă.

- DELETE şterge o resursă existentă pe server.

Aceste patru verbe sunt cele mai folosite. Uzual PUT şi DELETE sunt considerate verisiuni specializate ale lui POST. Următoarele sunt şi ele verbe HTTP, dar cu un grad de utilizare mai scăzut:

- HEAD este similar cu GET, dar nu conţine mesajul conţinutului (body message). Este folosit pentru a recupera header-urile serverului pentru o anumită resursă, în general pentru a verifica dacă resursa s-a modificat.

- TRACE este folosit pentru a recupera calea pe care o cerere a parcurs-o de la client la server şi înapoi.

- OPTIONS este folosit pentru a obţine capacităţiile serverului. Pe partea de client poate fi folosit pentru a modifica o cerere astfel încât să poată fi deservită de server.

Cu ajutorul URL-urilor şi al verbelor un client poate iniţializa cereri către server. Serverul va răspunde cu anumite coduri de stare şi mesaje utile. Aceste coduri de stare sunt importante, deoarece clientul pe baza acestor coduri va interpreta răspunsul serverului. Protocolul HTTP specifică anumite coduri pentru diferite tipuri de răspuns:

- 1xx este clasa codurilor de informare.

- 2xx este clasa codurilor pentru cererile care s-au finalizat cu succes. Cel mai comun cod este codul 200 care ne informează că cererea a fost procesată cu succes şi totul s-a desfăşurat cum ar fi trebuit. Alte tipuri de coduri, mai puţin folosite sunt 202, 204, 205, 206. Codul de stare 202 indică faptul că o cerere a fost acceptată, dar nu conţine neapărat şi resursa în răspuns. Acest cod este util în comunicaţiile asincrone. Codul 204 este folosit pentru a informa clientul că răspunsul nu are conţinut, iar codul de stare 206 indică faptul că răspunsul are un conţinut parţial.

- 3xx este clasa codurilor pentru redirecţionare. Această clasă presupune ca clientul să ia acţiuni suplimentare, cea mai comună fiind ca acesta să folosească un alt URL pentru a putea accesa resursa. Codul 301 informează clientul că resursa pe care doreşte să o acceseze este localizată la un nou URL, acesta fiind mutată definitiv. Codul 303 infomează faptul că resursa se află provizoriu la un alt URL. Codul de stare 304 indică faptul că resursa nu s-a modificat.

- 4xx este clasa erorilor provenite de la client. Aceste coduri sunt folosite atunci când serverul consideră că eroarea survine de la client, fie acesta doreşte să acceseze o resursă inexistentă, fie realizează o cerere greşită. Cel mai cunoscut cod al acestei clase este codul 404 care indică faptul că resursa nu este disponibilă sau nu există pe server. Printre alte coduri ale acestei clase mai întalnim: codul 401 care informează că cererea necesită autorizaţie, codul 403 informează clientul că serverul a interzis accesul către resursă, codul 409 informează că se produce un conflict, serverul nu reuşeşte să deservească cererea, deoarece clientul vrea să modifice o resursă mai nouă decât ştie clientul.

- 5xx este clasa erorilor provenite de la server. Codurile acestei clase sunt folosite pentru a indica o eroare provenită de la server în timpul procesării cererii. Cel mai cunoscut cod al acestei clase este codul 500 care indică o problemă internă produsă la server. Alte coduri din acestă clasă sunt: 501 care informează că nu suportă încă cererea, codul 503 care informează că serviciul nu este disponibil, acest lucru se poate întâmpla dacă un sistem al serverului este inactiv sau dacă serverul este supraîncărcat, de cele mai multe ori serverul nu va răspunde, iar cererea va expira.

În cazul serviciului de autentificare prin voce aceste coduri de informare au fost prinse în excepţii astfel încât utilizatorul să cunoscă statusul cererii lor.

Clientul adresează cereri modulului de autentificare, iar mai departe acest modul va adresa cereri atât serviciului de recunoaştere de cifre conectate, cât şi serviciului de recunoaştere de vorbitor. În etapa de înrolare în sistem clientul va trimite formularul de înscriere împreună cu setul de înregistrări printr-o cerere de tip HTTP POST modulului de autentificare. Modulul de autentificare va trimite la rândul lui cereri HTTP POST modulului de recunoaştere de cifre şi modulului de recunoaştere de vorbitor.

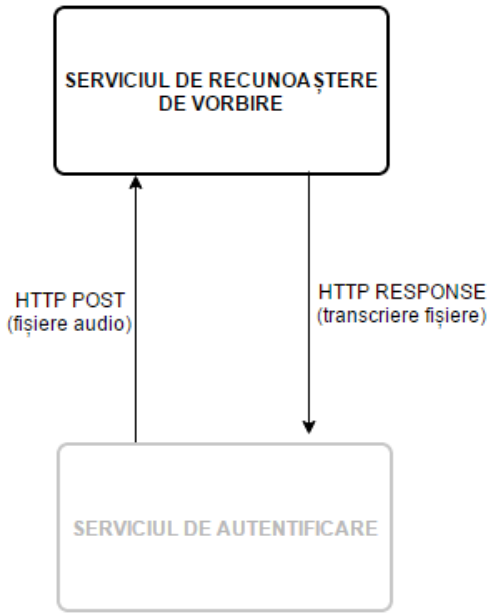

Fig. 3.3 Comunicarea între Serviciul de autentificare și Serviciul de recunoaștere de cifre in etapa de înrolare

Serviciul de autentificare trimite printr-o cerere HTTP POST fișierele audio înregistrate de utilizator în etapa de înrolare în sistem Serviciului de recunoaștere de cifre. Serviciul de recunoaștere va deservi cererea și va trimite printr-un HTTP RESPONSE transcrierea fișierelor audio. Comunicarea între Serviciul de autentificare și Serviciul de recunoaștere de cifre în etapa de înrolare este reprezentată în **Error! Reference source not found.** Desigur, există și posibilitatea ca Serviciul de recunoaștere de vorbire să nu fie activ la momentul trimiterii cererii, iar în cazul acesta se va trimite mesajul "*Connection to Trancriber failed*!" tot print-un HTTP RESPONSE.

Tot în etapa de înrolare în sistem a unui utilizator Serviciul de autentificare va trimite o cerere HTTP POST către Serviciul de recunoaștere de vorbitor. Această cerere cuprinde atât fișierele audio înregistrate de utilizator, cât și un număr de identificare specifică fiecărui utilizator (user id) și un obiect de tip String. Rolul obiectului de tip String are rolul de a informa Serviciul de recunoaștere de vorbire asupra etapei de lucru, mai exact precizează dacă utilizatorul se află în etapa de înrolare sau în etapa de autentificare. În cazul în care obiectul de timp String este reprezentat de cuvântul "training" atunci Serviciul de recunoaștere de vorbitor va crea pentru utilizator un model, iar dacă acesta este reprezentat de cuvantul "decoding" va face decodarea pentru fișierul audio primit de la Serviciul de autentificare.

Serviciul de recunoaștere de vorbitor va trimite un răspuns de tip HTTP RESPONSE Serviciului de autentificare reprezentat printr-un cod de informare. Astfel dacă totul decurge normal, deci utilizatorului i se creează un model, acesta va trimite codul 200 și mesajul "true". În cazul în care apar probleme la acest modul va trimite un cod de eroare pe care Serviciul de Autentificare îl va propaga mai departe utilizatorului pentru a fi informat cu privire la problema aparuta. Utilizatorul, în caz de o eroare provenită de la Serviciul de recunoaștere de vorbitor va primi mesajul "training process failed". Comunicarea între Serviciul de recunoaștere de vorbitor și Serviciul de autentificare în etapa de înrolare în sistem este reprezentată în **Error! Reference source not found.**.

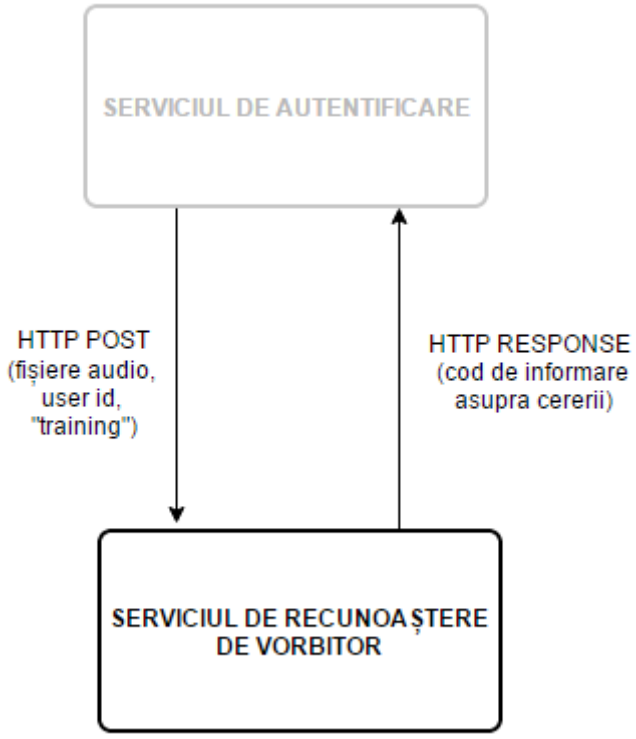

Fig. 3.4 Comunicarea între Serviciul de recunoaștere de vorbitor și Serviciul de autentificare în etapa de înrolare

În cazul autentificării unui utilizator în cont un singur fișier audio este trimis Serviciului de autentificare pe care îl va trimite mai departe prin cereri HTTP POST Serviciului de recunoaștere de vorbire și Serviciului de recunoaștere de vorbitor. Răspunsul în cazul acesta din partea Serviciului de recunoaștere de vorbire este unul de tip HTTP RESPONSE, reprezentat de transcrierea acelui fișier audio. Serviciul de recunoaștere de vorbitor va trimite ca răspuns mesajul "true" dacă scorul pentru modelul utilizatorului a fost mai mare decât modelul UBM, iar dacă scorul pentru modelul UBM-ului este mai mare se va trimite mesajul "false". Mesajele sunt transmise tot prin protocolul HTTP, fiind de tipul HTTP RESPONSE.

3.3 SERVICIUL REST

REST (Representational State Transfer) definește un set de principii arhitecturale pentru proiectarea serviciilor Web. În arhitectura REST datele și funncționalitățile sunt considerate resurse și sunt accesate folosind URI-uri (Uniform Resource Identifiers).

Un concept important care ţine de arhitectura REST este existenţa resurselor. În mod uzual o resursă este o entitate care poate fi stocată într-un computer şi poate fi reprezentată printr-un sir de biți: un document, un rând dintr-o bază de date, sau rezultatul rulării unui algoritm. Fiecare resursă are atașat un identificator global (un URI). Pentru a manipula resursele, componentele rețelei, clienții şi serverele, comunică printr-o interfață standardizată (de exemplu HTTP) si schimbă reprezentări ale resurselor (documentele care transporta informațiile).

Orice resursă trebuie să aibă cel puţin un URI ce reprezintă numele şi adresa resursei. Dacă o informație nu are un URI, nu este o resursă şi nu putem spune ca se află pe Web.

REST este un stil de arhitecura creat pentru aplicațiile conectate printr-o rețea. Ideea este că în loc să se folosească mecanisme complexe precum CORBA, RPC sau SOAP pentru a face conexiunea între mașini, simplul HTTP este folosit pentru a face apeluri între mașini. In multe feluri, chiar si World Wide Web bazat pe HTTP poate fi vazut ca o arhitecura bazată pe REST.

Această arhitectură este o arhitecura simplă, dar oferă toate functionalitățile pe care le poate oferi un serviciu Web.

Serviciul REST are urmatoarele avantaje:

- este independent de platformă. Astfel nu contează dacă serverul este Unix, clientul este Mac sau orice altceva, ceea ce reprezintă un avantaj major.
- este independent de limbajul de programare. Astfel un server Java poate comunica cu un client C# sau orice altceva.
- poate fi folosit in prezența zidurilor de protecție (firewalls).
- oferă o bună infrastructură pentru metoda HTTP GET. Acest lucru poate îmbunătați performanțele dacă resursa returnată de server nu este alterată frecvent și dacă aceasta nu are o natură dinamică.
- ușor de integrat în site-urile web deja existente. Acest lucru este foarte util, deoarece dezvoltatorii nu trebuie să scrie totul de la început.
- simplu de implementat în comparație cu alte tipuri de arhitecturi (ex: SOAP, CORBA).

Serviciul REST prezintă și dezavantaje. Un dezavantaj al acestui serviciu este folosirea de cookie-uri. Un cookie reprezintă o mică bucată de text stocată pe calculatorul unui utilizator sub formă de pereche nume-valoare care este folosită de site-urile web pentru a păstra evidența utilizatorilor ca de exemplu de a păstra informații legate de numele utilizatorului, parolă, preferințe, etc. Aceste cookie-uri prezintă anumite avantaje, dar și numeroase dezavantaje.

Un dezavantaj major al acestor cookie-uri este că acestea nu sunt sigure. Un cookie reprezintă o mică bucată de text stocată pe calculatorul unui utilizator sub formă de pereche nume-valoare care este folosită de site-urile web pentru a păstra evidența utilizatorilor. Un dezavantaj major al acestor cookie-uri este că acestea nu sunt sigure. Cookie-urile sunt păstrate în text clar și pot păstra un risc de securitate din moment ce oricine ar putea deschide și manipula un cookie. Poți cripta sau decripta manual un astfel de cookie, dar acest lucru necesită cod scris în plus și poate afecta performanțele aplicației datorită timpului necesar criptării și decriptării.

Toate modulele Serviciului Web de autentificare pe bază de voce sunt expuse ca și servicii REST.

Serviciul de autentificare pe bază de voce oferă două servicii: serviciul de creare a modelelor sau antrenare și serviciul de decodare a fișierelor audio. Serviciul de antrenare este o resursă a Serviciului de autentificare prin voce și poate fi accesat prin următorul URL: http://dev.speed.pub.ro:10082/SpeakerVerificationServer/Rest/server/training.

În tehnologia REST se folosește adnotația @Path pentru a putea crea URL-ul către o resursă. Astfel construirea URL-ului pentru resursă se va face treptat: se va folosi adnotația @Path înainte de declararea clasei, apoi se va folosi înainte de declararea metodei ce se dorește a fi

expusă ca și resursă REST. În cazul nostru clasei SpeakerVerification (cea care conține cele două resurse/metode) i se atribuie numele „server”, astfel URL: http://dev.speed.pub.ro:10082/SpeakerVerificationServer/Rest/server/ este calea către clasă. Ficare resursă are o astfel de adnotație și are atribuit un URL, în caz contrar aceasta nu este considerată resursă pentru că nu poate fi accesată.

Mai jos este reprezentat antetul metodei “training”, cea responsabilă de crearea modelelor pentru utilizatori:

```
@POST
@Path("/training")
@Consumes(MediaType.MULTIPART_FORM_DATA)
public Response training(@FormDataParam("id") String id,
            @FormDataParam("blob") List<FormDataBodyPart> blob1)
throws SecurityException, IOException{}
```

Adnotația @POST indică faptul că cererea pentru acestă resursă trebuie să fie de tip HTTP POST. Adnotația @Path(“/training”) va construi URL-ul către resursă.

Adnotația `@Consumes(MediaType.MULTIPART_FORM_DATA)` indică faptul că resursa așteptă să primească un formular de tip Multipart. Așa cum se observă se vor extrage din acest formular id-ul utilzatorului și fișierele audio. Această metodă este apelată doar în etapa de înregistrare în sistem.

Serviciul de decodare a fișierelor audio este și el o resursă a Serviciului de recunoaștere de vorbitor. Acesta este o metodă aflată în clasa Speaker Verification, la fel ca și metoda de antrenare, și poate fi accesat prin următorul URL: http://dev.speed.pub.ro:10082/SpeakerVerificationServer/Rest/server/decoding.

Metoda “decoding” este cea care este accesată de fiecare dată când un utilizator încercă autentificarea în cont. Mai jos este reprezentat antetul acestei metode:

```
@POST
@Path("/decoding")
@Consumes(MediaType.MULTIPART_FORM_DATA)
public Response decoding(@FormDataParam("id") String id,
      @FormDataParam("blob") List<FormDataBodyPart> blob1)
throws Exception{ }
```

Serviciul de recunoaștere de cifre are expusă o singură metodă ca și serviciu REST, fiind cea care va face decodarea atât în etapa de înrolare în sistem a unui utilizator, cât și la fiecare autentificare a unui utilizator. La fel ca în cazul anterior și aceasta este considerată resursă pentru că are atribuit un URL. Acestă metodă poate fi accesată prin următorul URL : http://dev.speed.pub.ro:10082/SpeakerVerificationServer/Rest/transcriber/decode

Mai jos este reprezentat antetul acestei metode:

```
@POST
@Path("/decode")
@Consumes(MediaType.MULTIPART_FORM_DATA)
public Response primeste(@FormDataParam("audioFile") List<FormDataBodyPart>
blob)
throws IOException, UnsupportedAudioFileException, FileNotFoundException {}
```

CAPITOLUL 4

SERVICIUL DE RECUNOAȘTERE DE CIFRE CONECTATE

4.1 ARHITECTURA SISTEMULUI DE RECUNOAȘTERE DE CIFRE

Serviciul de recunoaștere de cifre are la bază un sistem de recunoaștere de vorbire independent de vorbitor. Arhitectura sistemului este prezentată în Fig. 4.1.

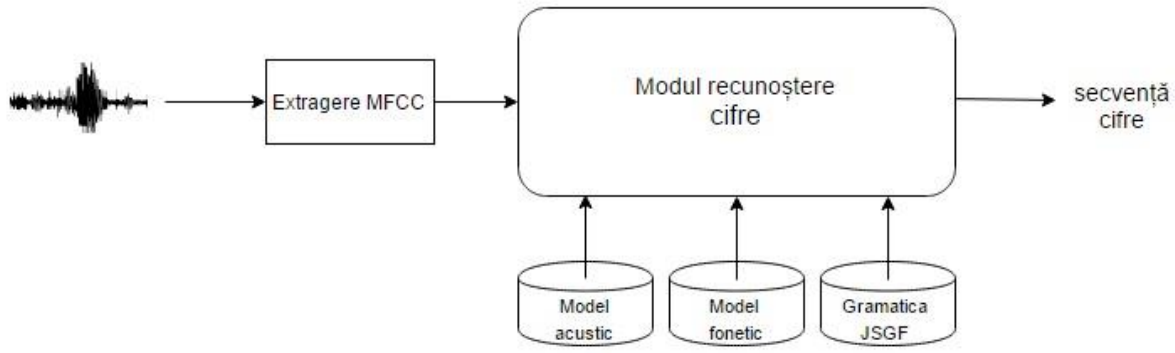

Fig. 4.1 Arhitectura sistemului de recunoaștere de cifre

Sarcina de recunoaștere a acestui sistem este de a transcrie fișierele audio ce conțin cifre. Lucrul acesta presupune că vorbitorul poate pronunța doar 10 cuvinte: zero, unu, doi, trei, patru, cinci, șase, șapte, opt, nouă, în orice ordine și de câte ori dorește.

Modelul acustic a fost creat folosind baza de date *rodigits*, o bază de date ce conține peste 90 de vorbitori diferiți, având fiecare înregistrate 100 de fișiere audio, înregistrate în condiții de laborator cu ajutorul unei aplicații care impunea anumite restricții de înregistrare. În fiecare

fişier audio este rostită o secvenţă de 12 cifre. Fiecare înregistrare audio are şi o transcriere textuală în baza de date. Acest model a fost antrenat folosind 50 de vorbitori, 80 de fişiere audio de la fiecare, 100 de senone şi o mixtură de 128 de Gaussiene. Rezultatele obţinute cu acest model vor fi discutate în secţiunea viitoare de rezultate.

Modelul fonetic se creează pe baza unui dicţionar fonetic. Dicţionarul fonetic folosit în cadrul sistemului de recunoaştere de cifre este reprezentat în Tabelul 4.1.

Modelul de limbă în cazul sistemului de recunoaştere de cifre nu este un model statistic. Succesiunile valide de cuvinte si probabilităţile lor de apariţie sunt specificate direct, folosind un model de limbă de tip gramatică cu stări finite.

Cuvânt	Scriere fonetică
zero	z_zero e_zero r_zero o_zero
unu	unu u_unu1 n_unu u_unu2
doi	doi d_doi o_doi i3_doi
trei	trei t_trei r_trei e_trei i3_trei
patru	patru p_patru a_patru t_patru r_patru u_patru
cinci	cinci k1_cinci1 i_cinci1 n_cinci k1_cinci2 i_cinci2
şase	sase s1_şase a_şase s_şase e_şase
şapte	sapte s1_şapte a_şapte p_şapte t_şapte e_şapte
opt	opt o_opt p_opt t_opt
nouă	noua n_nouă o1_nouă w_nouă a1_nouă

Tabelul 4.1 Dicţionarul fonetic

O gramatică cu stări finite este un model de tip graf în care nodurile reprezintă cuvinte ale vocabularului sarcinii de recunoaştere, iar tranziţiile între cuvinte sunt reprezentate de arcele grafului.

În cazul sistemului nostru de recunoaştere de cifre un nod în gramatica cu stări finite este reprezentată de o cifră. Aceste noduri se vor lega la nodurile de intrare şi de ieşire notate cu N (de la null, deoarece trecerea prin aceste noduri nu corespunde cu niciun cuvânt). Legarea fiecărui cuvânt de intare, respectiv de ieşire se face utilizând arce direcţionale, de la nodul de intrare către fiecare cuvânt şi de la fiecare cuvânt către nodul de ieşire. Tranziţia de la nodul de ieşire spre nodul de intrare se adaugă pentru a putea recunoaşte o secvenţă de cifre, fără de care s-ar fi putut realiza recunoaşterea doar unei singure cifre.

Gramatica cu stări finite a sarcinii noastre de recunoaştere este reprezentată în Fig. 5.4.2. Se observă că numai 10 din nodurile grafului sunt nodurile ce reprezintă cuvintele ce pot fi recunoscute, celelalte noduri (notate cu N) fiind noduri de infrastructură.

Implementarea gramaticii s-a făcut folosind formatul Java Speech Gramar (JSGF). O stfel de gramatică este definită într-un fişier. Mai jos este reprezentat conţinutul fişierului nostru de gramatică:

#JSGF V1.0;

grammar rodigits;

public <numbers> = (zero | unu |doi | trei | patru | cinci |sase| sapte | opt |noua)*;

Așa cum se observă definția gramaticii este formată din două părți: antetul gramaticii și corpul gramaticii. Prima linie a fișierului reprezintă un identificator care indică faptul că fișierul este o definiție JSGF și specifică versiunea formatului folosit. Următoarea linie din fișier reprezintă numele gramaticii, în cazul de față gramatica poartă numele de "rodigits".

Ultima linie reprezintă corpul gramaticii ce definește reguli. Așa cum se observă regula este definită ca un set de expansiuni alternative separate prin caracterul bară verticală și spații. Ceea ce impune regula este că vorbitorul poate să pronunțe cuvântul unul sau cuvântul doi, sau cuvântul trei, etc. Caracterul asterisc de la sfârșit indică faptul că acea expansiune poate să fie pronunțată de zero sau de mai multe ori.

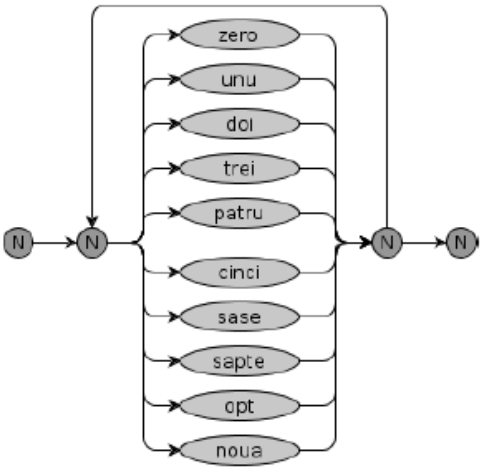

Fig. 5.4.2 Reprezentarea grafică a gramaticii cu stări finite

4.2 IMPLEMENTAREA SERVICIULUI DE RECUNOAȘTERE DE CIFRE

Serviciul de recunoaștere de cifre are la bază un sistem de recunoaștere de cifre pentru limba română a cărui funcționalități sunt expuse ca și servicii REST. Modulul care oferă servicii de recunoaștere de cifre a fost implementat în Java, în mediul de dezvoltare Eclipse și este un proiect de tipul Dynamic Web ce rulează pe un server Tomcat. Proiectul este alcătuit din mai multe pachete dintre care cele mai importante: pachetul "models", în care avem modelul acustic, modelul fonetic și gramatica cu stări finite folosită, pachetul "speech2text" ce conține clasele principale ale proiectului și pachetul "WebContent" ce conține librăriile folosite pentru ca proiectul să fie expus ca și serviciu REST.

4.2.1 *Clasa Transcriber*

Clasa *Transcriber* conține două metode importante: metoda *config()* și metode *decode()*. Metoda *config()* va fi cea care va configura sistemul de recunoaștere și este o metodă de tip void. Configurarea presupune încărcarea modelului acustic, dicționarului și a gramaticii. Se

creează un obiect denumit *recognizer*, de tipul StreamSpeechRecognizer pus la dispoziție de utilitarul Sphinx folosind configurările anterioare. Acest obiect este cel ce se ocupă de recunoașterea propiu-zisă.

Apelând metoda *decode()* se va obține transcrierea fișierelor. În acestă metodă este folosit obiectul de tip StreamSpeechRecognizer configurat anterior. Se va apela funcția getHypothesis() pusă la dispoziție de utilitarul Sphinx pentru a obține transcriera. După obținera transcrierii se va opri sarcina de recunoaștere a obiectului *recognizer* prin apelul funcției stopRecognition().

4.2.2 *Clasa GetAudioFiles*

Clasa *GetAudioFiles* din pachetul "speech2text" este singura clasă a proiectului care este expusă ca și serviciu REST. Această clasă este accesată și în etapa de înrolare a unui utilizator, cât și în etapa de autentificare. În oricare dintre aceste etape funcția *primeste* de tip *Response* extrage din formularul primit fișierele audio.

Pentru decodarea fișierulor audio se creează un obiect de tip *Transcriber*. Se apelează funcția *config()* a obiectului pentru a realiza configurările sistemului de recunoaștere, iar apoi se apelează funcția *decode()* pentru a obține transcrierea fișierului audio.

4.3 REZULTATE

Pentru testarea sistemului de recunoaștere de vorbitor s-a folosit baza de date *rodigits* și s-au realizat mai multe experimente.

4.3.1 *Experiment 1*

Experimentul 1 a presupus crearea unui model dependent de vorbitor antrenat pe baza a 80 de fișiere audio ale unui singur vorbitor existent în baza de date. Modelul a fost antrenat folosind 100, respectiv 200 de senone, fiecare senone folosind câte 1, 2, 4 respectiv 8 densități Gaussiene de probabilitate pentru a modela parametrii acustici de ieșire. Pentru evaluarea modelelor s-au folosit restul de 20 de fișiere audio ale aceluiași vorbitor. Rezultatele pentru WER obținute pentru modelele antrenate cu 100 și 200 de senone cu 1, 2, 4, 8 densități Gaussiene cu modele fonemice dependente de context sunt reprezentate în **Error! Reference source not found.**.

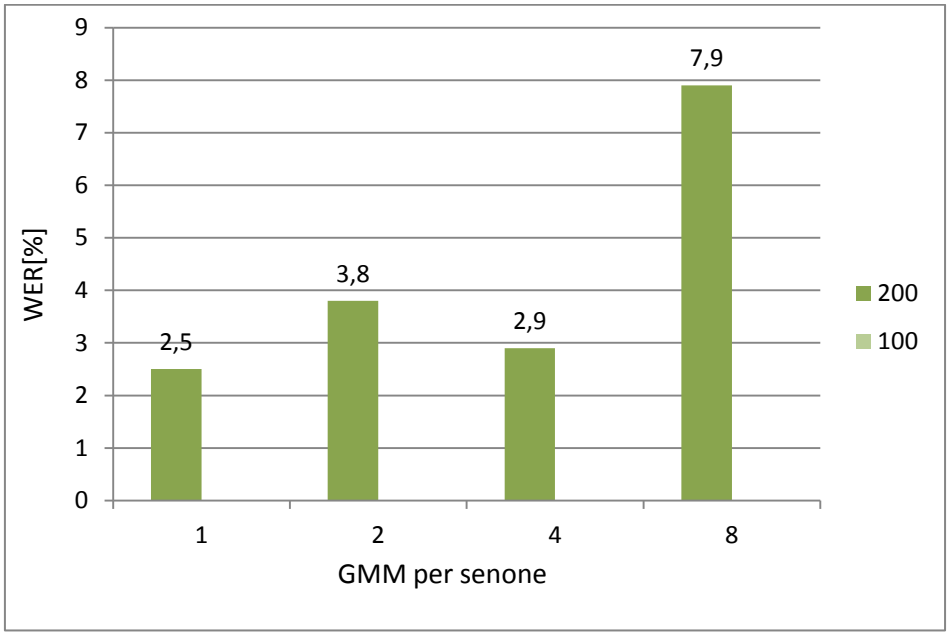

Fig. 4.3 Rezultate WER obținute în cadrul primului experiment

În Fig. 4.4 sunt prezentate rezultatele SER obținute pentru modele antrenate în cadrul acestui experiment.

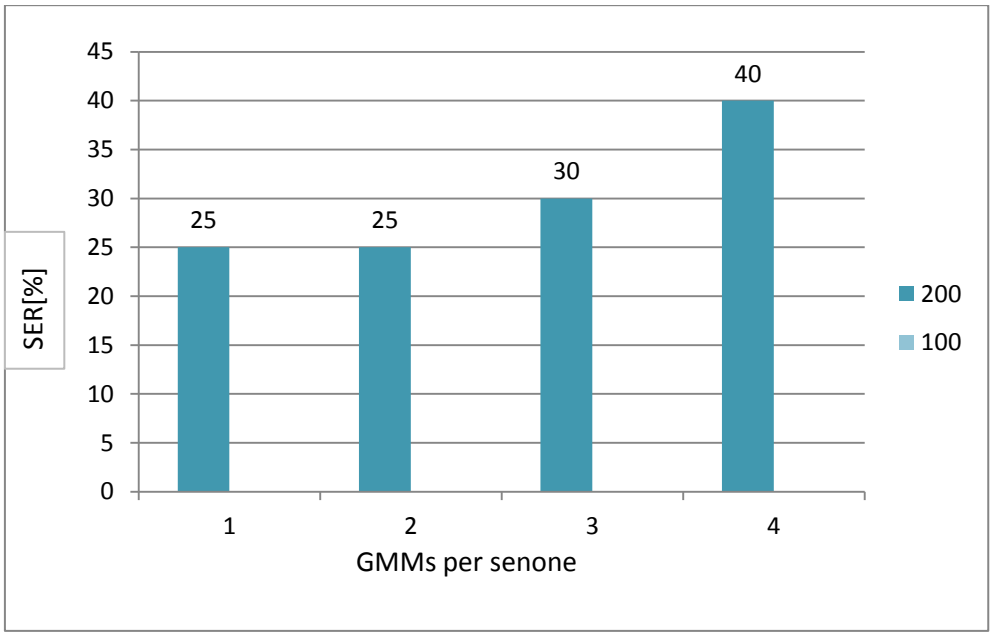

Fig. 4.4 Rezultatele SER obținute în cadrul primului experiment

Rezultatele obținute de sistemul dependent de vorbitor în urma decodării clipurilor audio ale altor 5 vorbitori, diferiți de cel folosit la antrenare. Pentru decodare s-au folosit câte 100 de fișiere per vorbitor. Modelul acustic folosit a fost cel antrenat cu 100 senone folosind 8 densități Gaussiene, folosind modele fonemice dependente de context.

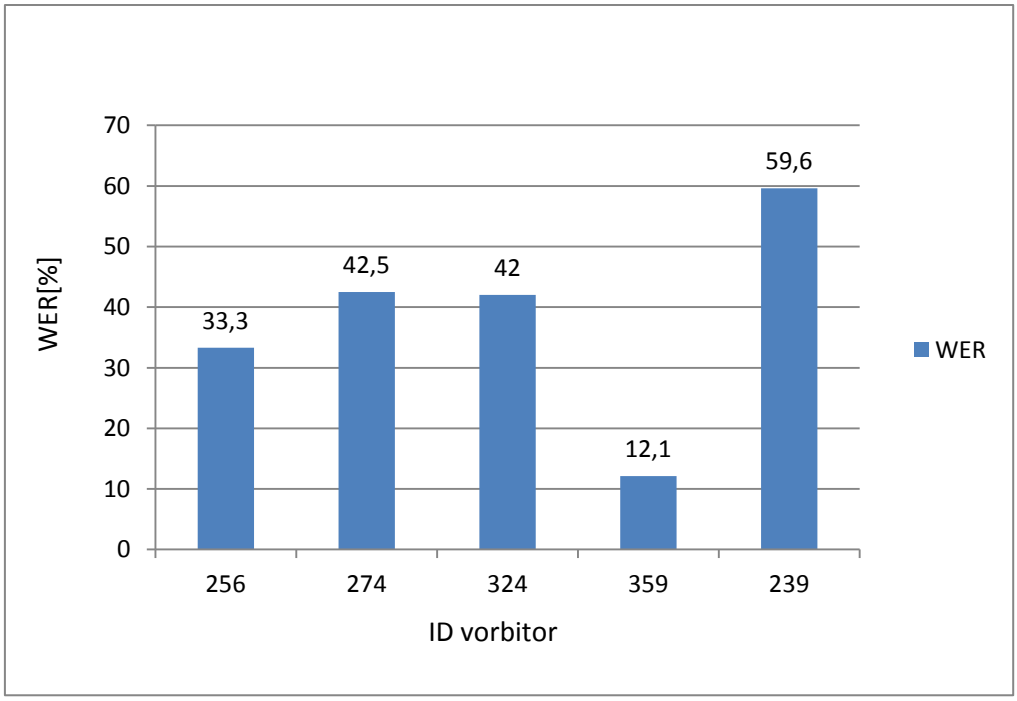

Fig. 4.5 Rezultatele WER obținute pentru 5 vorbitori din baza de date

Așa cum era de așteptat erorile sunt mai mari în cazul vorbitorilor necunoscuți, deoarece modelul este specializat pe baza caracteristicilor unui singur vorbitor, cel folosit în etapa de antrenare.

Un lucru ce trebuie remarcat în cadrul acestui experiment este faptul că modelele antrenate cu 100 de senone au obținut rezultate mai bune (erori nule) decât cele antrenate cu 200 de senone. O posibila explicație a acestui fapt este că in cazul modelului cu un numar mai mare de senone, este necesara estimarea unui numar mai mare de parametri in faza de antrenare. Acest lucru poate cauza supra-specializarea modelului, si ar putea fi tratat extinzand setul de date de antrenare.

4.3.2 *Experiment 2*

În cadrul experimentului 2 s-a creat un sistem independent de vorbitor, folosind 50 de vorbitori. Am antrenat modele folosind câte 80 de fișiere de la fiecare vorbitor, folosind 100 și 200 de senone, fiecare senone folosind câte 1, 2, 4 respectiv 8 densități Gaussiene de probabilitate.

În Fig. 4.6 și Fig. 4.7 sunt prezentate rezultatele WER, respectiv SER obținute în cazul decodării cu 100 de fișiere de la 40 de vobitori necunoscuți pentru modele antrenate în cadrul acestui experiment.

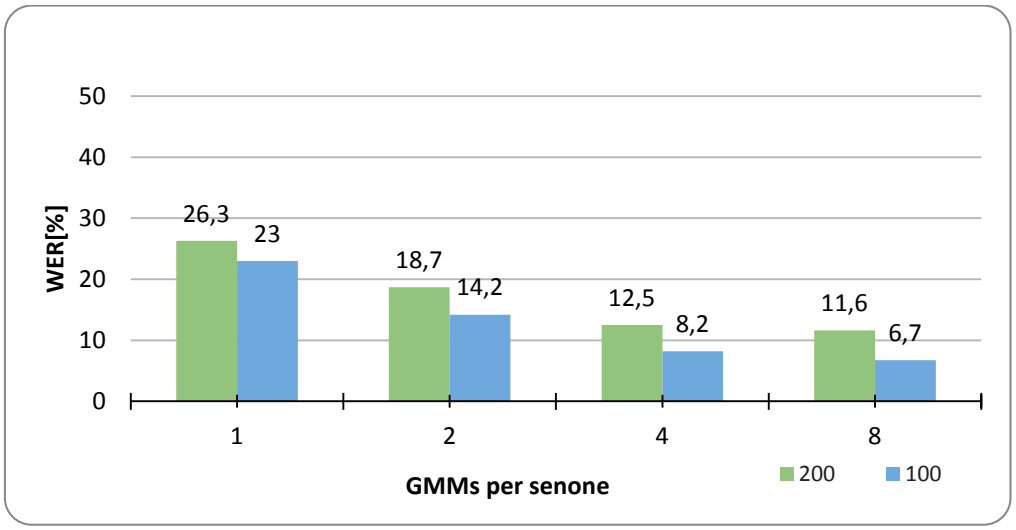

Fig. 4.6 Rezultatele WER obținute în cazul a 40 de vorbitori necunoscuți

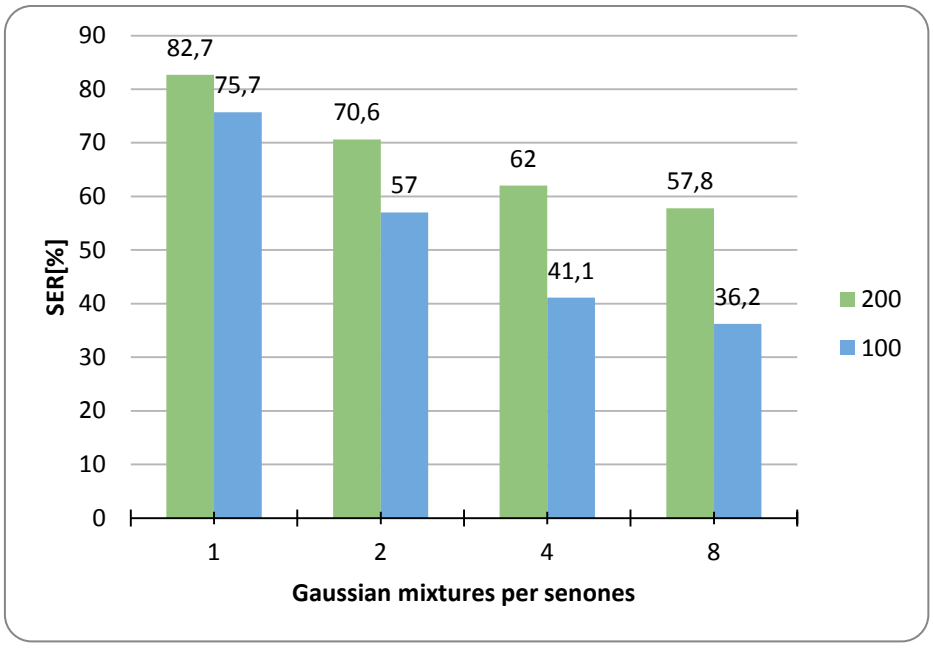

Fig. 4.7 Rezultatele SER obținute în cazul a 40 de vorbitori necunoscuți

În Fig. 4.8 sunt reprezentate rezultatele obținute de cei 50 de vorbitori folosiți pentru modelul antrenat cu 100 de senone cu 128 densități Gaussiene de probabilitate. În procesul de decodare s-au folosit restul de 20 de fișiere ce nu au fost folosite în etapa de antrenare de la fiecare vorbitor.

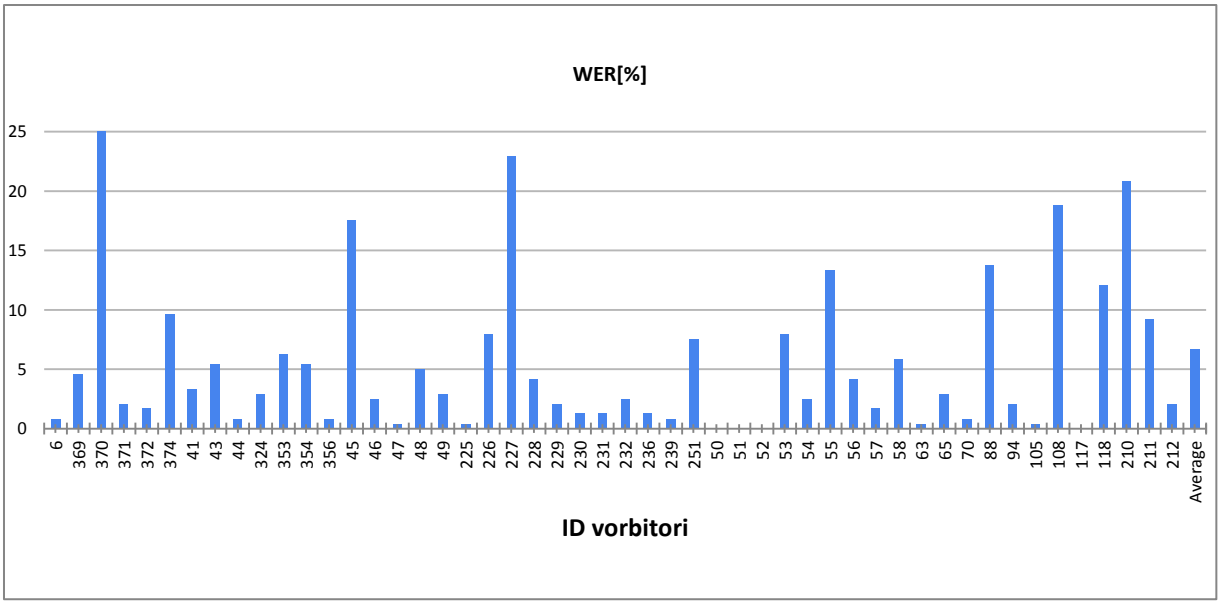

Fig. 4.8 Rezultatele obținute de cei 50 de vorbitori folosiți în etapa de antrenare

4.3.3 *Experiment 3*

În cadru acestui experiment s-a încercat optimizarea sistemului de recunoaștere de cifre independent de vorbitor prin înlăturarea din procesul de antrenare a vorbitorilor care au obținut un WER mai mare decât media și prin crearea unor modele acustice antrenate cu 100 și 200 de senone cu 1, 2, 4, 8, 16, 32, 64 și 128 de densități Gaussiene de probabilitate, modele fonetice dependente de vorbitor.

În Fig. 4.9 este reprezentat WER-ul obținut în urma decodării fișierelor audio pentru 40 de vorbitori necunoscuți (câte 100 de clipuri audio per vorbitor).

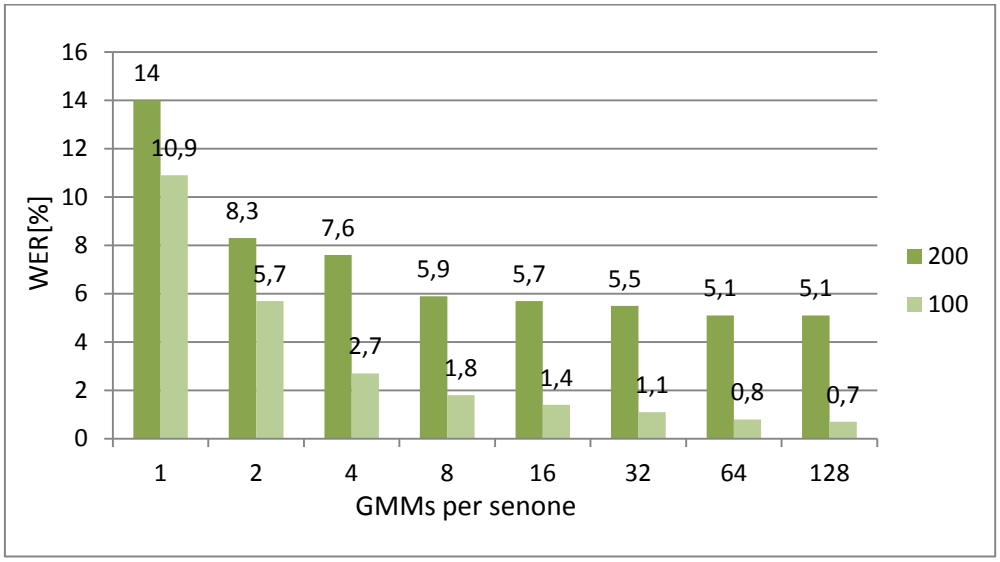

Fig. 4.9 Rezultatele obținute de cei 40 de vorbitori necunoscuți existenți în baza de date

CAPITOLUL 5

SERVICIUL DE RECUNOAȘTERE DE VORBITOR

5.1 ARHITECTURA SISTEMULUI DE RECUNOAȘTERE DE VORBITOR

Serviciul de recunoaștere de vorbitor are la bază un sistem de verificare de vorbitor. Sistemul va verifica dacă vorbitorul este utilizatorul pretins. Arhitectura sistemului de verificare de vorbitor este reprezentată în **Error! Reference source not found.**.

Sistemul de verificare de vorbitor va fi folosit atât în etapa de înrolare, cât și în etapa de autentificare a utilizatorilor. Astfel, în etapa de înrolare sarcina serviciului de recunoaștere de vorbitor constă din: extragerea caracteristicilor semnalului vocal și crearea unui model specific fiecărui utilizator pe baza acestor caracteristici. În etapa de autentificare sarcina sistemului de recunoaștere de vorbitor este de a extrage caracteristicile vocale din fișierul audio primit și va compara aceste caracteristici cu modelul vorbitorului pretins și cu UBM-ul.

UBM-ul sistemului de verificare de vorbitor a fost creat pe baza a aproximativ 5 ore de vorbire de la 90 de vorbitori, având 256 densități Gaussiene de probabilitate. Modelele vorbitorilor se vor crea pe baza acestui UBM, folosind algoritmul MAP.

În etapa de autentificare, modelul vorbitorului pretins și modelul UBM vor primi scoruri obținute evaluând probabilitatea datelor de intare sub fiecare dintre cele două modele. Modelul cu scorul cel mai mare este declarat câștigător. Astfel dacă UBM-ul a obținut un scor mai mare vorbitorul care dorește autentificare este considerat un impostor și nu va fi lăsat să acceseze contul. În caz contrar, vorbitorul va fi considerat utilizatorul pretins, iar dacă și celelalte verificări s-au finalizat cu succes atunci acesta va fi lăsat să își acceseze contul.

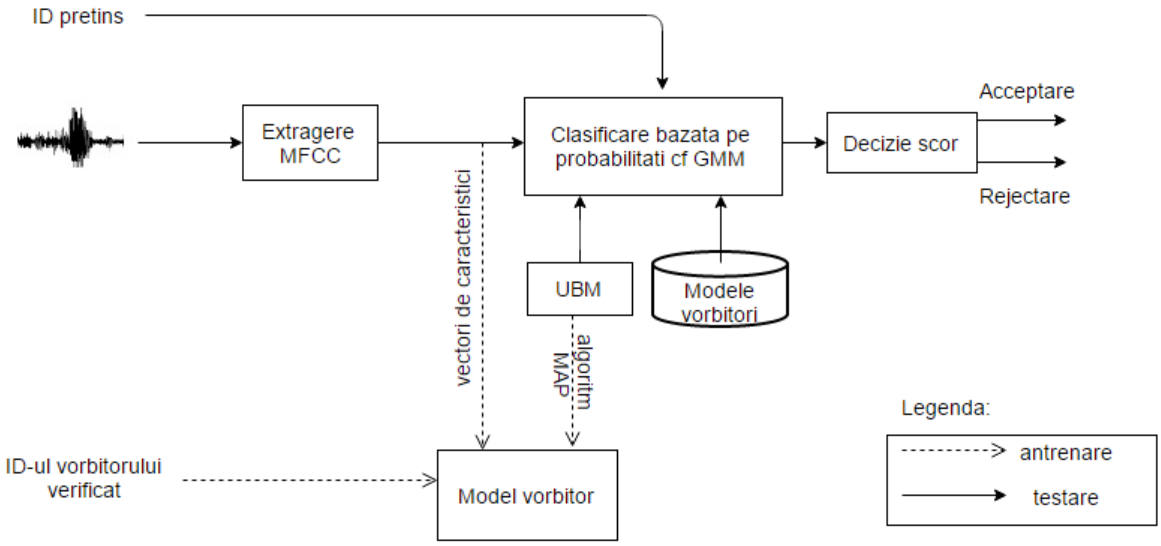

Fig. 5.1 Arhitectura sistemul de verificare de vorbitor

5.2 IMPLEMENTAREA SERVICULUI DE RECUNOAŞTERE DE VORBITOR

Serviciul de recunoaştere de vorbitor a fost dezvoltat în limbajul de programare Java, mediul de dezvoltare Eclipse LUNA, fiind un proiect de tip Dynamic Web. Sistemul de recunoaştere de vorbitor este inclus ca librărie în proiectul principal. Diagrama UML de clasă a proiectului este prezentată în figura de mai jos:

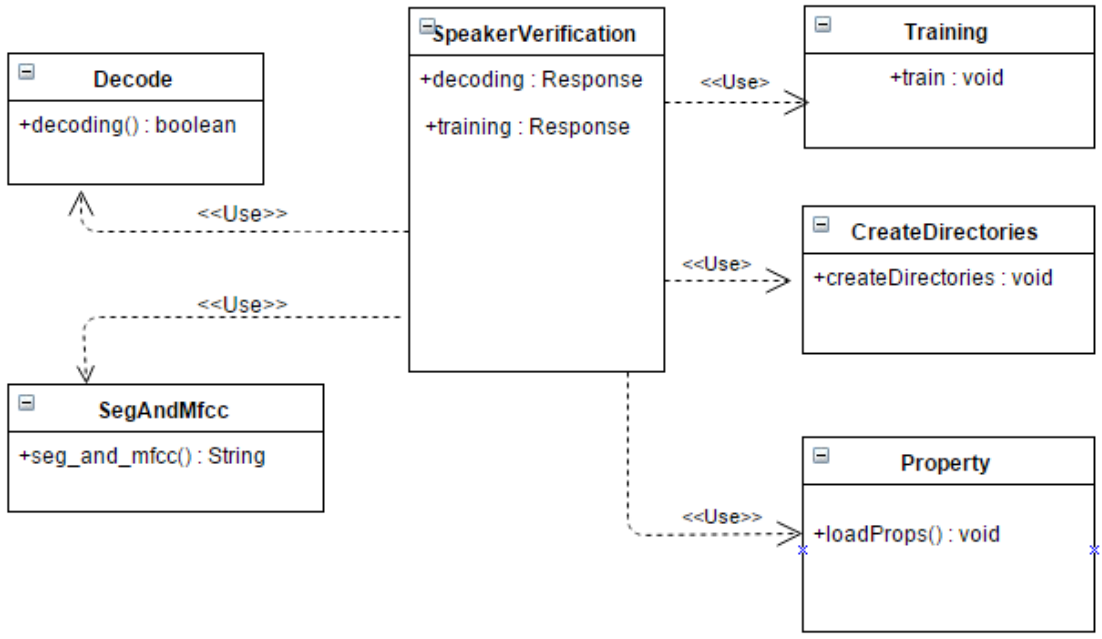

Fig. 5.2 Diagrama de tip clasă a proiectului

În continuare se vor descrie clasele principale ale proiectului.

5.2.1 *Clasa SpeakerVerification.java*

Clasa *SpeakerVerification.java* este singura clasă care este expusă REST (mai multe detalii implementare privind expunerea REST se găsesc în sectiuniile 4.2 şi 4.3), fiind clasa pricipală de unde se vor apela celelalte resurse. Prima metodă a acestei clase este metoda *training,* metodă de tip *Response,* ce va fi apelată la fiecare înrolare a unui nou utilizator în sistem.

Din formularul primit de la modulul de autentificare se vor extrage id-ul şi fişierele audio de înrolare ale utilizatorului. Aici se vor face următoarele verificări înainte de crearea modelului: se va verifica dacă utilizatorul cu id-ul respectiv are deja un model existent în sistem şi se va verifica dacă sunt cel puţin 10 fişiere audio primite. În caz contrar, răspunsul înapoiat de serviciul de recunoaştere de vorbitor va fi "Speaker already in the system", respectiv "Less than 10 audio files". Dacă verificările sunt finalizate cu succes se va începe procesul de antrenare a modelului. Astfel se vor salva fişierele audio pe disk, se vor crea fişierele seg şi fişierele mfcc, apelându-se funcţia *seg_and_mfcc,* din clasa *SegAndMfcc.java*, urmând să se apeleze funcţia *train* din clasa *Trainer.java.*

Cea de-a doua metodă expusă REST este metoda *decoding*, metodă de tip Response, ce va fi apelată în etapa de autentificare a utilizatorilor. Această metodă primeşte ca parametri un id de tip String şi fişierul ce urmează a fi decodificat. Se va verifica aici dacă există un model pentru utilizatorul cu id-ul primit şi dacă fişierul audio primit este în format wav. În caz contrar, autentificarea nu poate fi făcută, iar serviciul de recunoaştere de vorbitor va trimite mesajul "The speaker is not in the system! ", respectiv "No wav format!" către serviciul de autentificare. Dacă verificările s-au finalizat cu succes, atunci se va începe etapa de testare: se vor extrage coeficienţii MFCC şi se va apela metoda *decoding* din clasa *Decode.* Această metodă va întoarce mesajul "true", dacă modelul utilizatorului pretins a fost cel câştigător şi va întoarce mesajul "false", în cazul în care UBM-ul este cel ales câştigător.

5.2.2 *Clasa SegAndMfcc.java*

Clasa SegAndMfcc conţine metoda *seg_and_mfcc,* ce odată apelată va crea fişierele seg şi mfcc. Această metodă este apelată atât în etapa de înrolare, cât şi în etapa de autentificare. În cazul în care s-au creat cu succes fişierele seg şi mfcc se va returna Stringul "successful", iar în caz contrar se va returna Stringul "error". În cadrul acestei metode se va apela metodele *buildSegFile* şi *extractFeatures,* metode puse la dispoziţie de utilitarul LIUM.

5.2.3 *Clasa Training.java*

Clasa *Training.java* conţine funcţia *train* ce primeşte ca şi parametri: id-ul utilizatorului pentru care se doreşte crearea modelului, id-urile fişierelor audio ce urmează a fi folosite pentru antrenarea modelului, calea către fişierele audio (salvate pe disk) şi calea către fişierul de configurare. Modelul utilizatorului se va face pe baza UBM-ului existent în sistem. Crearea modelului se va face pe baza adaptării modelului UBM la caracteristicile vocale ale utilizatorului prin algoritmul MAP. Astfel se vor apela metodele initMAP şi trainMAP ce sunt puse la dispoziţie de utilitarul LIUM.

5.2.4 *Clasa Decode.java*

Clasa *Decode.java* conţine metoda *decoding* ce primeşte ca parametri: id-ul utilizatorului, id-ul fişierului ce trebuie decodificat, căile către fişierul configurabil şi fişierul audio. Aici se va

apela de două ori metoda *detectGender*, metodă implementată cu ajutorul utilitarului LIUM. După prima apelare a acestei metode se va obține scorul pentru modelul vorbitorului, iar a doua ne va oferi scorul obținut de UBM. Se va apela funcția *dumpResults* ce va salva scorurile obținute într-un fișier de tipul *csv* și va determina care dintre modele este cel câștigător.

5.2.5 *Clasa CreateDirectories.java*

Clasa CreateDirectories.java conține funcția de tip void *createDirectories* ce primește ca parametri: căile către directorul principal pentru antrenare/testare și directorul unde se găsesc fișierele audio de antrenare/testare și id-ul utilizatorului. Această funcție va crea directoarele utile în manipularea fișierelor, deoarece Serviciul de recunoaștere de vorbitor va stoca toate fișierele utile pe disk-ul local.

5.3 REZULTATE

În cadrul acestei lucrări s-au realizat mai multe experimente pentru a determina modelul vorbitorului cu cele mai bune performanțe privind măsurătorile FRR și FAR.

5.3.1 *Experimentul 1*

Pentru primul experiment s-au folosit vorbitorii existenți în baza de date *rodigits*. Acest experiment a presupus variația atât a numărului de fișiere audio folosite de la fiecare vorbitor, cât și numărul de densități Gaussiene de probabilitate folosite pentru fiecare model. S-au ales 10, 20 și 30 de fișiere audio pentru antrenare de la 91 de vorbitori, iar numărul de densități Gaussiene de probabilitate folosite a fost de 8, 16, 32, 64, 128, respectiv 256. În etapa de testare s-a folosit UBM-ul folosit în crearea fiecărui model.

Rezultatele FRR și FAR obținute în cadrul acestui experiment sunt reprezentate în Fig. 5.3, respectiv Fig. 5.4. Așa cum se observă din cele două figuri s-au obținut rezultate FRR și FAR mai bune pentru modelele antrenate cu 30 de fișiere de la fiecare vorbitor. Odată cu creșterea numărului densităților Gaussiene de probabilitate scad cele două erori, indiferent de numărul de fișiere folosit pentru crearea modelelor.

Modelul antrenat cu 30 de fișiere audio de la fiecare vorbitor, fiind format din 256 densități Gaussiene de probabilitate a obținut rezultatele cele mai bune, obținând un FRR de 0.01[%] și un FAR de 1.82[%].

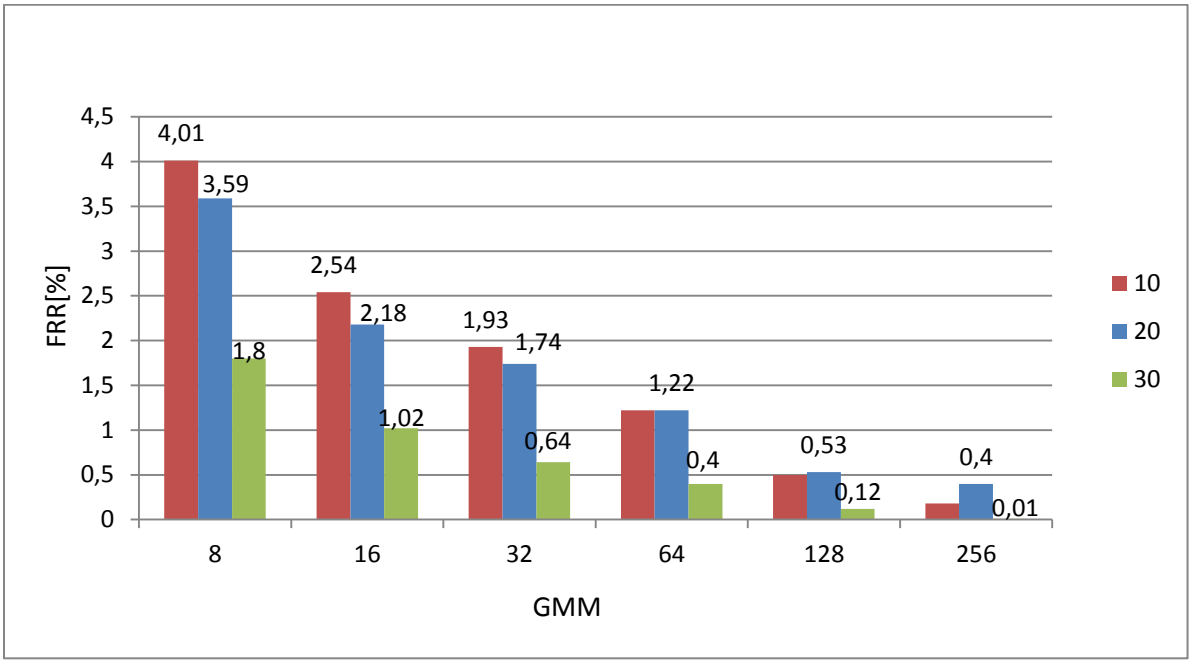

Fig. 5.3 Rezultate FRR obținute în cadrul primul experiment

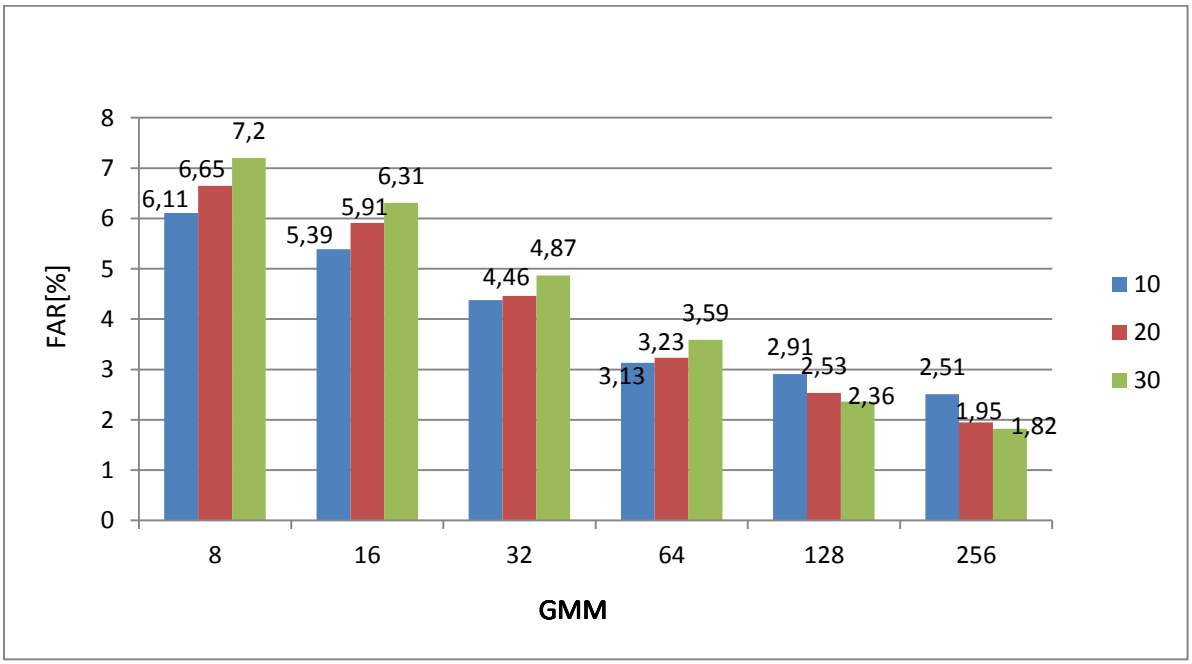

Fig. 5.4 Rezultate FAR obținute în cadrul primul experiment

5.3.2 *Experimentul 2*

Cel de-al doilea experiment a presupus eliminarea porțiunilor de liniște din fișierele audio. Pentru a elimina perioadele de liniște am creat un program Matlab. Programul pornește cu plasarea la începutul fișierelor audio a unei ferestre de 6000 de eșantioane (valoarea a fost stabilită experimental). Pe baza acestei ferestre stabilim amplitudinea maxima permisă pentru perioadele cu liniște. Ulterior, se parcurge întreg fișierul audio cu o fereastra de 140 eșantioane (valoare stabilită de asemenea experimental). Dacă în fereastra curenta se găsesc cel putin 6 eșantioane cu ampiltudinea mai mare decât cea de liniște atunci eșantionul central al ferestrei este considerat semnal util. După ce fiecare eșantion a fost etichetat ca fiind semnal de liniște/semnal util se realizeaza o post-procesare pentru a filtra acele semnalele utile foarte scurte (rezultate probabil dintr-o eroare de detecție): se consideră porțiuni de

semnal util doar acele porțiuni care conțin consecutiv peste 800 de eșantioane etichetate în prima faza ca fiind de semnal util.

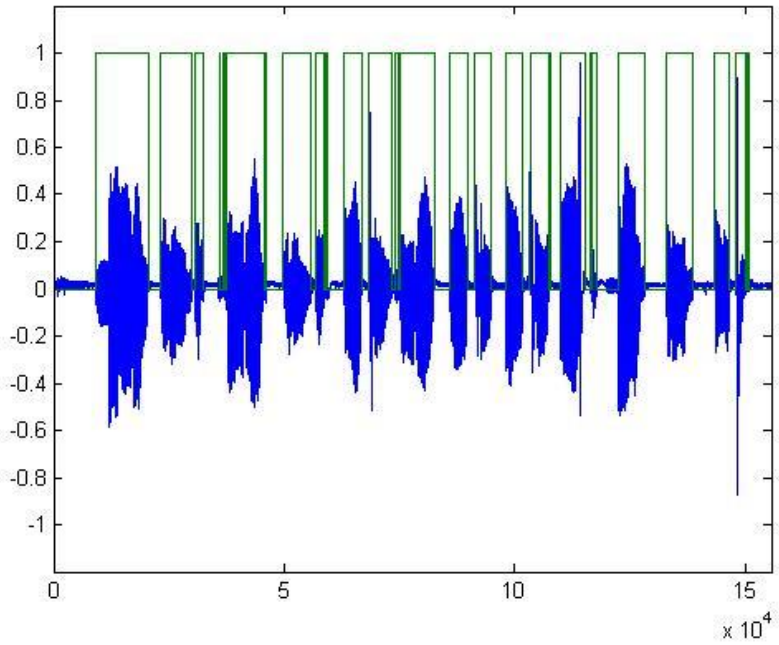

Fig. 5.5 Identificarea porțiunilor de semnal util a unui fișier audio

S-au creat modele folosind 10 fișiere audio de la fiecare vorbitor, folosind în total 91 de vorbitori. Modelel au fost create folosind 8, 16, 32, 64, 128 și 256 densități Gaussiene de probabilitate. Pentru etapa de testare s-au folosit 70 de fișiere de la fiecare vorbitor (aprox. 5.3 ore de vorbire). UBM-ul folosit pentru testare a fost creat pe baza a aproximativ o oră de vorbire de la cei 91 de vorbitori, folosind 8, 16, 32, 64, 128 și 256 densități Gaussiene de probabilitate. Rezultate FRR și FAR sunt reprezentate în Fig. 5.6, respectiv Fig. 5.7.

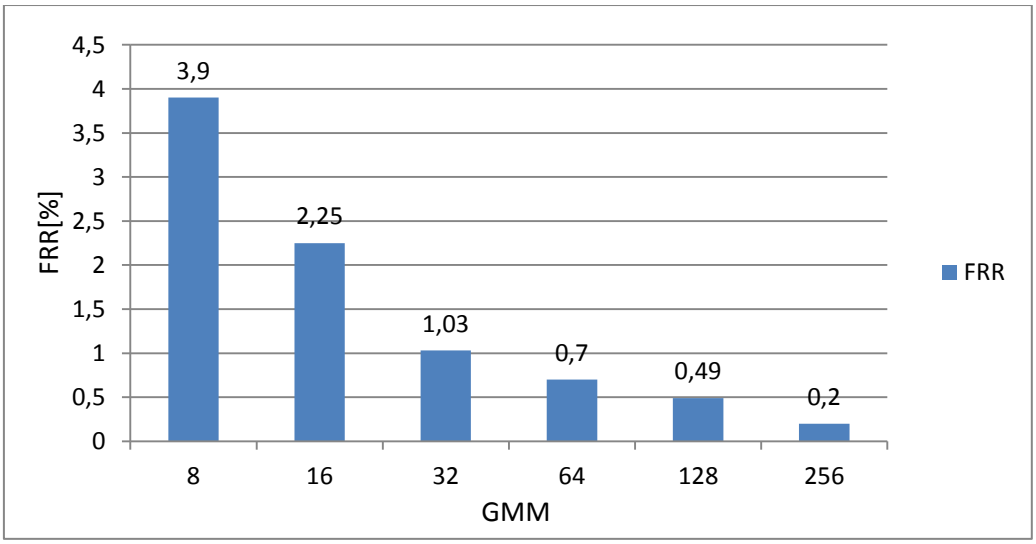

Fig. 5.6 Rezultate FRR obținute în cadrul experimentului 2

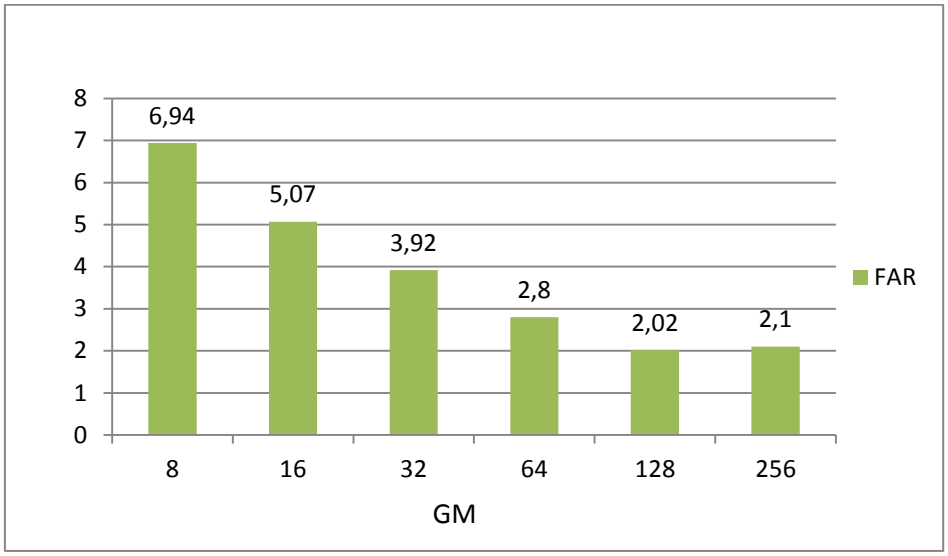

Fig. 5.7 Rezultate FAR obținute în cadrul experimentului 2

5.3.3 *Experimentul 3*

În cadrul acestui experiment s-au folosit pentru etapa de antrenare și etapa de testare vorbitorii ce se găseau în baza de date a Serviciului de autentificare prin voce pentru a testa sistemul în condiții reale de utilizare. Înregistrările din acestă bază de date nu sunt făcute sub anumite restricții ca în cazul înregistrărilor de pâna acum și sunt provenite de la utilizatorii reali ai Serviciului de autentificare. Pentru acest experiment s-au ales 10 vorbitori. Pentru crearea modelelor s-au ales 10 fișiere audio de la fiecare vorbitor, iar pentru etapa de testare s-au ales aproximativ 5 fișiere de la fiecare vorbitor (numărul de fișiere de test diferă de la vorbitor, la vorbitor în funcție de câte fișiere de autetificare s-au găsit în baza de date de la fiecare).

În cadrul acestui experiment s-au făcut trei teste. Primul test a constat din crearea modelelor utilizatorilor formate din 8, 16, 32, 64, 128 și 256 densități Gaussiene de probabilitate, folosindu-se in etapa de testare aceleași UBM-uri folosite pentru antrenarea modelelor. Rezultatele FRR și FAR obținute în cadrul acestui test sunt reprezentate în Fig. 5.8, respectiv Fig. 5.9.

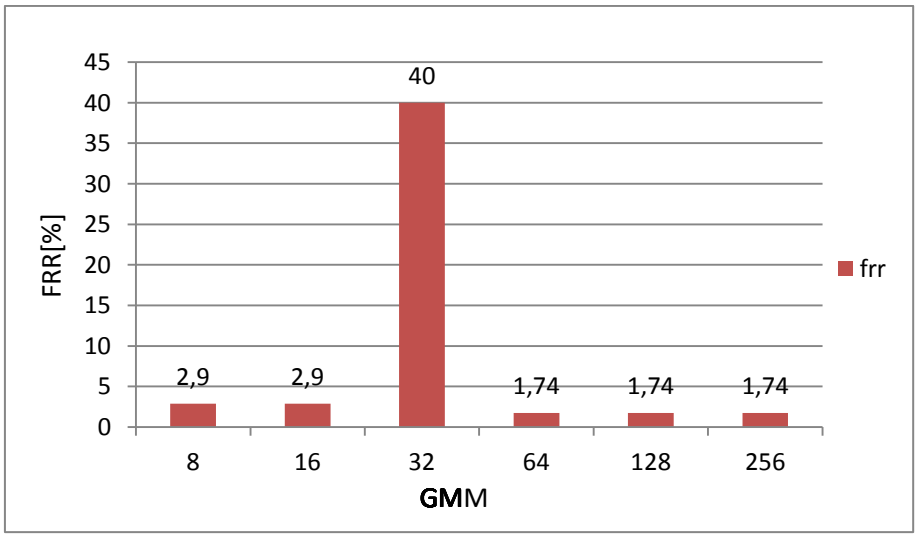

Fig. 5.8 Rezultate FRR obținute după primul test din cadrul experimentului 3

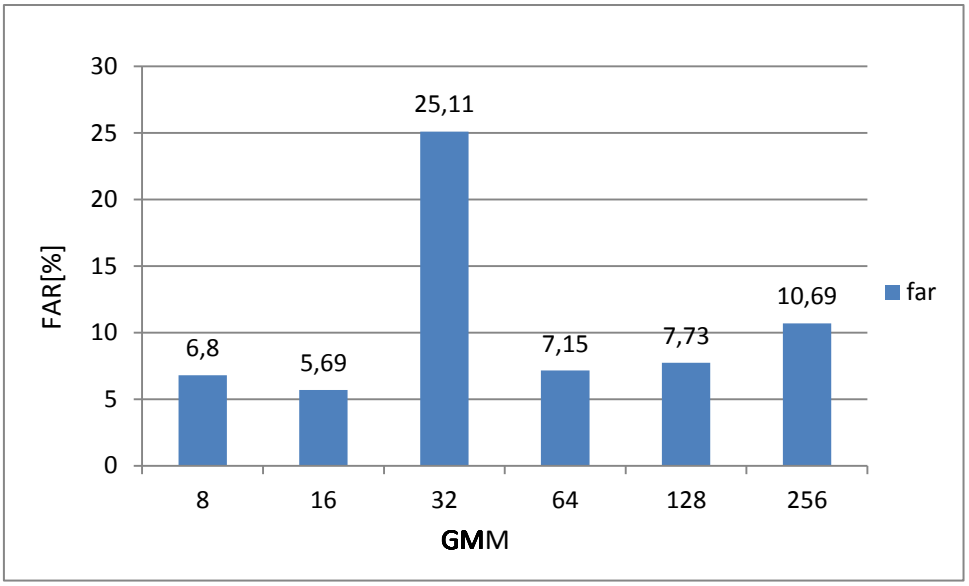

Fig. 5.9 Rezultate FRR obținute după primul test din cadrul experimentului 3

Rezultatele acestui test au fost surprizătoare, deorece ne-am fi așteptat ca și de data aceasta (la fel cum s-a întâmplat în cadrul primului experiment) rezultate cele mai bune să le obțină modelele formate din 256 densități Gaussiene de probabilitate, testate cu UBM-ul format tot din 256 densități Gaussiene de probabiliate. Din acest experiment reiese că scenariu optim de a folosi sitemul de verificare este de a antrena modelel utilizatorilor pe baza UBM-ului format din 16 densități Gaussiene de probabilitate, în etapa de testare folosindu-se același UBM.

Pentru cel de-al doilea test s-au creat modele formate din 8, 16, 32, 64, 128 și 256 densități Gaussiene de probabliltate. În etapa de testare s-a folosit un UBM creat pe baza a aproximativ 7.5 ore de vorbire, având o mixtură de 256 de Gaussiene. Rezultatele FRR și FAR obținute sunt reprezentate în Fig. 5.10, respectiv Fig. 5.11.

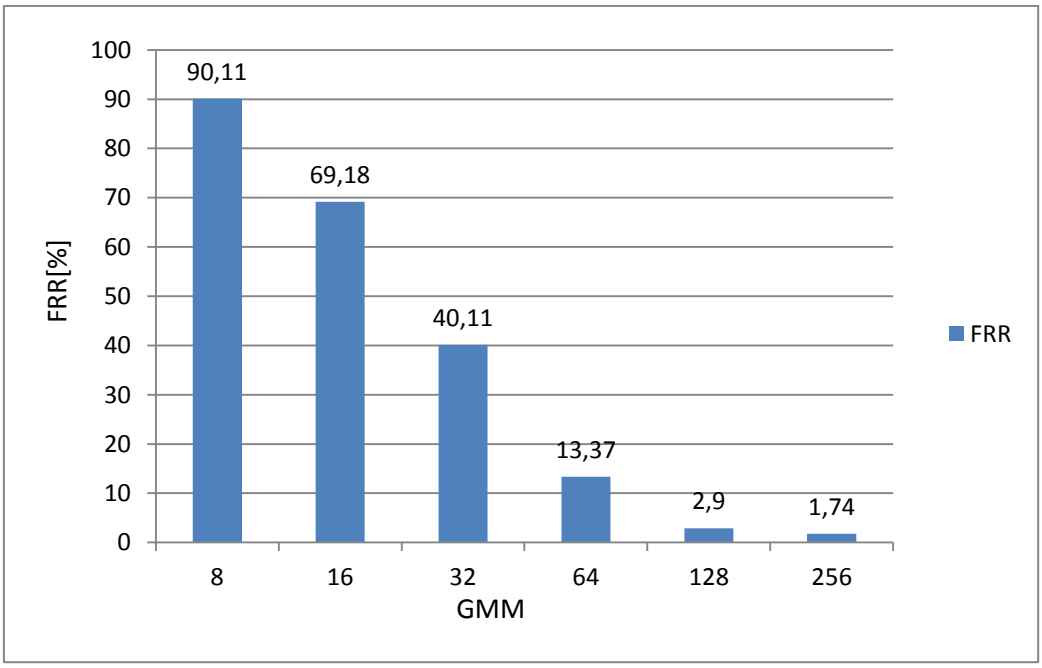

Fig. 5.10 Rezultate FRR obținute după cel de-al doilea test din cadrul experimentului 3

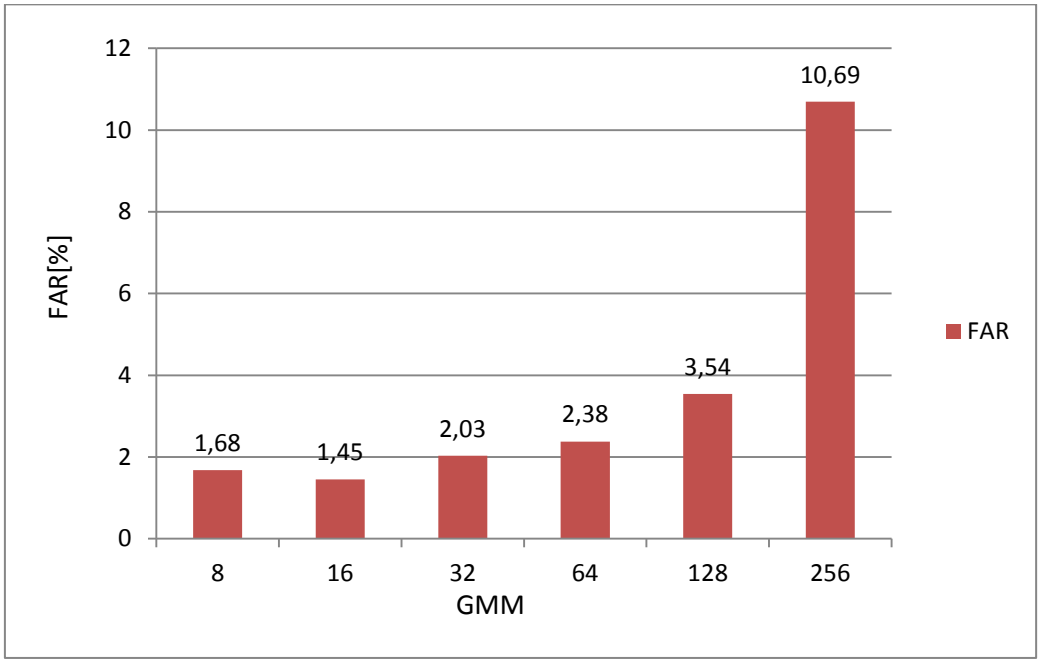

Fig. 5.11 Rezultate FRR obținute după cel de-al doilea test din cadrul experimentului 3

Rezultatele FRR și FAR obținute pentru fiecare vorbitor pentru modelele create pe baza UBM-ului format din 64 densități Gaussiene de probabilitate sunt reprezenate în Fig. 5.12, respectiv Fig. 5.13.

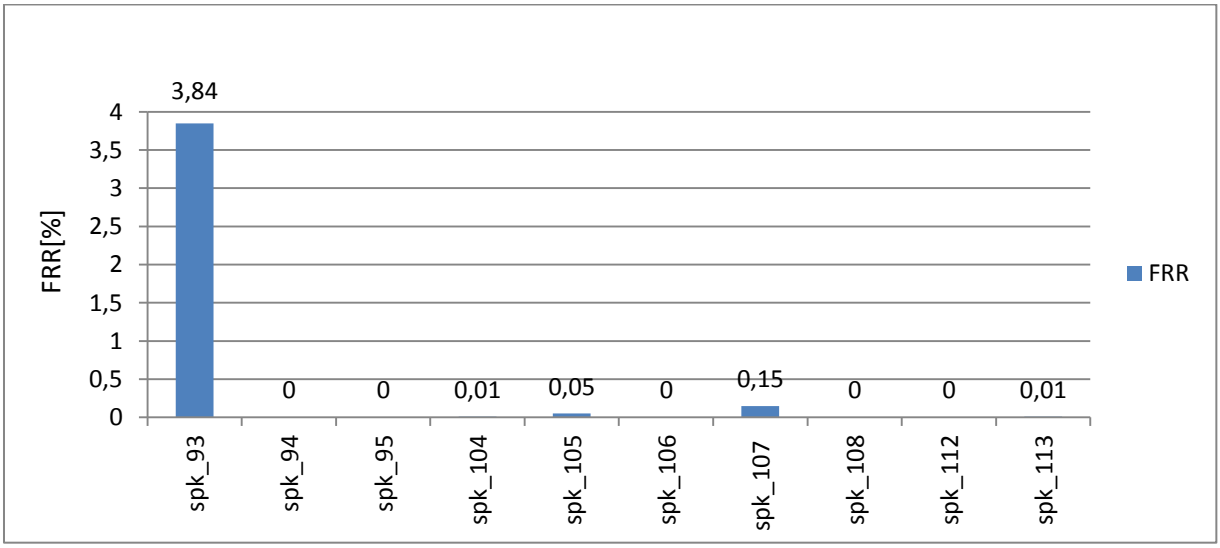

Fig. 5.12 Rezultate FRR obținute de fiecare vorbitor pentru modelul format din 64 densități de probabilitate

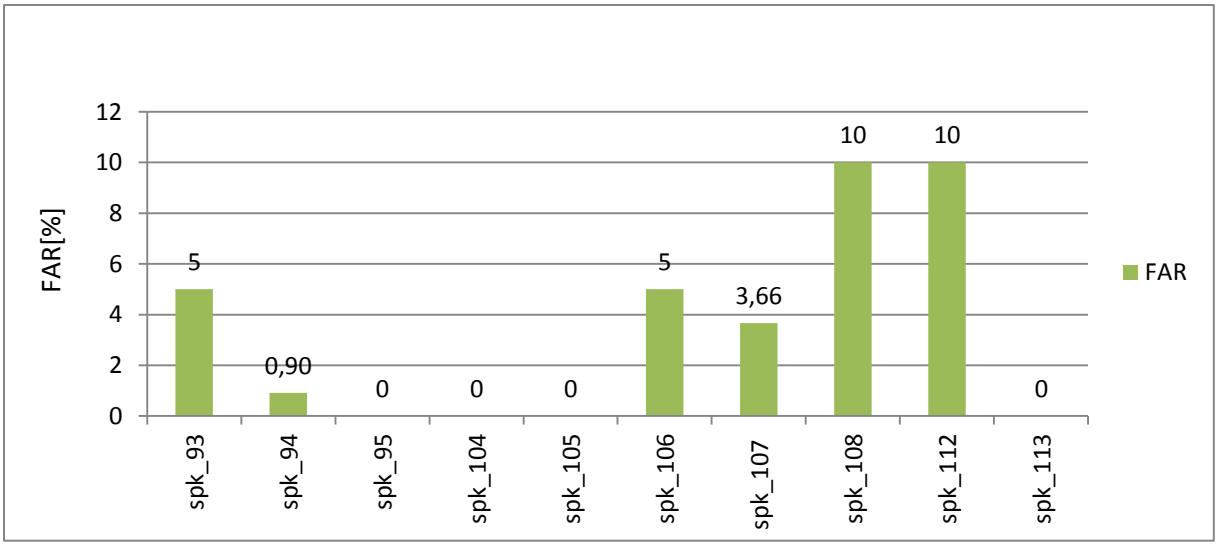

Fig. 5.13 Rezultate FAR obținute de fiecare vorbitor pentru modelul format din 64 densități de probabilitate

Pentru cel de-al treilea test din cadrul acestui experiment s-au creat modele ale utilizatorilor pe baza UBM-ului format din 8, 16, 32, 64, 128, respectiv 256 densități Gaussiene de probabilitate create pe baza a aprox 7.3 ore de vorbire de la 91 de vorbitori. În etapa de testare s-a folosit UBM-ul format din 16 densități Gaussiene de probabilitate. Rezultatele FRR și FAR obținute sunt reprezentate în **Error! Reference source not found.**, respectiv Fig. 5.15 .

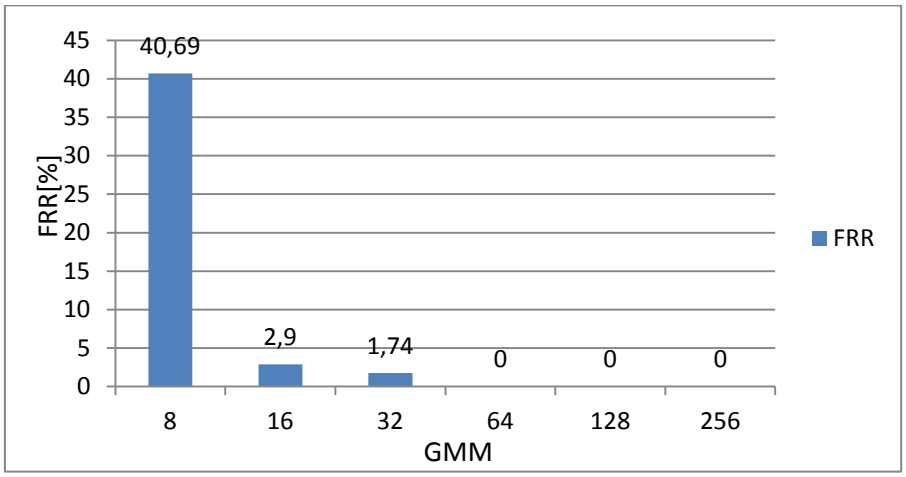

Fig. 5.14 Rezultate FRR obținute folosind în etapa de testarea UBM-ul format din 16 densități Gaussiene de probabilitate

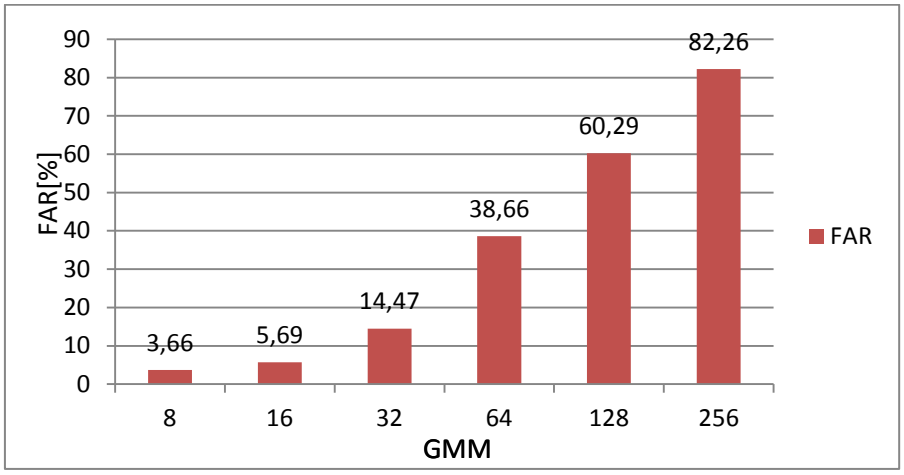

Fig. 5.15 Rezultate FAR obținute folosind în etapa de testarea UBM-ul format din 16 densități Gaussiene de probabilitate

Din urma acestui experiment putem trage concluzia că folosind în etapa de testare un UBM format dintr-un număr de densități Gaussiene de probabilitate mai mare decât numărul folosit în crearea modelelor se obține o eroare de falsă rejecție mare, dar eroare de falsă acceptare este mică. În cazul contrar, în care se folosțe în etapa de testare un UBM format dintr-un număr mai mic de densități Gaussiene de probabilitate decât cel al modelului vorbitorilor, se obțin erori de falsă rejecție foarte mici, dar erori de falsă acceptare foarte mari.

După rularea celor trei experimente am ales ca modelele vorbitorilor să se creeze pe baza UBM-ului antrenat cu 7,3 ore de vorbire provenit de la 91 de vorbitori diferiți, având 256 densități Gaussiene de probabilitate, folosind în etapa de testare același UBM.

CONCLUZII

Pentru a crea un serviciu de autentificare prin voce robust a fost nevoie de realizarea mai multor experimente pentru determinarea parametrilor optimi. S-au făcut numeroase teste atât pentru sistemul de recunoaştere de cifre, cât şi pentru sistemul de verificare de vorbitor. Testele au fost făcute atât în condiţii de laborator, cât şi în condiţii reale.

În cazul sistemului de recunoaştere de vorbire modelul cu cele mai bune performanţe a fost modelul antrenat cu 100 de senone cu 256 densităţi Gaussiene de probabilitate. Tot în cadrul testelor realizate pentru acest sistem s-a observat ca modelele antrenate cu 200 de senone aveau performanţe mai slabe, decât cele antrenate cu 100 de senone. O posibila explicaţie a acestui fapt este că in cazul modelului cu un numar mai mare de senone, este necesara estimarea unui numar mai mare de parametri in faza de antrenare. Acest lucru poate cauza supra-specializarea modelului, si ar putea fi tratat extinzand setul de date de antrenare.

În cazul sistemului de recunoaştere de vorbitor s-au realizat trei experimente. Primul experiment ne-a dus la concluzia că avem nevoie de un număr de aproximativ 30 de fişiere audio de antrenare de la fiecare vorbitor pentru a obţine performanţe bune. O altă concluzie ce trebuie luată în cadrul experimentelor realizate este că trebuie facut un compromis privitor la erorile de falsă rejecţie şi falsă acceptare, deorece atunci când avem o eroare de falsă rejecţie mică, vom avea o eroare de falsă accepatre mare, şi invers.

În cazul celui de-al doilea experiment am putut observa că porţiunile de linişte influenţează ratele de erori. Eliminând porţiunile de linişte din înregistrările audio s-au obţinut rezultate în general mai bune. În urma experimentului 3 putem trage concluzia că folosind în etapa de testare un UBM format dintr-un număr de densităţi Gaussiene de probabilitate mai mare decât numărul folosit în crearea modelelor se obţine o eroare de falsă rejecţie mare, dar eroare de falsă acceptare este mică. În cazul contrar, în care se foloşte în etapa de testare un UBM format dintr-un număr mai mic de densităţi Gaussiene de probabilitate decât cel al modelului vorbitorilor, se obţin erori de falsă rejecţie foarte mici, dar erori de falsă acceptare foarte mari.

În general modul de alegere al modelului se va face în funcție de nivelul de securitate dorit. Astfel sistemele care se doresc a fi foarte sigure au o eroare de falsă acceptare mică, dar o eroare de falsă rejecție mare, ceea ce poate fi supărător pentru utilizator.

Atât sistemul de recunoaștere de cifre, cât și sistemul de verificare de vorbitor au fost expuse ca servicii REST. Așa cum s-a discutat, serviciul REST prezintă atât avantaje, cât și dezavantaje. Printre cele mai importante avantaje ale acestui serviciu se numără și ușurința și simplitatea de implementare. Un alt avantaj important este acela că este independent de platformă și de limbajul de programare. Astfel un client C++ poate apela un server Java sau invers. Serviciul REST prezintă și dezavantaje. Un dezavantaj major este acela ca se folosesc cookie-uri. Acestea sunt păstrate în text clar și pot păstra un risc de securitate din moment ce oricine ar putea deschide și manipula un cookie. Poți cripta sau decripta manual un astfel de cookie, dar acest lucru necesită cod scris în plus și poate afecta performanțele aplicației datorită timpului necesar criptării și decriptării.

Consider că lucrarea de față și-a atins scopul propus, acela de a realiza un sistem robust de autentificare pe bază de voce care să asigure un nivel înalt de securitate. Ambele module din componența Serviciului de autentificare au fost proiectate astfel încât să asigure un nivel ridicat de securitate. Pentru proiectarea unor astfel de module care să respecte aceste cerințe s-au realizat mai multe experimente prin care s-au creat numeroase modele. Dintre toate modelele create s-au păstrat acele modele pentru care am obținut rezulatele cele mai bune, asta însemnând pentru sistemul de recunoaștere de vorbire o rată de cuvinte eronată mică, iar pentru sistemul de recunoaștere de vorbitor s-au creat modele pentru fiecare vorbitor după modelul acelora care au obținut o rată de falsă acceptare si o rată de falsă rejecție cât mai mică.

Bibliografie

[1] Roberto Togneri, "An Overview over Speaker Identification: Accuracy and Robustness Issues", IEEE Circuits and System Magazine, 2011

[2] Horia Cucu, "Towards a speaker-independent, large-vocabulary continuous speech recognition system for Romanian", PhD Thesis, Universitatea "Politehnica" București, Oct 2011.

[3] Andi Buzo, "Automatic speech recognition over mobile telecommunication channels", PhD Thesis, Universitatea "Politehnica" București, Oct 2011.

[4] Tehnologia Limbajului Vorbit, Raport de activitate,
http://www.sdettib.pub.ro/admin/images/raport/2013-10-28-2604237169-Referat_1_final.pdf

[5] Aplicații pentru sinteza vorbirii, Referat III,
http://www.racai.ro/media/Referat3TBoros1.pdf

[6] Securizarea accesului la sistemele de comunicații prin metode de identificare biometrică,
http://www.agir.ro/buletine/739.pdf

[7] Horia Cucu, Proiect de cercetare-dezvoltare în Tehnologia Vorbirii, îndrumar de proiect,
http://speed.pub.ro/speed3/wp-content/uploads/2014/02/Indrumar-de-proiect-PCDTV-v11.pdf

[8] Douglas A. Reynolds, Automatic Speaker Recognition Using Gaussian Mixture Speaker Models,
https://www.ll.mit.edu/publications/journal/pdf/vol08_no2/8.2.4.speakerrecognition.pdf

[9] Anca Popescu, Carmen Pătrașcu, Interfețe Om-Mașină, Curs 2,
http://lpsv.pub.ro/images/IOM/CURS_-_IOM/IOM_2_APsmall.pdf

[10] Yaxin Zhang, "Phoneme-Based Vector Quantization in a Discrete HMM Speech Recognizer", IEEE Transactions on speech and audio processing, vol. 5, no. 1, Ianuarie 1997

[11] Y. Zhang, R. Togneri, Speaker-independent isolated word recognition using multiple hidden Markov models, IEE Proc.-Vis. Image Signal Process., Vol. 141, No. 3, June 1994

[12] Domokos Jozsef, "Contribuții la recunoșterea vorbirii continue și la procesarea limbajului natural", Teză de doctorat, Universitatea Tehnică din Cluj-Napoca, 2009

[13] Margaret Rouse, Protocolul HTTP,
http://searchwindevelopment.techtarget.com/definition/HTTP

[14] Margaret Rouse, HTTP 1.1,
http://searchsoa.techtarget.com/definition/HTTP-11

[15] Swati Dhingra,"REST vs. SOAP. How to choose the best Web service",
http://searchsoa.techtarget.com/tip/REST-vs-SOAP-How-to-choose-the-best-Web-service

[16] Lalit Raghuvanshi, "Advantages and Desadvantages "
http://www.webcodeexpert.com/2013/03/what-is-cookie-advantages-and.html