# Construcţia unui sistem simplu de recunoaştere a vorbirii continue

*- recunoaşterea secvenţelor audio ce conţin cifre -*

## I. Introducere

Procesul recunoaşterii vorbirii continue (RVC) vizează transformarea unui semnal audio ce conţine vorbire într-o succesiune de cuvinte. Un sistem de recunoaştere a secvenţelor audio de cifre este un sistem de recunoaştere a vorbirii continue cu vocabular redus, în sensul că singurele cuvinte ce pot fi recunoscute de către sistem sunt cele zece cifre ale sistemului zecimal: zero, unu, ..., nouă.

### Arhitectura generală a unui sistem de recunoaştere a vorbirii continue

Arhitectura generală a unui sistem de recunoaştere automată a vorbirii (RAV) este prezentată în Figura 1. Din figură reies două lucruri esenţiale în ceea ce priveşte procesul de recunoaştere a vorbirii: a) recunoaşterea se face utilizând o serie de parametri vocali extraşi din semnalul vocal (nu direct, utilizând mesajul vorbit) şi b) recunoaşterea se face pe baza unor modele (acustic, fonetic şi lingvistic) dezvoltate în prealabil.

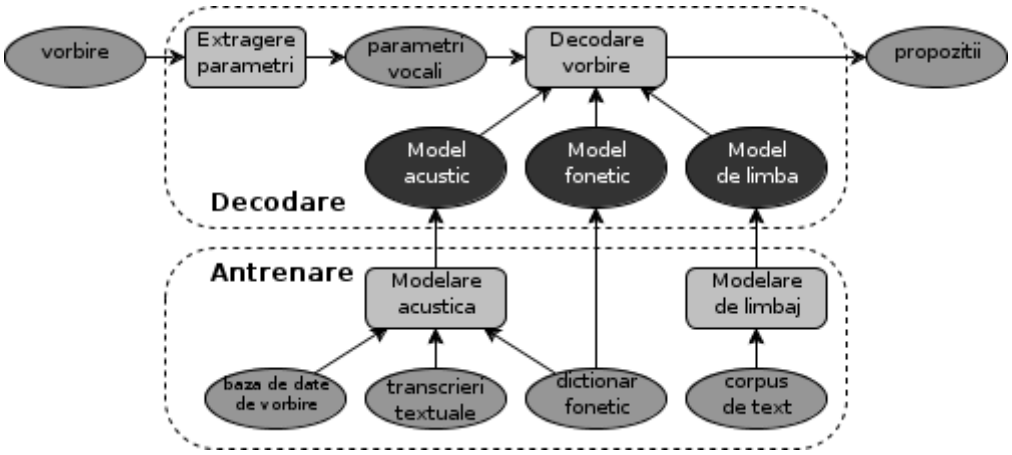


**Figura 1. Arhitectura generală a unui sistem de recunoaştere a vorbirii**

Modelul acustic are rolul de a estima probabilitatea unui mesaj vorbit, dată fiind o succesiune de cuvinte. În sistemele de RVC state-of-the-art modelul acustic nu foloseşte cuvinte ca unităţi acustice de bază pentru că: a) fiecare sarcină de RAV are un vocabular de cuvinte diferit pentru care nu există modele deja antrenate şi nici date de antrenare disponibile şi b) numărul de cuvinte diferite dintr-o limbă este prea mare. În locul cuvintelor, se utilizează unităţi acustice de bază sub-lexicale, de exemplu foneme, sau, mai recent, chiar unităţi sub-fonemice numite senone. Prin urmare, modelul acustic este format dintr-un set de modele pentru foneme (sau senone) care se conectează în timpul procesului de decodare pentru a forma modele pentru cuvinte şi apoi modele pentru succesiuni de cuvinte. Acestea sunt utilizate în final pentru a estima probabilitatea ca mesajul rostit

de vorbitor să fie format dintr-o succesiune sau alta de cuvinte. Experienţa a arătat că această abordare generativă pentru modelele acustice poate fi foarte bine implementată utilizând modele Markov ascunse (hidden Markov models - HMM). Pentru mai multe informaţii despre modul cum funcţionează modelele acustice consultaţi bibliografia îndrumarului de proiect.

Modelul de limbă este utilizat în timpul decodării pentru a estima probabilităţile tuturor secvenţelor de cuvinte din spaţiul de căutare. În general, rolul unui model de limbă este de a estima probabilitatea ca o secvenţă de cuvinte $W = w_1, w_2, ..., w_n$, să fie o propoziţie validă a limbii. Probabilitatea acestor secvenţe de cuvinte ajută foarte mult modelul acustic în procesul de decizie. De exemplu, aceste două fraze: *casa ta e mare?* şi *casat a Ema re?* sunt similare din punct de vedere acustic, însă cea de-a doua nu are niciun sens. Rolul modelului de limbă este acela de a asigna o probabilitate mult mai mare primei secvenţe de cuvinte şi, prin urmare, de a ajuta sistemul de RAV să decidă în favoarea acesteia.

Modelul fonetic are rolul de a conecta modelul acustic (care estimează probabilităţile acustice ale *fonemelor*) cu modelul de limbă (care estimează probabilităţile secvenţelor de *cuvinte*). De cele mai multe ori modelul fonetic este un dicţionar de pronunţie care asociază fiecărui cuvânt din vocabular una sau mai multe secvenţe de foneme adecvate, reprezentând modul în care se poate pronunţa respectivul cuvânt.

Figura 1 ilustrează, de asemenea, procesele implicate în dezvoltarea unui sistem de RAV, dar şi resursele necesare creării modelelor acustice, lingvistice şi fonetice. Modelul acustic se construieşte strict pe baza unui set de clipuri audio înregistrate asociate cu transcrierea textuală a mesajelor vorbite şi a unui dicţionar fonetic ce cuprinde toate cuvintele din respectiva transcriere textuală. În cazul sistemelor de RVC cu vocabular extins se folosesc modele de limbă statistice, care se construiesc utilizând corpusuri de text de dimensiuni cât mai mari şi cât mai adaptate domeniului din care fac parte mesajele vorbite ce trebuie decodate. În sistemele de RVC cu vocabular redus (cum este şi sistemul pe care îl veţi dezvolta în continuare) se folosesc preponderent modele de limbă de tip gramatică cu stări finite, iar pentru construcţia acestora nu este nevoie de resurse textuale.

## II. Înregistrarea unei baze de date de clipuri audio

Aşa cum am precizat mai sus, pentru a construi modelul acustic al unui sistem de recunoaştere a vorbirii este nevoie în primul rând de o bază de date de clipuri audio înregistrate. Pentru proiectul de recunoaştere de secvenţe audio ce conţin cifre veţi înregistra un set de 100 de clipuri audio a câte 12 cifre fiecare. Pentru înregistrarea clipurilor audio veţi folosi o aplicaţie de înregistrări accesibilă on-line la adresa http://speed.pub.ro/speech-recorder. Pentru a accesa această aplicaţie aveţi nevoie de numele de utilizator şi de parola care v-au fost comunicate la prima şedinţă de proiect.

Conectaţi un microfon la calculator şi verificaţi ca volumul acestuia să fie la maxim.

Porniţi browserul şi încărcaţi pagina http://speed.pub.ro/speech-recorder. În cazul în care aplicaţia nu se încarcă va trebui să instalaţi plug-in-ul Java pentru browser.

După ce aplicaţia s-a încărcat, urmaţi instrucţiunile de pe primul ecran pentru a calibra microfonul şi sistemul audio. Este foarte important ca înregistrările să se facă într-o cameră în care nu există surse de zgomot constant (de exemplu un ventilator sau un frigider), dar nici surse de zgomot intermitent (de exemplu un televizor, alte persoane care vorbesc, etc.). Etapa de calibrare are ca scop configurarea corectă a setărilor microfonului în condiţii de linişte. În cazul în care aveţi probleme la etapa de calibrare şi nu reuşiţi să treceţi de această etapă modificând volumul microfonului, vorbind mai tare/încet, utilizând posibilele setări speciale +10/20/30dB ale driverului audio, contactaţi îndrumătorul de proiect trimiţâdu-i un email.

După ce etapa de calibrare a fost depăşită cu succes, completaţi în câmpurile corespunzătoare numele de utilizator şi parola ce v-au fost comunicate de îndrumătorul de proiect la prima şedinţă. Aplicaţia va afişa un al doilea ecran în care trebuie să selectaţi din câmpul *Select speaker* numele vorbitorului curent. În cazul în care rulaţi aplicaţia pentru prima dată va trebui să selectaţi *new speaker* pentru a crea un nou vorbitor (pentru persoana în cauză). Completaţi apoi câmpurile cerute şi apăsaţi butonul *Save* pentru a salva datele noului vorbitor. Înainte de a putea continua va trebui să acceptaţi termenii şi condiţiile de utilizare ale aplicaţiei apăsând butonul *I agree*.

Odată ce noul vorbitor a fost creat aplicaţia va reveni la ecranul de bază, iar noul vorbitor va fi activ. Selectaţi în acest moment, din câmpul *Phrase group*, grupul de fraze corespunzător proiectului vostru: *rodigits*. Aplicaţia va afişa fraza 1 a acestui grup de fraze (*7779 6569 0341*) şi statusul ei (*not recorded yet*). În acest moment puteţi înregistra această primă frază apăsând butonul *Record*, pronunţând cele 12 cifre **cât mai natural** una câte una (şapte, şapte, şapte, nouă, ..., unu) şi apasând în final butonul *Stop*. Dacă verificarea semnalului audio se încheie cu succes fraza va fi uploadată pe server, iar aplicaţia va afişa automat informaţiile pentru fraza a doua. Continuaţi cu înregistrările în acest mod până când terminaţi de înregistrat toate cele 100 de fraze.

Ori de câte ori aveţi impresia că înregistrarea curentă nu a fost făcută corect (a apărut un zgomot neaşteptat, v-aţi bâlbâit, aţi pronunţat un cuvânt în plus sau în minus, aţi ezitat, etc.) verificaţi corectitudinea ei apăsând butonul *Play last* şi ascultând înregistrarea. În cazul în care înregistrarea nu a fost făcută corect apăsaţi butonul <, reveniţi astfel la fraza anterioară şi reînregistraţi-o. După ca aţi terminat de înregistrat toate frazele puteţi utiliza butoanele <, >, <<. >> pentru a parcurge şi reasculta toate clipurile audio, reînregistrandu-le pe cele care vi se par incorecte.

După ce aţi terminat de înregistrat şi de verificat tot setul de fraze „rodigits" îl veţi anunţa pe îndrumătorul de proiect trimiţâdu-i un email în care să specificaţi numele vorbitorului (aşa cum l-ati creat în aplicaţia de înregistrări on-line) pentru care doriţi extragerea fişierelor audio din baza de date. În cel mai scurt timp veţi fi contactat pentru a vi se comunica locaţia (de pe serverul de dezvoltare) unde se află clipurile audio. Până atunci, puteţi continua cu crearea dicţionarului fonetic şi a listei de foneme (de clipurile audio veţi avea nevoie când veţi ajunge la capitolul IV. Antrenarea modelului acustic).

## III. Crearea dicţionarului fonetic şi a listei de foneme

Un dicţionar fonetic este un instrument lingvistic care specifică modul în care se pronunţă cuvintele unei limbi. Altfel spus, dicţionarul fonetic face corespondenţa între forma scrisă şi forma fonetică a

cuvintelor unei limbi. Forma fonetică a unui cuvânt este o succesiune de foneme (fonemul fiind unitatea de sunet fundamentală a unei limbi). Fonemele limbii române, codificarea internă utilizată în continuare şi câteva exemple de cuvinte transcrise fonetic sunt prezentate în Tabelul 1.

Într-un sistem de recunoaştere a vorbirii continue dicţionarul fonetic are rolul de a face legătura între modelul acustic (care modelează modul de producere a *sunetelor specifice limbii*) şi modelul de limbă (care modelează modul în care se succed *cuvintele limbii*). În conecinţă dicţionarul fonetic trebuie să conţină toate cuvintele posibile pentru o anumită sarcină de recunoaştere a vorbirii alături, bineînţeles, de transcrierile fonetice ale acestor cuvinte. Pentru mai multe informaţii despre rolul dicţionarului fonetic în cadrul unui sistem de recunoaştere a vorbirii continue consultaţi bibliografia îndrumarului de proiect.

Pentru sarcina curentă (recunoaşterea secvenţelor audio ce conţin cifre) dicţionarul fonetic trebuie să conţină numai cele zece cifre ale sistemului zecimal: zero, unu, doi, trei, patru, cinci, şase, şapte, opt şi nouă. Sistemul de recunoaştere a vorbirii continue CMU Sphinx utilizează dicţionare fonetice în format text, ce conţin câte un cuvânt pe fiecare linie a fişierului. Transcrierea fonetică a fiecărui cuvânt trebuie să apară pe aceeaşi linie. Fonemele trebuie separate de câte un spaţiu. Crearea dicţionarului fonetic în formatul agreeat este descrisă mai jos.

Logaţi-vă la serverul de dezvoltare, apoi creaţi un director de lucru temporar:

```
mkdir phonetics
cd phonetics
```

Creaţi şi editaţi un nou fişier numit rodigits.dic utilizând editorul vim:

```
vi rodigits.dic
```

Utilizând Tabelul 1 editaţi acest fişier astfel încât să aveţi pe fiecare linie câte o cifră alături de transcrierea ei fonetică. Câteva instrucţiuni despre modul de editare al unui fişier text utilizând editorul vim sunt prezentate aici. În final fişierul ar trebui să arate în felul următor:

```
zero z e r o
unu u n u
...
nouă n o1 w a1
```

**Notă:** în cazul în care nu puteţi tasta caractere cu diacritice utilizând sistemul de operare şi tastatura calculatorului pe care lucraţi, puteţi copia denumirile cifrelor ce conţin diacritice din lista următoare: şase, şapte, nouă. În consolă se poate da paste utilizând combinaţia de taste ctrl+alt+V.

| Fonemul | | | Exemple de cuvinte | |
|---|---|---|---|---|
| **Nr.** | **Simbol IPA** | **Simbol intern** | **Forma scrisă** | **Forma fonetică** |
| 1 | a | a | s**a**t | s a t |
| 2 | ə | a1 | gur**ă** | g u r a1 |
| 3 | e | e | mar**e** | m a r e |
| 4 | i | i | l**i**ft | l i f t |
| 5 | j | i1 | tar**i** | t a r i1 |
| 6 | i̯ | i2 | **î**ntre | i2 n t r e |
| 7 | o | o | l**o**c | l o c |
| 8 | u | u | ş**u**t | s1 u t |
| 9 | y | y | ecr**u** | e c r y |
| 10 | ø | o2 | bl**eu** | b l o2 |
| 11 | e̯ | e1 | d**e**al | d e1 a l |
| 12 | j | i3 | f**i**ară | f i3 a r a1 |
| 13 | o̯ | o1 | **o**ase | o1 a s e |
| 14 | w | w | sa**u** | s a w |
| 15 | c | k2 | **ch**em | k2 e m |
| 16 | b | b | **b**ar | b a r |
| 17 | p | p | **p**ar | p a r |
| 18 | k | k | a**c**um | a k u m |
| 19 | ʧ | k1 | **c**enuşă | k1 e n u s1 a1 |
| 20 | g | g | **g**alben | g a l b e n |
| 21 | ʤ | g1 | **g**irafă | g1 i r a f a1 |
| 22 | ɟ | g2 | un**gh**i | u n g2 |
| 23 | d | d | **d**ar | d a r |
| 24 | t | t | **t**ot | t o t |
| 25 | f | f | **f**aţa | f a t1 a |
| 26 | v | v | **v**apor | v a p o r |
| 27 | h | h | **h**arta | h a r t a |
| 28 | ʒ | j | a**j**utor | a j u t o r |
| 29 | ʃ | s1 | co**ş** | k o s1 |
| 30 | l | l | **l**ac | l a c |
| 31 | m | m | **m**ăr | m a1 r |
| 32 | n | n | **n**as | n a s |
| 33 | s | s | **s**are | s a r e |
| 34 | z | z | **z**ar | z a r |
| 35 | r | r | **r**isc | r i s k |
| 36 | ʦ | t1 | **ţ**ăran | t1 a1 r a n |

**Tabelul 1. Fonemele limbii române**

În cazul sistemelor de recunoaştere a vorbirii cu vocabular redus (cum este şi cel pe care tocmai îl construiţi) manualul CMU Sphinx recomandă utilizarea de foneme specifice fiecărui cuvânt. Altfel spus ni se recomandă să modelăm diferit un acelaşi fonem X în funcţie de cuvântul (şi de poziţia în cuvânt) în care apare acesta. În consecinţă veţi modifica în continuare fişierul rodigits.dic adaugând în numele fiecărui fonem numele cuvântului din care face parte, cât şi poziţia fonemului în cuvânt (numai acolo unde este cazul). După această operaţie dicţionarul fonetic ar trebui să arate astfel:

```
zero z_zero e_zero r_zero o_zero
unu u_unu1 n_unu u_unu2
...
nouă n_nouă o1_nouă w_nouă a1_nouă
```

**Notă:** în cazul în care nu puteţi tasta caractere cu diacritice utilizând sistemul de operare şi tastatura calculatorului pe care lucraţi, puteţi copia denumirile cifrelor ce conţin diacritice din lista următoare: şase, şapte, nouă. În consolă se poate da paste utilizând combinaţia de taste ctrl+alt+V.

În continuare trebuie să creăm lista tuturor fonemelor ce vor fi modelate de modelul acustic. Este necesar sa scriem un fişier text care să conţină câte un fonem pe fiecare linie, iar liniile trebuie să fie sortate în ordine alfabetică. Crearea acestui fişier se poate face manual (parcurgând dicţionarul fonetic creat anterior) sau automat (urmând intrucţiunile de mai jos).

Creaţi un nou fişier (rodigits.phones.temp) care să conţină toate cuvintele şi toate fonemele din dicţionarul fonetic (câte un cuvânt/fonem pe linie):

```
sed 's/ /\n/g' rodigits.dic > rodigits.phones.temp
```

Din această listă selectaţi numai fonemele (profitând de faptul că ele conţin caracterul "_") şi listaţi-le într-un nou fişier numit rodigits.phones.temp.onlyPhones:

```
grep "_" rodigits.phones.temp > rodigits.phones.temp.onlyPhones
```

Adăugaţi la sfârşitul acestui fişier "fonemul" SIL (acest "fonem" este utilizat pentru a modela zonele de linişte):

```
echo "SIL" >> rodigits.phones.temp.onlyPhones
```

Sortaţi fonemele în ordine alfabetică, redenumiţi fişierul rezultat şi ştergeţi toate celelalte fişiere temporare:

```
sort -u rodigits.phones.temp.onlyPhones > rodigits.phones.temp.onlyPhones.sorted
mv rodigits.phones.temp.onlyPhones.sorted rodigits.phones
rm rodigits.phones.temp*
```

# IV. Antrenarea modelului acustic

Antrenarea modelului acustic se va face în cadrul unui proiect CMU Sphinx şi, aşa cum am precizat în prima secţiune, presupune existenţa următoarelor resurse:

- un set de clipuri audio ce conţin vorbire (set pe care l-ati înregistrat în prealabil),

- transcrierea textuală corespunzătoare cuvintelor pronunţate în clipurile audio (deja existentă),
- un dicţionar fonetic cuprinzând toate cuvintele (dicţionar pe care l-ati creat în prealabil),
- un dicţionar cu elemente acustice care nu sunt foneme (linişte, tuse, râs, muzică, etc.);

## Crearea unui proiect CMU Sphinx numit rodigits

Logaţi-vă pe serverul de dezvoltare şi creaţi un director numit projects în care veţi crea ulterior toate proiectele:

```
cd ~
mkdir projects
```

Intraţi in directorul projects şi creaţi un nou director numit rodigits pentru proiectul curent. Apoi intraţi în directorul rodigits:

```
cd ~/projects
mkdir rodigits
cd ~/projects/rodigits
```

Configuraţi directorul rodigits pentru a putea fi utilizat ca proiect CMU Sphinx:

```
sphinxtrain_biosinf -t rodigits setup
mkdir logdir
```

## Adăugarea resurselor necesare în directorul proiectului

Logaţi-vă pe serverul de dezvoltare şi intraţi în directorul proiectului:

```
cd ~/projects/rodigits
```

Copiaţi clipurile audio înregistrate în prealabil în directorul wav (important: modificaţi comanda următoare pentru a include propriul SPEAKER_ID, aşa cum vi l-a specificat îndrumătorul de proiect):

```
mkdir wav
cp -r /home/biosinfShare/resources/wav/SPEAKER_ID/ ~/projects/rodigits/wav/
```

Copiaţi transcrierea textuală a cuvintelor pronunţate în clipurile audio în directorul etc:

```
cp /home/biosinfShare/resources/transcription/rodigits.data ~/projects/rodigits/etc/
```

Copiaţi dicţionarul fonetic şi lista de foneme create în prealabil în directorul etc:

```
cp ~/phonetics/rodigits.dic ~/projects/rodigits/etc/
cp ~/phonetics/rodigits.phones ~/projects/rodigits/etc/
```

Copiaţi dicţionarul cu elemente acustice care nu sunt foneme în directorul etc:

```
cp /home/biosinfShare/resources/phonetic/rodigits.filler ~/projects/rodigits/etc/
```

Copiaţi scripturile createFileIds.sh şi createTranscriptions.sh în directorul etc:

```
cp -r /home/biosinfShare/scripts/createFileIds.sh ~/projects/rodigits/etc/
cp -r /home/biosinfShare/scripts/createTranscriptions.sh ~/projects/rodigits/etc/
```

## Transformarea fişierelor necesare procesului de antrenare în formatul corespunzător

Înainte de a putea începe antrenarea modelului acustic, trebuie creat un fişier cu lista clipurilor audio ce vor fi folosite la antrenare şi un alt fişier cu transcrierea textuală a respectivelor clipuri audio. Din motive de eficienţă vom crea tot acum şi lista clipurilor audio ce vor fi folosite la evaluare, cât şi fişierul cu transcrierea textuală a acestora.

Logaţi-vă pe serverul de dezvoltare şi intraţi în directorul proiectului, în subdirectorul etc:

```
cd ~/projects/rodigits/etc/
```

Vizualizaţi scriptul createFileIds.sh utilizând editorul vim:

```
vi createFileIds.sh
```

Intraţi în modul de editare (tastând “i”) şi schimbaţi valoarea variabilei SPEAKER_ID pentru a utiliza propriul SPEAKER_ID, aşa cum vi l-a specificat îndrumătorul de proiect. Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând “:w” şi apoi Enter) şi părăsiţi editorul (tastând “:q” şi apoi Enter).

Creaţi lista tuturor clipurilor audio înregistrate:

```
./createFileIds.sh > rodigits.fileids.all
```

Creaţi lista clipurilor audio ce vor fi utilizate pentru antrenare selectând primele 50 şi ultimele 30 de clipuri audio din lista tuturor clipurilor (rodigits.fileids.all):

```
head -50 rodigits.fileids.all > rodigits.fileids.train
tail -30 rodigits.fileids.all >> rodigits.fileids.train
```

Celelalte clipuri audio vor fi utilizate pentru evaluare. Creaţi un fişier cu lista acestora:

```
diff rodigits.fileids.train rodigits.fileids.all | grep '>' | sed 's/> //' > rodigits.fileids.test
```

Vizualizaţi scriptul createTranscriptions.sh utilizând editorul vim:

```
vi createTranscriptions.sh
```

Intraţi în modul de editare (tastând “i”) şi schimbaţi valoarea variabilei SPEAKER_ID pentru a utiliza propriul SPEAKER_ID, aşa cum vi l-a specificat îndrumătorul de proiect. Ieşiţi din modul de editare

(tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi părăsiţi editorul (tastând ":q" şi apoi Enter).

Creaţi transcrierea textuală (în formatul corespunzător cerut de CMU Sphinx) pentru toate clipurile audio:

```
./createTranscriptions.sh > rodigits.transcription.all
```

Creaţi un fişier care să conţină numai transcrierea textuală pentru clipurile audio ce vor utilizate pentru antrenare selectând primele 50 şi ultimele 30 de linii din transcrierea textuală:

```
head -50 rodigits.transcription.all > rodigits.transcription.train
tail -30 rodigits.transcription.all >> rodigits.transcription.train
```

Celelalte linii din transcrierea textuală vor fi utilizate pentru evaluare. Creaţi un fişier cu lista acestora:

```
diff rodigits.transcription.train rodigits.transcription.all | grep '>' | sed 's/> //' >
rodigits.transcription.test
```

## Configurarea procesului de antrenare

Orice proiect de recunoaştere de vorbire CMU Sphinx are un fişier de configurare se specifică parametrii modelului acustic ce va fi antrenat şi locaţia resurselor necesare în procesul de antrenare. Pentru primul proiect pe care îl veţi face nu veţi pastra valorile implicite ale parametrilor modelului acustic, specificând numai locaţia resurselor.

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului, în subdirectorul etc şi vizualizaţi fişierul de configurare al proiectului cu ajutorul editorului vim:

```
cd ~/projects/rodigits/etc/
vi sphinx_train.cfg
```

Identificaţi în acest fişier liniile care specifică locaţia resurselor utilizate în procesul de antrenare (dicţionarul fonetic, lista de foneme, dicţionarul de sunete care nu sunt foneme, lista de clipuri audio de antrenare şi transcrierea textuală a acestor clipuri audio):

```
$CFG_DICTIONARY     = "$CFG_LIST_DIR/$CFG_DB_NAME.dic";
$CFG_RAWPHONEFILE   = "$CFG_LIST_DIR/$CFG_DB_NAME.phone";
$CFG_FILLERDICT     = "$CFG_LIST_DIR/$CFG_DB_NAME.filler";
$CFG_LISTOFFILES    = "$CFG_LIST_DIR/${CFG_DB_NAME}_train.fileids";
$CFG_TRANSCRIPTFILE = "$CFG_LIST_DIR/${CFG_DB_NAME}_train.transcription";
```

Intraţi în modul de editare (tastând "i") şi modificaţi aceste linii astfel:

```
$CFG_DICTIONARY    = "$CFG_LIST_DIR/rodigits.dic";
$CFG_RAWPHONEFILE  = "$CFG_LIST_DIR/rodigits.phones";
$CFG_FILLERDICT    = "$CFG_LIST_DIR/rodigits.filler";
$CFG_LISTOFFILES   = "$CFG_LIST_DIR/rodigits.fileids.train";
$CFG_TRANSCRIPTFILE = "$CFG_LIST_DIR/rodigits.transcription.train";
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter).

## Crearea fişierelor cu coeficienţi MFCC corespunzătoare clipurilor audio de antrenare

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului şi creaţi fişierele cu coeficienţi cepstrali executând următoarea comandă:

```
cd ~/projects/rodigits/
/usr/local/sphinx/lib/sphinxtrain/scripts/000.comp_feat/slave_feat.pl
```

**Notă:** este normal ca în urma execuţiei comenzii de mai sus să apară eroarea Failed to open control ... : No such file or directory at .../make_feats.pl line 89. Ea ne atenţionează asupra faptului că lista de fişiere audio de evaluare nu a fost găsită (acest lucru este normal, pentru că această listă nu a fost încă generată).

Execuţia acestei comenzi are ca efect crearea a 80 de fişiere cu coeficienţi cepstrali corepunzătoare celor 80 de clipuri audio de antrenare. Verificaţi faptul că fişierele cu coeficienţi cepstrali au fost create corect (important: modificaţi comenzile următoare pentru a include propriul SPEAKER_ID, aşa cum vi l-a specificat îndrumătorul de proiect):

```
ls feat/SPEAKER_ID/*
ls feat/SPEAKER_ID/* | wc -l
ls wav/SPEAKER_ID/* | wc -l
```

Prima comandă din grupul celor de mai sus va afişa toate fişierele din subdirectorul feat/SPEAKER_ID. A doua comandă va afişa numărul de fişiere din respectivul subdirector (rezultatul ar trebui să fie 80 pentru că doar 80 de clipuri audio se folosesc pentru antrenare). A treia comandă va afişa numarul de fişiere din subdirectorul wav/SPEAKER_ID (rezultatul ar trebui să fie 100 pentru că în total ar trebui să aveţi 100 de clipuri audio înregistrate).

## Antrenarea modelului acustic

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului şi porniţi antrenarea modelului acustic executând următoarea comandă:

```
cd ~/projects/rodigits/
sphinxtrain_biosinf run
```

Aşteptaţi terminarea procesului de antranare (ar trebui să dureze aproximativ 2 minute). În cazul în care apar erori verificaţi faptul că fişierele cu resurse sunt în formatul corespunzător şi că nu aţi

modificat greşit fişierul de configurare. În cazul în care nu reuşiţi să identificaţi şi să rezolvaţi problema discutaţi cu îndrumătorul de proiect.

# V. Crearea modelului de limbă (gramaticii)

Sistemele de recunoaştere de vorbire cu vocabular mare de ultimă generaţie utilizează modele de limbă statistice de tip n-gram. Aceste modele de limbă se construiesc pe baza unor corpusuri mari de text specific sarcinii de recunoaştere, estimând probabilităţile de apariţie ale cuvintelor şi secvenţelor de cuvinte specifice respectivei sarcini. Modelele de limbă de tip n-gram se utilizează apoi în procesul de decodare (recunoaştere a vorbirii) pentru a calcula probabilităţile secvenţelor de cuvinte propuse de modelul acustic. Pentru mai multe informaţii despre modele de limbă statistice şi despre rolul modelului de limbă în cadrul unui sistem de recunoaştere a vorbirii continue consultaţi bibliografia îndrumarului de proiect.

Sarcina de recunoaştere a secvenţelor audio ce conţin cifre este o sarcină de recunoaştere cu vocabular foarte redus pentru care nu se pretează modelele de limbă statistice. Mai mult, cifrele şi succesiunile de cifre din cadrul clipurilor audio înregistrate apar cu probabilităţi aproximativ egale (nu se poate afirma că o cifră este pronunţată sistematic mai des decât alta). În aceste condiţii, modelele de limbă de tip gramatică cu stări finite (FSG – finite state grammar) sunt mult mai potrivite. O gramatică cu stări finite este un model de tip graf în care nodurile reprezintă cuvinte ale limbii, iar tranziţiile între cuvinte sunt reprezentate de arcele grafului. Un astfel de model de limbă specifică în mod explicit toate secvenţele de cuvinte permise de gramatica sarcinii de recunoaştere. Mai mult, fiecărui arc în parte îi poate fi asignat un cost specificând astfel probabilitatea ca un cuvânt să fie precedat de un altul (sau altfel spus probabilitatea respectivei secvenţe de două cuvinte).

În Figura 2 este reprezentată grafic gramatica cu stări finite a sarcinii noastre de recunoaştere. Se poate observa că modelul este format din 14 noduri, dintre care numai zece reprezintă cuvinte ale limbii (cifrele de la zero la nouă), iar celelalte patru sunt utilizate pentru a realiza „intrarea”, respectiv „ieşirea” din graf şi pentru tranziţia înapoi. Tranziţiile arată modul în care se poate parcurge această gramatică şi implicit secvenţele de cuvinte permise: în fiecare clip audio se pot pronunţa una sau mai multe cifre.

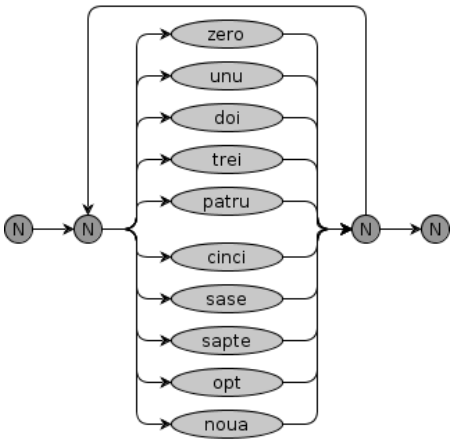


**Figura 2. Gramatica cu stări finite a sarcinii de recunoaştere rodigits**

Implementarea unei astfel de gramatici FSG poate fi făcută foarte simplu utilizând formatul Java Speech Gramar (JSGF) aşa cum vom vedea în continuare.

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului, în subdirectorul etc şi creaţi fişierul rodigits.jsgf utilizând editorul vim:

```
cd ~/projects/rodigits/etc/
vi rodigits.jsgf
```

Intraţi în modul de editare (tastând "i") şi scrieţi următoarele trei linii:

```
#JSGF V1.0;
grammar rodigits;
public <numbers> = (zero | unu | doi | trei | patru | cinci | şase | şapte | opt | nouă) * ;
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter). Prima linie pe care aţi scris-o în fişier specifică numele gramaticii, iar cea de-a doua linie specifică cuvintele şi succesiunile de cuvinte permise astfel: numbers este *zero* sau *unu* sau ... *nouă* de oricâte ori. Sau-ul este specificat prin carecterul |, iar faptul că cifrele se pot repeta de oricâte ori este specificat de caracterul *. Pentru mai multe detalii despre formatul Java Speech Grammar şi modul în care puteţi crea şi alte tipuri de gramatici cu stări finite consultaţi site-ul JSGF precizat în bibliografia îndrumarului de proiect.

Transformaţi (cu ajutorul utilitarului sphinx_jsgf2fsg) gramatica sarcinii de recunoaştere din format JSGF (Java Speech Grammar Format) în formatul intern FSG (Finite State Grammar) utilizat de CMU Sphinx:

```
sphinx_jsgf2fsg -jsgf rodigits.jsgf -fsg rodigits.fsg
```

Vizualizaţi fişierul rezultat (rodigits.fsg) utilizând editorul vim:

```
vi rodigits.fsg
```

Observaţi modul în care se specifică nodurile, tranziţiile şi probabilităţile de tranziţie ale gramaticii cu stări finite în formatul intern uztilizat CMU Sphinx. Ca exerciţiu, desenaţi pe o bucată de hârtie graful definit in acest fişier şi comparaţi-l cu graful prezentat în Figura 2.

# VI. Evaluarea sistemului de recunoaştere şi interpretarea rezultatelor

Cele trei componente fundamentale ale unui sistem de recunoaştere a vorbirii (modelul acustic, modelul de limbă şi modelul fonetic) sunt în acest moment disponibile. În consecinţă se poate trece la decodarea clipurilor audio de evaluare şi apoi la compararea transcrierii textuale rezultate cu transcrierea textuală de referinţă în vederea evaluării sistemului de recunoaştere de secvenţe audio ce conţin cifre.

## Configurarea procesului de decodare (şi evaluare)

Pentru a configura procesul de decodare vom edita acelaşi fişier de configurare care a fost modificat şi pentru antrenare (sphinx_train.cfg). De această dată nu ne vom concentra asupra parametrilor de antranare, ci asupra parametrilor de decodare.

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului, în subdirectorul etc şi vizualizaţi fişierul de configurare al proiectului cu ajutorul editorului vim:

```
cd ~/projects/rodigits/etc/
vi sphinx_train.cfg
```

Identificaţi în acest fişier linia care specifică numele scriptului utilizat pentru decodare:

```
$DEC_CFG_SCRIPT = 'psdecode.pl';
```

Intraţi în modul de editare (tastând “i”) şi modificaţi acestă linie astfel:

```
$DEC_CFG_SCRIPT = 'psdecode_fsg.pl';
```

Identificaţi apoi liniile care specifică numele şi locaţia resurselor utilizate în procesul de decodare (dicţionarul fonetic, dicţionarul de sunete care nu sunt foneme, lista de clipuri audio de evaluare şi transcrierea textuală a acestor clipuri audio) şi numele subdirectorului în care se vor salva rezultatele:

```
$DEC_CFG_DICTIONARY    = "$DEC_CFG_BASE_DIR/etc/$DEC_CFG_DB_NAME.dic";
$DEC_CFG_FILLERDICT    = "$DEC_CFG_BASE_DIR/etc/$DEC_CFG_DB_NAME.filler";
$DEC_CFG_LISTOFFILES   = "$DEC_CFG_BASE_DIR/etc/${DEC_CFG_DB_NAME}_test.fileids";
$DEC_CFG_TRANSCRIPTFILE =
"$DEC_CFG_BASE_DIR/etc/${DEC_CFG_DB_NAME}_test.transcription";
$DEC_CFG_RESULT_DIR    = "$DEC_CFG_BASE_DIR/result";
```

Modificaţi aceste linii astfel:

```
$DEC_CFG_DICTIONARY    = "$DEC_CFG_BASE_DIR/etc/rodigits.dic";
$DEC_CFG_FILLERDICT    = "$DEC_CFG_BASE_DIR/etc/rodigits.filler";
$DEC_CFG_LISTOFFILES   = "$DEC_CFG_BASE_DIR/etc/rodigits.fileids.test";
$DEC_CFG_TRANSCRIPTFILE = "$DEC_CFG_BASE_DIR/etc/rodigits.transcription.test";
$DEC_CFG_RESULT_DIR    = "$DEC_CFG_BASE_DIR/result.cd_cont_200_8";
```

Identificaţi apoi linia care specifică numele modelului de limbă ce va fi utilizat la decodare:

```
$DEC_CFG_LANGUAGEMODEL  =
"$DEC_CFG_LANGUAGEMODEL_DIR/${DEC_CFG_DB_NAME}.lm.DMP";
```

Şi modificaţi-o astfel:

```
$DEC_CFG_LANGUAGEMODEL  = "$DEC_CFG_LANGUAGEMODEL_DIR/rodigits.fsg";
```

Identificaţi apoi linia care specifică aplicaţia cu care se va face alinierea textului rezultat în urma recunoaşterii cu textul de referinţă (în vederea evaluării ratei de eroare la nivel de cuvânt):

```
$DEC_CFG_ALIGN = "builtin";
```

Şi modificaţi-o astfel:

```
$DEC_CFG_ALIGN = "sclite";
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter).

## Crearea fişierelor cu coeficienţi MFCC corespunzătoare clipurilor audio de evaluare

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului şi creaţi fişierele cu coeficienţi cepstrali executând următoarea comandă:

```
cd ~/projects/rodigits/
/usr/local/sphinx/lib/sphinxtrain/scripts/000.comp_feat/slave_feat.pl
```

Execuţia acestui script are ca efect crearea a 100 de fişiere cu coeficienţi cepstrali corepunzătoare tuturor clipurilor audio înregistrate (atât cele utilizate pentru antrenare, cât şi cele utilizate pentru evaluare). Verificaţi faptul că fişierele cu coeficienţi cepstrali au fost create corect (important: modificaţi comenzile următoare pentru a include propriul SPEAKER_ID, aşa cum vi l-a specificat îndrumătorul de proiect):

```
ls feat/SPEAKER_ID/*
ls feat/SPEAKER_ID/* | wc -l
ls wav/SPEAKER_ID/* | wc -l
```

Prima comandă din grupul celor de mai sus va afişa toate fişierele din subdirectorul feat/SPEAKER_ID. A doua comandă va afişa numărul de fişiere din respectivul subdirector (rezultatul

ar trebui să fie 100 pentru că în total sunt 100 de clipuri audio înregistrate). A treia comandă va afişa numarul de fişiere din subdirectorul wav/SPEAKER_ID (rezultatul ar trebui să fie tot 100).

## Decodarea clipurilor audio de evaluare

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului şi porniţi procesul de evaluare executând următoarea comandă:

```
cd ~/projects/rodigits/
/usr/local/sphinx/lib/sphinxtrain/scripts/decode/slave.pl
```

Aşteptaţi terminarea procesului de decodare (ar trebui să dureze aproximativ 1 minut). În cazul în care apar erori încercaţi să identificaţi problema vizualizand fişierul de log al procesului de decodare:

```
vi logdir/decode/rodigits-1-1.log
```

În cazul în care nu reuşiţi să identificaţi şi să rezolvaţi problema discutaţi cu îndrumătorul de proiect. În cazul în care procesul de decodare se încheie cu succes ar trebui ca pe ecran să vă apară un mesaj similar cu acesta:

```
MODULE: DECODE Decoding using models previously trained (decode script: psdecode_fsg.pl)
    Decoding 20 segments starting at 0 (part 1 of 1)
    0%
    Aligning results to find error rate
    SENTENCE ERROR: 50.0% (10/20)   WORD ERROR RATE: 12.9% (30/240)
```

## Vizualizarea şi interpretarea rezultatelor evaluării

Evaluarea unui sistem de recunoaştere a vorbirii se face comparând în mod automat cele două transcrieri textuale ale clipurilor audio de evaluare: transcrierea textuală de referinţă (cea despre care se ştie a priori că este corectă) şi transcrierea textuală ipotetică (cea rezultată în urma procesului de decodare). Analiza se face frază cu frază, în două etape: a) fraza ipotetică şi fraza de referinţă sunt aliniate printr-un algoritm care urmăreşte să minimizeze numărul de erori de transcriere, după care b) se numără toate erorile de recunoaştere la nivel de cuvânt (cuvinte inserate, cuvinte susbtituite, respectiv cuvinte şterse). Se disting astfel două criterii de performanţă standard folosite în evaluarea sistemelor de recunoaştere a vorbirii continue: rata de eroare la nivel de propoziţie (SER - sentence error rate) şi rata de eroare la nivel de cuvânt (WER – word error rate). Rata de eroare la nivel de propoziţie se calculează ca raport între numărul de propoziţii corecte (propoziţii ce nu conţin nicio greşeală) şi numărul total de propoziţii. Rata de eroare la nvel de cuvânt se calculează cu formula:

$$WER[\%] = \frac{\#Erori\ de\ inserţii + \#Erori\ de\ substituţie + \#Erori\ de\ ştergere}{\#Cuvinte\ în\ transcrierea\ de\ referinţă} \times 100$$

Aşa cum se observă din mesajul afişat în urma execuţiei procesului de decodare (dat ca şi exemplu mai sus), în urma evaluării transcrierii ipotetice s-a obţinut o rată de eroare la nivel de propoziţie

de 50% şi o rată de eroare la nivel de cuvânt de 12.9%. Mai multe informaţii despre rezultatele evaluării sistemului de recunoaştere puteţi afla vizualizând întreg raportul de evaluare:

```
cd ~/projects/rodigits/
vi result.cd_cont_200_8/rodigits.align
```

Primul tabel din raportul de evaluare afişează pe câte o linie sumarul rezultatelor pentru diverşii vorbitori din baza de date de evaluare. În cazul de faţă aveţi un singur vorbitor în baza de date, deci o singură linie în tabel. Pe coloane aveţi mai multe informaţii: id-ul vorbitorului, numărul de propoziţii de evaluare, numărul de cuvinte de evaluare, procentul de cuvinte recunoscute corecte, procentul de cuvinte substituite/şterse/inserate, rata de eroare la nivel de cuvânt şi rata de eroare la nivel de propoziţie.

La sectiunea *Sentence Recognition Performance* se oferă mai multe detalii cu privire la erorile la nivel de propoziţie, iar la secţiunea *Word Recognition Performance* se oferă mai multe detalii cu privire la erorile la nivel de cuvânt.

Urmează alte cinci secţiuni foarte interesante care prezintă statistici despre perechile de cuvinte confundate cel mai frecvent (Confusion Pairs), cuvintele inserate cel mai frecvent (*Insetions*), cuvintele şterse cel mai frecvent (*Deletions*), cuvintele substituite cel mai frecvent (*Substitutions*) şi cuvintele recunoscute fals cel mai frecvent (*Falsely Recognized*).

În final, raportul de evaluare prezintă toate frazele ipotetice aliniate la frazele de referinţă corespunzătoare specificând, de asemenea, şi tipurile de erori apărute la fiecare frază în parte.

Toate aceste informaţii şi rezultate din raportul de evaluare al sistemului de recunoaştere vor fi comparate cu rezultatele ce vor fi obţinute în secţiunea următoare şi vor fi în final prezentate şi discutate în raportul final al proiectului de cercetare-dezvoltare.

# VII. Optimizarea modelului acustic

Înainte de a putea face câteva optimizări simple ale modelului acustic trebuie să studiaţi şi să înţelegeţi câteva aspecte importante privind modul de construcţie al unui astfel de model:

- arhitectura generala a modelului acustic (platforma HMM-GMM). Informaţii despre acest subiect găsiţi în bibliografia îndrumarului de proiect: [3] pag. 31, [4] pag. 23, [5] pag. 8, [6] pag. 306.
- alegerea unităţilor de vorbire (foneme, trifoneme, senone) şi dependenţa/independenţa de context. Informaţii despre aceste subiecte găsiţi în bibliografia îndrumarului de proiect: [3] pag. 33, [5] pag. 52, [6] pag. 310.

Utilizând deja noţiunile amintite mai sus (şi explicate pe larg în bibliografie), vom sumariza în continuare modul particular în care toolkitul de recunoaştere a vobirii pe care îl folosim (CMU Sphinx) realizează antrenarea modelului acustic. În mod implicit modelul acustic creat de CMU Sphinx este construit cu unităţi de vorbire dependente de context de tip trifonem (fonem căruia i se precizează vecinii stânga-dreapta). Tot în mod implicit aceste trifoneme sunt implementate ca HMM-uri cu trei stări emisive, fiecare având o distribuţie de probabilitate de ieşire impementată cu

un GMM. Fiecare astfel de stare-model se numeste *senone* şi ea poate fi comună mai multor trifoneme (în funcţie de similarităţile dintre ele: de exemplu trifonemele p-a-r şi t-a-v pot avea senone-ul iniţial comun). În mod implicit numărul de senone este configurat la 200, iar numărul de densităţi de probabilitate pentru fiecare GMM este configurat la 8.

În consecinţă modelul acustic antrenat în prealabil modelează cele câteva foneme specificate în dicţionarul fonetic utilizând modele fonetice dependente de context având în total 200 de senone, fiecare senone folosind câte 8 densităţi Gaussiene de probabilitate pentru a modela parametri acustici de ieşire. Procesul de antrenare implementat de CMU Sphinx conduce la obţinerea acestui model acustic astfel:

1. iniţializarea modelelor fonetice independente de context (câte un model cu câte trei senone folosind câte o singură densitate Gaussiană pentru fiecare *fonem* în parte)
2. antrenarea modelelor fonetice (independente de context) rezultate la pasul anterior
3. transformarea modelelor fonetice independente de context în modele fonetice dependente de context (se obţine câte un model cu câte trei senone folosind câte o singură densitate Gaussiană pentru fiecare *trifonem* în parte; în total modelul acustic poate avea *oricâte senone*!)
4. antrenarea modelelor fonetice (dependente de context) rezultate la pasul anterior
5. legarea stărilor (senonelor) modelelor fonetice dependente de context pe baza similarităţilor dintre ele (se obţine câte un model cu câte trei senone folosind câte o singură densitate Gaussiană pentru fiecare trifonem în parte; în total modelul acustic va avea *200 senone*!)
6. antrenarea modelelor fonetice (dependente de context) rezultate la pasul anterior
7. dublarea numărului de densităţi Gaussiene în cadrul fiecărei senone (se obţine câte un model cu câte trei senone folosind câte *două densităţi Gaussiane* pentru fiecare trifonem în parte; în total modelul acustic va avea 200 senone!)
8. antrenarea modelelor fonetice (dependente de context) rezultate la pasul anterior
9. dublarea numărului de densităţi Gaussiene în cadrul fiecărei senone (se obţine câte un model cu câte trei senone folosind câte *patru densităţi Gaussiane* pentru fiecare trifonem în parte; în total modelul acustic va avea 200 senone!)
10. antrenarea modelelor fonetice (dependente de context) rezultate la pasul anterior
11. dublarea numărului de densităţi Gaussiene în cadrul fiecărei senone (se obţine câte un model cu câte trei senone folosind câte *opt densităţi Gaussiane* pentru fiecare trifonem în parte; în total modelul acustic va avea 200 senone!)

În consecinţă, pentru a obţine modelul acustic cu modele fonetice dependente de context având 200 de senone şi 8 densităţi Gaussiene per senone s-a trecut şi prin etapele în care modelul acustic era format din:

- modele fonetice independente de context (după pasul 2)
- modele fonetice dependente de context cu o singură densitate Gaussiană per senone (după pasul 6)
- modele fonetice dependente de context cu 2 densităţi Gaussiane per senone (după pasul 8)
- modele fonetice dependente de context cu 4 densităţi Gaussiane per senone (după pasul 10)

Toate aceste modele se află în directorul proiectului, în subdirectorul model_parameters. Puteţi verifica acest lucru astfel:

```
cd ~/projects/rodigits
ls -all model_parameters
```

Lista subdirectoarelor afişate ar trebui să fie următoarea:

```
rodigits.cd_cont_200
rodigits.cd_cont_200_1
rodigits.cd_cont_200_2
rodigits.cd_cont_200_4
rodigits.cd_cont_initial
rodigits.cd_cont_untied
rodigits.ci_cont
rodigits.ci_cont_flatinitial
```

În mod implicit decodarea realizată în capitolul VI a utilizat modelele din subdirectorul rodigits.cd_cont_200 (modelele fonetice dependente de context cu 8 densităţi Gaussiane per senone). În continuare ne propunem să configurăm procesul de decodare pentru a utiliza pe rând modelele din subdirectoarele:

- rodigits.ci_cont (modelele fonetice independente de context)
- rodigits.cd_cont_200_1 (modelele fonetice dependente de context cu o densitate Gaussiană per senone)
- rodigits.cd_cont_200_2 (modelele fonetice dependente de context cu 2 densităţi Gaussiene per senone)
- rodigits.cd_cont_200_4 (modelele fonetice dependente de context cu 4 densităţi Gaussiene per senone)

## Decodarea clipurilor audio de evaluare folosind modele fonetice independente de context

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului, în subdirectorul etc şi vizualizaţi fişierul de configurare al proiectului cu ajutorul editorului vim:

```
cd ~/projects/rodigits/etc/
vi sphinx_train.cfg
```

Identificaţi în acest fişier linia care specifică modelele acustice utilizate pentru decodare:

```
# Models to use.
$DEC_CFG_MODEL_NAME = "$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}";
```

Intraţi în modul de editare (tastând “i”), copiaţi linia respectivă şi comentaţi una dintre copii (pentru a păstra un back-up) şi modificaţi cealaltă copie astfel încât să specificaţi locaţia modelelor fonetice independente de context. Această zonă a fişierului ar trebui să arate în final ca mai jos:

```
# Models to use.
#$DEC_CFG_MODEL_NAME =
"$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}";
$DEC_CFG_MODEL_NAME = "$CFG_EXPTNAME.ci_${CFG_DIRLABEL}";
```

Identificaţi apoi linia care specifică numele subdirectorului în care se vor salva rezultatele:

```
$DEC_CFG_RESULT_DIR     = "$DEC_CFG_BASE_DIR/result.cd_cont_200_8";
```

Şi modificaţi-o astfel încât numele subdirectorului în care se vor salva rezultatele să fie sugestiv:

```
$DEC_CFG_RESULT_DIR     = "$DEC_CFG_BASE_DIR/result.ci_cont";
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter).

Intraţi apoi în directorul proiectului şi porniţi procesul de evaluare executând următoarea comandă:

```
cd ~/projects/rodigits
/usr/local/sphinx/lib/sphinxtrain/scripts/decode/slave.pl
```

Aşteptaţi terminarea procesului de decodare (ar trebui să dureze aproximativ 1 minut). În cazul în care apar erori încercaţi să identificaţi problema vizualizand fişierul de log al procesului de decodare:

```
vi logdir/decode/rodigits-1-1.log
```

În cazul în care nu reuşiţi să identificaţi şi să rezolvaţi problema discutaţi cu îndrumătorul de proiect. În cazul în care procesul de decodare se încheie cu succes ar trebui ca pe ecran să vă apară un mesaj similar cu acesta:

```
MODULE: DECODE Decoding using models previously trained (decode script: psdecode_fsg.pl)
    Decoding 20 segments starting at 0 (part 1 of 1)
    0%
    Aligning results to find error rate
    SENTENCE ERROR: 55.0% (11/20)   WORD ERROR RATE: 6.3% (15/240)
```

Aceste rezultate, precum şi informaţiile şi rezultatele detaliate din raportul de evaluare generat la acest pas (result.ci_cont/rodigits.align) vor fi comparate cu rezultatele obţinute în prealabil şi cu rezultatele ce vor fi obţinute în continuare şi vor fi în final prezentate şi discutate în raportul final al proiectului de cercetare-dezvoltare.

## Decodarea clipurilor audio de evaluare folosind modele fonetice dependente de context cu mai puţine densităţi Gaussiene per senone

Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului, în subdirectorul etc şi vizualizaţi fişierul de configurare al proiectului cu ajutorul editorului vim:

```
cd ~/projects/rodigits/etc/
vi sphinx_train.cfg
```

Identificaţi în acest fişier linia care specifică modelele acustice utilizate pentru decodare:

```
# Models to use.
#$DEC_CFG_MODEL_NAME =
"$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}";
$DEC_CFG_MODEL_NAME = "$CFG_EXPTNAME.ci_${CFG_DIRLABEL}";
```

Intraţi în modul de editare (tastând "i"), ştergeţi linia decomentată (cea care specifica locaţia modelelor fonetice independente de context), apoi copiaţi linia comentată (pentru a păstra un back-up) şi decomentaţi una dintre copii. Modificaţi copia decomentată astfel încât să specificaţi locaţia modelelor fonetice dependente de context ce folosesc o singură densitate Gaussiană per senone. Această zonă a fişierului ar trebui să arate în final ca mai jos:

```
# Models to use.
#$DEC_CFG_MODEL_NAME =
"$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}";
$DEC_CFG_MODEL_NAME =
"$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}_1";
```

Identificaţi apoi linia care specifică numele subdirectorului în care se vor salva rezultatele:

```
$DEC_CFG_RESULT_DIR    = "$DEC_CFG_BASE_DIR/result.cd_cont_200_8";
```

Şi modificaţi-o astfel încât numele subdirectorului în care se vor salva rezultatele să fie sugestiv:

```
$DEC_CFG_RESULT_DIR    = "$DEC_CFG_BASE_DIR/result.cd_cont_200_1";
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter).

Copiaţi definiţia modelului acustic din subdirectorul ce conţine modelele fonetice dependente de context ce folosesc 8 densităţi Gaussiane per senone (rodigits.cd_cont_200) în subdirectorul ce conţine modelele fonetice dependente de context ce folosesc o singură densitate Gaussiană per senone (rodigits.cd_cont_200_1):

```
cd ~/projects/rodigits
cp model_parameters/rodigits.cd_cont_200/mdef model_parameters/rodigits.cd_cont_200_1/
```

Porniţi apoi procesul de evaluare executând următoarea comandă:

```
/usr/local/sphinx/lib/sphinxtrain/scripts/decode/slave.pl
```

Aşteptaţi terminarea procesului de decodare (ar trebui să dureze aproximativ 1 minut). În cazul în care apar erori încercaţi să identificaţi problema vizualizand fişierul de log al procesului de decodare:

```
vi logdir/decode/rodigits-1-1.log
```

În cazul în care nu reuşiţi să identificaţi şi să rezolvaţi problema discutaţi cu îndrumătorul de proiect. În cazul în care procesul de decodare se încheie cu succes ar trebui ca pe ecran să vă apară un mesaj similar cu acesta:

```
MODULE: DECODE Decoding using models previously trained (decode script: psdecode_fsg.pl)
    Decoding 20 segments starting at 0 (part 1 of 1)
    0%
    Aligning results to find error rate
    SENTENCE ERROR: 40.0% (8/20)   WORD ERROR RATE: 9.6% (23/240)
```

Aceste rezultate, precum şi informaţiile şi rezultatele detaliate din raportul de evaluare generat la acest pas (result.cd_cont_200_1/rodigits.align) vor fi comparate cu rezultatele obţinute în prealabil şi cu rezultatele ce vor fi obţinute în continuare şi vor fi în final prezentate şi discutate în raportul final al proiectului de cercetare-dezvoltare.

Repetaţi paşii de mai sus realizând decodarea clipurilor audio de evaluare cu modele fonetice dependente de context cu 2 şi respectiv 4 densităţi Gaussiene per senone.

## Antrenarea unui model acustic având în total 100 de senone

În continuare ne propunem să verificăm dacă numărul de 200 de senone (configurat implicit într-un proiect CMU Sphinx) este optim sau nu pentru sarcina de recunoaştere a vorbirii propusă în acest îndrumar (recunoaşterea secvenţelor audio ce conţin cifre).

Vom începe investigaţia antrenând un nou model acustic având în total 100 de senone disponibile pentru modelarea fonemelor ce formează cuvintele specifice sarcinii de recunoaştere. Pentru a realiza acest lucru ar trebui urmăriţi în principiu paşii descrişi în capitolul IV. Însă, pentru această sarcină nu vom crea un nou proiect CMU Sphinx, în consecinţă nu va trebui nici să adăugăm alte fişiere de resurse şi nici să transformăm fişierele cu resurse în formatul cerut de CMU Sphinx. Vom modifica numai fişierul de configurare al proiectului rodigits astfel încât numărul de senone să fie 100 (nu 200, aşa cum era în mod implicit):

```
cd ~/projects/rodigits/etc/
vi sphinx_train.cfg
```

Identificaţi în acest fişier linia care specifică numărul de senone:

```
# Number of tied states (senones) to create in decision-tree clustering
$CFG_N_TIED_STATES = 200;
```

Intraţi în modul de editare (tastând “i”) şi modificaţi-o astfel:

```
# Number of tied states (senones) to create in decision-tree clustering
$CFG_N_TIED_STATES = 100;
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter).

Intraţi în directorul proiectului şi porniţi antrenarea modelului acustic executând următoarea comandă:

```
cd ~/projects/rodigits
sphinxtrain_biosinf run
```

Aşteptaţi terminarea procesului de antranare (ar trebui să dureze aproximativ 2 minute). În cazul în care apar erori verificaţi faptul că fişierele cu resurse sunt în formatul corespunzător şi că nu aţi modificat greşit fişierul de configurare. În cazul în care nu reuşiţi să identificaţi şi să rezolvaţi problema discutaţi cu îndrumătorul de proiect.

După încheierea cu succes a procesului de antrenare verificaţi faptul că au fost generate modelele acustice corespunzătoare listând subdirectoarele din directorul proiectului, subdirectorul model_parameters:

```
cd ~/projects/rodigits
ls -all model_parameters
```

Lista subdirectoarelor afişate ar trebui să fie următoarea:

```
rodigits.cd_cont_100
rodigits.cd_cont_100_1
rodigits.cd_cont_100_2
rodigits.cd_cont_100_4
rodigits.cd_cont_200
rodigits.cd_cont_200_1
rodigits.cd_cont_200_2
rodigits.cd_cont_200_4
rodigits.cd_cont_initial
rodigits.cd_cont_untied
rodigits.ci_cont
rodigits.ci_cont_flatinitial
```

Observaţi că, pe lângă subdirectoarele existenete anterior (cele care conţin modele acustice cu 200 de senone şi număr variabil de densităţi Gaussiene), în această listă apar acum şi subdirectoarele rodigits.cd_cont_100* ce conţin modele acustice cu 100 de senone şi număr variabil de densităţi Gaussiene.

## Decodarea clipurilor audio de evaluare folosind modele acustice cu 100 de senone

Urmează ca acum să evaluăm aceste 4 noi modele acustice. Vom începe cu modelul acustic cu 100 de senone şi 8 densităţi Gaussiene per senone. Logaţi-vă pe serverul de dezvoltare, intraţi în directorul proiectului, în subdirectorul etc şi vizualizaţi fişierul de configurare al proiectului cu ajutorul editorului vim:

```
cd ~/projects/rodigits/etc/
vi sphinx_train.cfg
```

Identificaţi în acest fişier linia care specifică modelele acustice utilizate pentru decodare:

```
# Models to use.
#$DEC_CFG_MODEL_NAME =
"$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}";
$DEC_CFG_MODEL_NAME =
"$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}_4";
```

Intraţi în modul de editare (tastând "i"), ştergeţi linia decomentată (cea care specifica locaţia modelelor fonetice dependente de context cu 4 densităţi Gaussiene), apoi decomentaţi linia comentată (ce specifică locaţia modelelor fonetice implicite). Această zonă a fişierului ar trebui să arate în final ca mai jos:

```
# Models to use.
$DEC_CFG_MODEL_NAME = "$CFG_EXPTNAME.cd_${CFG_DIRLABEL}_${CFG_N_TIED_STATES}";
```

Identificaţi apoi linia care specifică numele subdirectorului în care se vor salva rezultatele:

```
$DEC_CFG_RESULT_DIR     = "$DEC_CFG_BASE_DIR/result.cd_cont_200_4";
```

Şi modificaţi-o astfel încât numele subdirectorului în care se vor salva rezultatele să fie sugestiv:

```
$DEC_CFG_RESULT_DIR     = "$DEC_CFG_BASE_DIR/result.cd_cont_100_8";
```

Ieşiţi din modul de editare (tastând Esc), salvaţi modificările (tastând ":w" şi apoi Enter) şi apoi închideţi editorul (tastând ":q" şi apoi Enter).

Intraţi apoi în directorul proiectului şi porniţi procesul de evaluare executând următoarea comandă:

```
cd ~/projects/rodigits
/usr/local/sphinx/lib/sphinxtrain/scripts/decode/slave.pl
```

Aşteptaţi terminarea procesului de decodare (ar trebui să dureze aproximativ 1 minut). În cazul în care apar erori încercaţi să identificaţi problema vizualizand fişierul de log al procesului de decodare:

```
vi logdir/decode/rodigits-1-1.log
```

În cazul în care nu reuşiţi să identificaţi şi să rezolvaţi problema discutaţi cu îndrumătorul de proiect. În cazul în care procesul de decodare se încheie cu succes ar trebui ca pe ecran să vă apară un mesaj similar cu acesta:

```
MODULE: DECODE Decoding using models previously trained (decode script: psdecode_fsg.pl)
    Decoding 20 segments starting at 0 (part 1 of 1)
    0%
    Aligning results to find error rate
    SENTENCE ERROR: 5.0% (1/20)   WORD ERROR RATE: 0.4% (0/240)
```

Aceste rezultate, precum şi informaţiile şi rezultatele detaliate din raportul de evaluare generat la acest pas (result.cd_cont_100_8/rodigits.align) vor fi comparate cu rezultatele obţinute în prealabil şi cu rezultatele ce vor fi obţinute în continuare şi vor fi în final prezentate şi discutate în raportul final al proiectului de cercetare-dezvoltare.

Repetaţi paşii de mai sus (ţinând cont şi de capitolul care prezintă decodarea folosind modele cu mai puţine densităţi Gaussiene per senone) realizând decodarea clipurilor audio de evaluare cu modele fonetice dependente de context cu 100 de senone şi 1, 2 şi respectiv 4 densităţi Gaussiene per senone.

# VIII. Construcţia unui sistem de recunoaştere independent de vorbitor

Acest capitol vă propune realizarea unui al doilea proiect de recunoaştere a secvenţelor audio ce conţin cifre, de această dată *independent de vorbitor* (îl vom numi rodigitsIndep). Pentru a justifica necesitatea unui sistem independent de vorbitor ar trebui să evaluaţi sistemul rodigits creat anterior pe fişierele de evaluare ale celorlalţi colegi de grupă (fişiere ce se află pe serverul de dezvoltare în directorul /home/biosinfShare/resources/wav/). În urma evaluării veţi constata că rezultatele de recunoaştere a vorbirii sunt mai slabe pentru ceilalţi vorbitori decât pentru voi înşivă. Explicaţia este simplă: modelul acustic din proiectul rodigits a fost antrenat doar cu vocea voastră şi, în consecinţă, vă recunoaşte bine numai pe voi!

Crearea unui proiect independent de vorbitor presupune reluarea tuturor etapelor prezentate în acest îndrumar, începând cu capitolul IV, cu o singură modificare: clipurile audio de antrenare şi evaluare nu vor mai fi doar clipurile înregistrate de voi, ci toate clipurile înregistrate de toţi colegii de grupă. Bineînţeles că această modificare poate duce şi la alte modificări (de exemplu adaptarea scripturilor createTranscriptions.sh şi createFileIds.sh), însă pe toate acestea le veţi aborda şi eventual discuta cu îndrumătorul de proiect la momentul respectiv.

# IX. Redactarea raportului final al proiectului de cercetare-dezvoltare

După efectuarea experimentelor prezentate în acest îndrumar veţi sumariza toate rezultatele obţinute şi veţi face o discuţie asupra lor într-un raport final aferent proiectului de cercetare-dezvoltare.

Raportul de cercetare-dezvoltare trebuie să înceapă cu o scurtă secţiune care să cuprindă noţiuni introductive cu privire la arhitectura sistemelor de recunoaştere a vorbirii, sub-sistemele implicate, etc. Această primă secţiune are rolul de a evidenţia nivelul de înţelegere (asupra sistemelor de recunoaştere a vorbirii) pe care l-aţi dobândit pe parcursul realizării proiectului.

O a doua secţiune a raportului de cercetare-dezvoltare va prezenta pe scurt cerinţa (tema) proiectului şi va enumera paşii ce au fost urmaţi pentru a dezvolta sistemul de recunoaştere a secvenţelor audio ce conţin cifre.

În final, partea cea mai consistentă a raportului trebuie să prezinte **toate** rezultatele obţinute şi să le discute în mod comparativ. În funcţie de stadiul în care aţi ajuns cu experimentele veţi avea rezultate de recunoaştere pentru:

- modelul acustic dependent de vorbitor:
    - cu 200 de senone:
        - independente de context
        - dependente de context, cu 1/2/4/8 densităţi Gaussiene per senone
    - cu 100 de senone:
        - independente de context
        - dependente de context, cu 1/2/4/8 densităţi Gaussiene per senone
- modelul acustic independent de vorbitor (cu acealaşi variaţii ale parametrilor)

Prezentarea individuală a rezultatelor (pentru fiecare experiment în parte)va trebui să se facă punând accentul pe rata de eroare la nivel de cuvânt (Percent Total Error = Word Error Rate), dar se pot discuta şi alte metrici adiacente cum sunt rata de cuvinte inserate (Percent Insertions), rata de cuvinte substituite (Percent Substitutions), rata de cuvinte şterse (Percent Deletions), rata de eroare la nivel de propoziţie (Sentences with errors), etc. De asemenea, în funcţie de rezultatele obţinute, pot fi prezentate perechile de cuvinte confundate foarte frecvent, cuvintele inserate/substituite/şterse cel mai frecvent şi eventual 1-2 exemple de fraze (evidenţiind varianta corectă a frazei şi varianta obţinută la ieşirea sistemului).

Compararea rezultatelor obţinute în diversele experimente se va face a) numai în funcţie de Word Error Rate (WER), dacă aţi efectuat multe experiemente sau b) în funcţie de mai multe metrici, dacă aveţi puţine rezultate. În primul caz rezultatele vor fi prezentate folosind tabele şi grafice de variaţie a ratei de eroare la nivel de cuvânt (WER). Rezultatele se vor discuta pe baza acestor tabele şi grafice şi, în final, se vor trage concluzii cu privire la dependenţa/independenţa de context, numărul optim de senone, numărul optim de densităţi Gaussiene, etc. pentru modelul acustic dependent de vorbitor, respectiv independent de vorbitor.

**Notă:** redactarea raport de cercetare-dezvoltare este obligatorie, iar transmiterea lui (prin e-mail) către îndrumătorul de proiect trebuie să se facă până pe data de **9 mai 2012**.

**Notă:** raporul final trebuie să fie redactat şi transmis îndrumătorului de proiect indiferent de stadiul în care aţi ajuns cu experimentele.

**Notă:** toate rezultatele prezentate în raportul de cercetare-dezvoltare vor fi însoţite de referinţe către rapoartele de evaluare de pe serverul de dezvoltare (calea către fişierele corespunzătoare).

# X. Anexa 1. Desfăşurarea activităţii pe server-ul de dezvoltare

Proiectul de recunoaştere a secvenţelor audio ce conţin cifre va fi realizat în întregime pe un server de dezvoltare aflat în cadrul laboratorului de cercetare SPEED. Studentul va trebui să dispună de un calculator conectat la Internet care va fi folosit pentru conectarea de la distanţă (prin intermediul Internetului) la serverul de dezvoltare.

Serverul de dezvoltare utilizează un sistem de operare de tip unix (Ubuntu 11.10). În consecinţă studentul trebuie să fie familiarizat sau să se familiarizeze cu modul de operare pe un astfel de sistem. În continuare sunt date câteva informaţii despre modul de conectare la server şi despre utilizarea editorului vim.

## Conectarea la serverul de dezvoltare

Conectarea la serverul de dezvoltare se face extrem de simplu în cazul în care calculatorul de care dispuneţi utilizează un sistem de operare de tip unix/linux. În acest caz conectarea la server se face utilizând un terminal în care executaţi următoarea comandă (important: modificaţi comanda următoare pentru a utiliza propriul nume de utilizator USER_NAME, aşa cum vi l-a comunicat îndrumătorul de proiect):

```
ssh -p 10022 USER_NAME@dev.speed.pub.ro
```

Serverul va raspunde cerându-vă să tastaţi parola contului vostru. Pentru a continua va trebui să tastaţi parola contului vostru, aşa cum v-a fost specificată îndrumătorul de proiect. În cazul conectării cu succes pe ecran vă va apărea următorul promt (unde veţi executa comenzile precizate în acest îndrumar):

```
USER_NAME@alpha:~$
```

În cazul în care calculatorul de care dispuneţi utilizează un sistemul de operare Windows atunci conectarea la serverul de dezvoltare se va face cu ajutorul unei aplicaţii intermediare numită *putty*. Această aplicaţie poate fi descărcată gratuit de pe siteul: http://putty.en.softonic.com/. După descărcare, porniţi aplicaţia şi completaţi în primul ecran *Host Name:* dev.speed.pub.ro şi *Port:* 10022. Pentru a salva aceste informaţii şi a nu fi nevoiţi să le completaţi de fiecare dată completaţi *Saved Sessions:* biosinfServer şi apăsaţi butonul *Save*. Pentru ca această aplicaţie să poată afişa corect caracterele diacritice, intraţi în meniul *Window*, submeniul *Translation* şi selectaţi *Remote Character Set*: UTF-8. Apoi reveniţi la ecranul iniţial, selectând meniul *Session* şi apăsaţi butonul *Open* pentru a stabili conexiunea. Serverul vă va cere numele de utilizator şi apoi parola (informaţii pe care vi le-a specificat îndrumătorul de proiect). În cazul conectării cu succes pe ecran vă va apărea următorul promt (unde veţi executa comenzile precizate în acest îndrumar):

```
USER_NAME@alpha:~$
```

**Recomandare:** primul lucru pe care ar trebui să îl faceţi după ce vă conectaţi pe server pentru prima dată este să vă schimbaţi parola. Pentru aceasta executaţi următoarea comandă şi introduceţi, pe rând, parola actuală, noua parola şi încă o dată noua parolă:

```
passwd
```

**Notă:** indiferent de modul de conectare la serverul de dezvoltare (terminal linux sau putty) aveţi posibilitatea şi de multe ori este chiar foarte util să deschideţi mai multe conexiuni simultane către server.

## Utilizarea editorului în mod text vim

În cadrul acestui proiect veţi lucra extensiv (creare, vizualizare, editare, modificare, etc.) cu fişiere text. Cel mai simplu mod de a manipula fişierele text aflate pe un server linux este cu ajutorul editorului vim.

Pentru a vizualiza un fişier deja existent sau a crea un fişier nou (cu numele filename) tastaţi comanda:

```
vi filename
```

În acest moment vizualizaţi prima parte a fişierului, iar cu ajutorul tastelor cu săgeţi şi a tastelor PgUp/PgDn puteţi vizualiza şi alte zone ale fişierului. Pentru a căuta un cuvânt (sau orice alt şir de caractere) tastaţi "/", apoi cuvântul respectiv şi în final Enter. Pentru a căuta acelaşi cuvânt din nou tastaţi "/" şi apoi Enter. Pentru a intra în modul de editare (în vederea modificării fişierului) tastaţi "i". Acum puteţi şterge şi scrie în acest fişier ca în oricare alt editor de text. Pentru a ieşi din modul de editare tastaţi Esc. Pentru a salva modificările făcute tastaţi ":w" şi apoi Enter. Pentru a părăsi editorul tastaţi ":q" şi apoi Enter. Pentru mai multe informaţii de utilizare şi alte comenzi de editare, copiere, etc. consultaţi site-ul vim precizat în bibliografia îndrumarului de proiect [8].

# XI. Bibliografie

[1] TODO Recunoașterea vorbirii continue, http://speed.pub.ro/biosinf/RecunoastereaVorbiriiContinue.pdf.

[2] Toolkitul CMU Sphinx şi tutorialele asociate, http://cmusphinx.sourceforge.net/wiki/

[3] H. Cucu, "Towards a speaker-independent, large-vocabulary continuous speech recognition system for Romanian," PhD Thesis, Bucharest, Romania, 2011.

[4] A. Buzo, "Automatic Speech Recognition over Mobile Communication Networks," PhD Thesis, Bucharest, Romania, 2011.

[5] D. Jurafsky, J. Martin, "Automatic Speech Recognition," Chapter 9 in *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition (2nd Ed.)*, Pearson Education, 2009.

[6] S. Renals, T. Hain, "Speech Recognition," in *Handbook of Computational Linguistics and Natural Language Processing*, 2010.

[7] Java Speech Grammar Format, http://java.sun.com/products/java-media/speech/forDevelopers/JSGF/index.html.

[8] Manualul editorului de text vi, http://www.cs.fsu.edu/general/vimanual.html.

**Notă:** documentele [3], [4], [5] și [6] se găsesc la http://speed.pub.ro/pcdtv/.