KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Andrejus Maričevas

LIGŲ PAVADINIMŲ ATPAŽINIMO TYRIMAS

Baigiamasis magistro projektas

Vadovas:
Doc. K. Ratkevičius

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

LIGŲ PAVADINIMŲ ATPAŽINIMO TYRIMAS

Baigiamasis magistro projektas
Informatika (M4016L21)

Vadovas:
Doc. Kąstytis Ratkevičius

Recenzentas:
Doc. Vytautas. Rudžionis

Projektą atliko:
Andrejus Maričevas

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS

(Fakultetas)

(Studento vardas, pavardė)

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „Pavadinimas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 ___ m. _____ ____ d.

Kaunas

Patvirtinu, kad mano, **Andrejaus Maričevo**, baigiamasis projektas tema „Ligų pavadinimų atpažinimo tyrimas” yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

_____ _____

(vardą ir pavardę įrašyti ranka) *(parašas)*

Maričevas A. Ligų pavadinimų atpažinimo tyrimas. Magistro baigiamasis projektas / vadovas doc. Kąstytis Ratkevičius; Kauno technologijos universitetas, Informatikos fakultetas.

Kaunas, 2015. 82 p.

SANTRAUKA

Šiuolaikinėms technologijoms sparčiai judant į priekį vien kompiuterių spartos ir aparatinės įrangos atnaujinimų neužtenka. Paprastai prieinama sąsaja tarp žmogaus ir kompiuterio virsta atgyvena – vis daugiau dirbama ties naujų galimybių sukurimu. Kalbinė sąsaja yra viena iš jų. Populiarėja asmeniniai asistentai mobiliuose įrenginiuose, kurie atlieka kasdienes užduotis – paiešką internete, užrašų įvedimą, įrenginio valdymą. Tačiau viršų ima tos kalbos, kuriomis susišneka didžioji dalis žmonių – anglų, kinų, ispanų ir kitos. Tokia maža šaliskaip Lietuva, su nepopuliare, nors ir turtinga kalba, susilaukia neduag dėmesio iš tokių technologijų gigantų kaip *Microsoft, Google* ir todėl mes patys turėtume tuo pasirūpinti. Vykdoma vis daugiau tyrimų siekiant realizuoti šnekos atpažinimo technologiją lietuvių kalbai – labiausiai pasižymėje šioje srityje yra tokie tyrėjai kaip P. Kasparaitis, S. Laurinčiukaitė, V. Rudžionis ar G. Raškinis. Nesistengiant prilygti didiesiems darbams, šis projektas yra apie galimybę apjungti du atpažintuvus – kitakalbį ir lietuviškąjį, siekiant ištirti skaičių pavadinimų ir lietuviškų vardų atpažinimo galimybes ligų kodų atpažinimui.

Šio darbo tikslas yra ištirti ligų kodų, susidedančių iš lietuviškų vardų ir skaičių pavadinimų, atpažinimą ir sudaryti bandomąją programą.

Tyrimo objektas – ligų kodai, kurių įvedimas grįstas lietuviškų vardų ir skaičių pavadinimų atpažinimu.

Tikslui pasiekti iškelti keli uždaviniai:

1. Atlikti literatūros šaltinių analizę kalbos atpažinimo ir susijusiomis temomis.
2. Paruošti ir ištirti kitakalbio atpažintuvo atpažinimo galimybes.
3. Įsisavinti HTK paketo programų veikimo principus ir ištirti lietuviškojo atpažintuvo atpažinimo tikslumą.
4. Sudaryti demonstracinę taikomosios programos versiją.
5. Pateikti išvadas ir rezultatus.

Projekto rengimo metu taikyti aprašomasis, kiekybinės analizės, eksperimentinis ir palyginamasis metodai. Atiekant tyrimus su kitakalbiu atpažintuvu siekta ištirti atpažinimo tikslumo priklausomybę nuo diktoriaus profilio ir jo apmokymo, taip pat atpažinimo tikslumo skirtumus tarp sinchroninės ir asinchroninės veiksenų. Darbo su HTK paketu metu buvo modeliuojamas naujas

lietuviškasis atpažintuvas, lyginami atpažinimo rezultatai keičiant būsenų skaičių ir pipildant jas įvairiu Gauso mišinių skaičiumi.

Tyrimo rezultatai parodė, kad išskirtinai žymaus ryšio tarp apmokytų ar ne diktorių profilių ir testavimo rezultatų nėra – visų keturių testuotų profilių rezultatai svyruoja nuo 94,12% iki 96,44%. Bet paaiškėjo, kad asinchroninės veiksenos atpažinimo rezultatai yra daug aukštesni nei sinchroninės veiksenos – asinchroninės veiksenos atpažinimo tikslumas yra 93,7%, kai tuo tarpu sinchroninės veiksenos tikslumas siekia tik 89,12%. Lietuviškojo atpažintuvo tyrimai buvo vedami dviem etapais. Pirmiausia buvo tiriama atpažinimo tikslumo priklausomybė nuo būsenų skaičiaus. Paaiškėjo, kad geriausi rezultatai pasiekti būsenų skaičiui esant lygiam raidžių skaičiui su 10 papildomų būsenų – skaičių pavadinimų atpažinimo tikslumas siekia 97,1%, o lietuviškų vardų maksimalus pasiektas atpažinimo tikslumas yra 84,87% su dviem papildomom būsenom. Toliau testuojant pridedant Gauso mišinius kaip papildomą parametrą nustatyta, kad geriausi rezultatai gaunami lietuviškų vardų atpažinimo testų metu naudojant 4 papildomas būsenas ir 10 Gauso mišinių – pasiektas 99,74% tikslumas. Skaičių pavadinimai tiksliausiai atpažįstami naudojant 2 papildomas būsenas ir 6 Gauso mišinius – 99,3% tikslumas.

Atlikus visus tyrimus ir bandymus padaryta išvada, kad kuriant hibridinį atpažintuvą tikslingiausia būtų naudoti kitakalbio atpažintuvo asinchroninės veiksenos režimą, o lietuviškajame derinti būsenų ir Gauso mišinių skaičių, ieškoti geriausios kombinacijos, mat nustatyta, kad parametrų skaičiaus didinimas gerus rezultatus pateikia tik iki tam tikros ribos, kol sistema neapkraunama per dideliu kiekiu papildomos informacijos.

SUMMARY

As the modern day technology advances, the CPU speed and the hardware upgrades are not enough. The common human-computer interface becomes vestige so the new aproaches and possibilities are sought and the spoken language interface is one of those. Personal AI assistants are becoming more popular on personal devices such as phone or laptop. They performr our everyday actions for us – remind us of a meeting, you can ask them to perform search on the internet or even ask to make a call. But the most common languages such as English, Chinese or German occupy the largest part of researches that are being carried on. As a result, small countries like Lithuania, whilst having beautiful and rich language do not get enough attention and effort from technological giants like *Google*, *Apple* or *Microsoft*, who are developing professional language-computer interfaces. Due to that we are the ones who have to take care of our legacy. More and more reasearches are being carried on in purpose to adapt speech recognition systems for Lithuanian language and the most significant job so far has been done by researchers like P. Kasparaitis, S. Laurinčiukaitė, V. Rudžionis and G. Raškinis. Having no intentions to match tehir work, this project is about the creation of a hybrid speech recognition system, combining Lithuanian recognizer and a non-native one in the meanwhile carrying on reasearches on recognition of Lithuanian names and digit names for the recognition of deasese codes.

The aim of this project is to investigate the recognition accuracy of Lithuanian names and digit names therefore to create a demo version of disease codes recognizer interface.

The object of the project – disease codes which are based on the recognition of Lithuanian names and digit names.

The goals of the research:

1. Perform an analysis of literature on speech recognition related topics.
2. Prepare the non-native language recognizer and investigate its recognition accuracy.
3. Master the operating principles of HTK suite and investigate the accuracy of Lithuanian recognizer.
4. Build a demo version of the recognizing software.
5. Provide conclusions and results.

The following methods used in this research: descriptive, experimental, quantitative and comparison method. In studies of the non-native language recognizer, the dependence of the speaker profile and its training were being investigated. The diferences in recognition accuracy between results of the synchronous and asynchronous modes were found. Using the HTK suite a new Lituanian

recognizer was built to obtain and compare the results by changing the number of states and in addition to varying the number of Gaussian mixtures.

The test results revealed that there is no significant change in recognition whether using trained or not trained user profile. All four test profiles provided results which vary from 94,12% to 96,44%. Although it turned aut the the asynchronous mode is capable of providing better results than the synchronous mode – it is 93,7% compared to 89,12% provided by the synchronous mode test results. The analysis of the Lithuanian speech recognizer was conucted in two phases. At first, the recognition accuracy dependencies on the number of states were researched. It appeared that the best digit recognition test results were achieved using 10 additional states – the result was 97,1%. The best result of Lithuanian names recognition was achieved using two additional numstates and is 84,87%.

In the second phase the recognition accuracy was being tested aster varying the number of Gaussian mixtures in addition to extra number of states. The best result of the Lithuanian names recognition was 99,74% using 4 additional states and 10 Gaussian mixtures. The highest accuracy of digit names was achieved using 2 additional states and 6 Gaussian mixtures – the result was 99,3%.

Upon completion of the test and examinations it was concluded, that the hybrid recognition system would best work using asynchronous mode of the non-native language recognizer together with Lithuanian recognizer coordinating its settings such as number of Gaussian mixtures and the number of states for the best results.

Turinys

TERMINŲ IR SANTRUMPŲ ŽODYNAS

ASCII – simbolių kodavimo sistema grįsta angliškų žodynu.

HTK – paslėptų Markovo modelių rinkinys (angl. Hidden Markov Model Toolkit).

UTF-8 – simbolių kodavimo sistema unikodo simboliams (tarp jų ir lietuviškiems).

PMM – paslėptas Markovo modelis (angl. Hidden Markov Model).

UPS – universaliųjų fonemų seka (angl. Universal Phone Set)

ASR – automatinis šnekos atpažinimas (Automati Scpeech recognition)

KTU ITPI – Kauno technologijos universiteto informacinių technologijų plėtros institutas

TLK-10-AM – Tarptautinės statistinės ligų ir sveikatos sutrikimų klasifikacijos dešimtasis pataisytas ir papildytas leidimas Australijos modifikacija

SAMPA – Kalbėjimo vertinimo metodas fonetine abėcėle (Speech Assessment Method Phonetic Alphabet)

X-SAMPA – Išplėstinis kalbėjimo vertinimo metodas fonetine abėcėle (Extended Speech Assessment Method Phonetic Alphabet)

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

ĮVADAS

Bendravimas, informacija, žinios ir komunikacija – terminai apibūdinantys dabartinę visuomenę. Kas valdo informaciją, tas valdo viską, bet kas greitai ja disponuoja ir manipuliuoja – tas yra pranašesnis. Informacija neatsiejama nuo technologijų, jų tobulėjimo. Tobulėja ir kalba, komunikacija. Tad turėtų tobulėti ir sąsaja tarp technologijos ir kalbos, o dar geriau jei kalba taptų technologija. Galimybė bendrauti su savo įrenginiu, balsu paliepti jam atlikti reikalingus veiksmus ir jo gebėjimas tave suprasti, kol pats asmuo dirba kitus darbus – tai yra ateitis.ateitis jau čia toms kalboms, kuriomis kalbama daugiausia – anglų, japonų, vokiečių. Lietuvių kalba yra per maža korporacijų (Apple, Google ir kt.) dalyvavimui procese, kokia toji kalba turtinga bebūtų.

Lietuvos tyrėjų veikla jau daugiau nei dešimtmetį sėkmingai vystoma šnekos atpažinimo tema. Yra ištirta nemažai galimybių kurti izoliuotų ar daugiažodžių komandų atpažintuvus (E. Baubartas), parašyta darbų apie daugiadiktorinių garsynų sudarymą. Bet didžioji dalis tyrimų atlikta ties atpažintuvais ar atpažinimo perspektyva naudojant Paslėtųjų Markovo grandinių modelius (Lipeika, Šilingas).

Šio projekto metu analizuojamas ne tik Paslėtųjų Markovo grandinių taikymas, bet ir šios technologijos derinimas su kitakalbiu atpažintuvu – hibridinio šnekos atpažinimo tehnologija. Naudojantis KTU ITPI resursais – sudarytais lietuviškų vardų ir skaičių pavadinimų garsynais bus siekiama ištirti atpažintuvų veikimo ypatybes.

Darbo tikslas. Ištirti ligų pavadinimų atpažinimą pagal jų kodus, susidedančius iš lietuviškų vardų ir skaičių pavadinimų komandų, ir sukurti demonstracinę atpažinimo programos versiją.

Magistrinio darbo objektas yra ligų pavadinimai, kurių atpažinimas bus vykdomas pagal ligų kodą, nes pačių pavadinimų kiekis yra per didelis tinkamai paruošti balso atpažinimo sistemą atsižvelgiant į darbo apimtį. Darbo tikslui pasiekti buvo iškelti tokie uždaviniai:

1. Atlikti literatūros šaltinių analizę šnekamosios kalbos atpažinimo, garsynų, gramatikų sudarymo bei transkribavimo temomis;
2. Paruošti ispanų kalbos atpažintuvą darbui, ištirti jo atpažinimo tikslumą;
3. Įsisavinti HTK atpažinimo programų paketo veikimą ir atlikti dalinį garsyno testavimą;
4. Paruošti bandomąją demonstracinės programos versiją.
5. Išanalizuoti gautus darbo rezultatus ir pateikti išvadas.

Tyrimo metodai. Atliekant darbą, buvo pritaikyti aprašomasis, kiekybinės analizės, palyginamasis, eksperimentinis bei bandymų – klaidų metodai. Darbo metu naudotasi *Microsoft Speech Recognizer 8.0 for Windows*, dirbant *MS Windows 7* aplinkoje; *MS Visual Studio 2010* rengiant kalbos

atpažinimo sistemos bandomąją versiją ir su tuo pačiu paketu sukurtos testavimo bei analizės programos. Didžioji dalis tyrimų ir testų buvo atlikta *KTU* (Kauno technologijos universiteto) *ITPI* (Informacinių tyrimų plėtros instituto) laboratorijoje. Baigiamojo bakalaurinio darbo tikslams pasiekti buvo naudojamasi *KSTL* (Kalbos signalų tyrimo laboratorijos) darbuotojų sukurtomis programomis.

Darbo aktualumas ir naujumas. Tokio tipo hibridinio atpažintuvo kurimui pagrindai padėti 2011-2013m. vykdyto projekto Infobalsas metu – buvo analizuota galimybė sudaryta dviejų skirtingų, bet panašių fonetinių savybių kalbų hibridinį atpažintuvą, apjungiant kiekvieno iš dviejų atpažintuvų geriausias savybes. Buvo atliekami tyrimai ir testavimai ligų, usiskundimų ir vaistų atpažinimo temomis. Šio projekto pagrindu UAB „Softdent“ 2015 m. pradėjo realizuoti komercinę medicinos įstaigoms skirtą balso sąsajos programinę įrangą.

Panašiomis temomis rašomi darbai daugelyje Lietuvos mokslo įstaigose – Šiaulių universitete, Vytauto Didžiojo universitete ir be abejonės, Kauno technologijos universitete. Darbų, kuriuose būtų aprašomi HTK programų paketu paremtų lietuviškų atpažintuvų sudarymo ir testavimo tyrimai yra iš tiesų nedaug, todėl šie tyrimai yra ypatingai svarbūs kalbos atpažinimo srityje.

Darbo struktūra. Šis darbas susideda iš dviejų pagrindinių dalių – analitinės ir praktinės dalies. Analitinėje dalyje aprašomi teoriniai balso atpažinimo principai, reikalingi komponentai, taikomos technologijos. Praktinėje dalyje aprašyta tyrimų eiga, gautų rezultatų analizė, interpretuojami gauti duomenys, aprašomos demonstracinės programos, jų veikimo principai. Išvadose pateikiamas apibendrinimas, aptariami gauti rezultatai. Pateikiamas terminų bei santrumpų sąrašas, lentelių bei paveikslų sąrašai. Pridedami 9 priedai, kuriuose pateiiamos didesnė lentelės, gramatikų failų turinys, demonstracinių programų kodai.

1. BALSO TECHNOLOGIJŲ APŽVALGA

Kalba egzistuoja jau tūkstančius metų, ji tobulėjo, evoliucionavo. Ilgainiui kalbos bendrauti tarpusavyje tapo negana – atsivėrus technologijų amžiui, tobulėjant kompiuteriams ir mašinoms žmogus suprato, kad nori bendrauti ir su jį supančia aplinka, pradėjo tirti, kaip tai galima įgyvendinti. Šiame skyriuje apžvelgsiu technologinę pažangą ir sprendimus, kuriuos naudojant toks bendravimas tapo įmanomu.

1.1 Šnekos atpažinimas

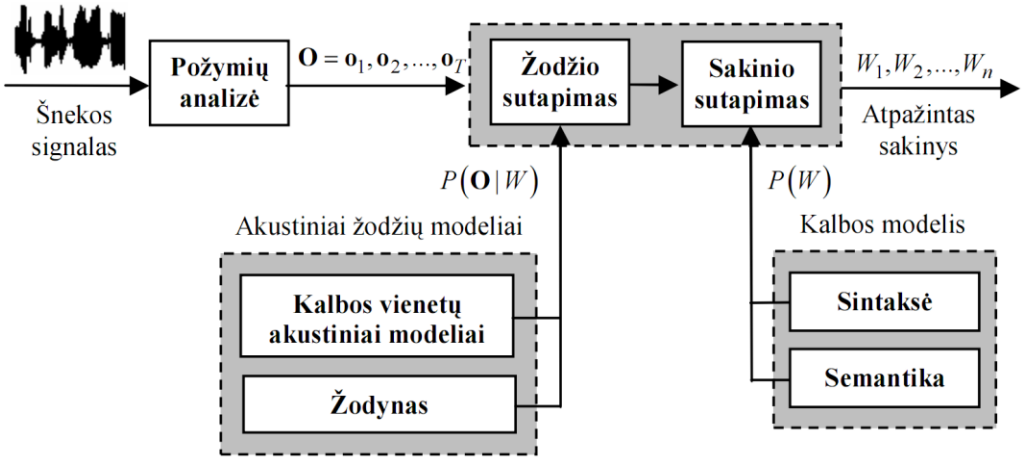
Kalbos suvokimas yra žmogaus įgimta ir vystoma savybė. Mes ją tobuliname, giliname. Kompiuterį priversti tai dryti yra be galo sunku dėl dar iki galo neišsiaiškintų žmogaus sugebėjimų. Jau daugiau nei 40 metų šios technologijos yra tobulinamos ir dar nė iš tolo nėra priartėta prie žmogaus suvokimo galimybių.

Technologijoms sparčiai tobulėjant, galima greitai sulaukti prasiplėtusios šios technologijos pritaikymo sferos – utopine atrodanti mintis, kad kada nors galėsime paskambinti savo šaldytuvui, gali greitai virsti visai realia. Taigi automatinis kalbos atpažinimas (angl. ASR – *automatic speach recognition*) yra kompiuteriniais modeliais paremta sistema, kuri atpažįsta žmogaus į mikrofoną ar kitą įrašimo įrenginį tariamus žodžius ir paverčia juos rašytiniu tekstu. Kitaip tariant, kompiuteris priima žmogaus balso įrašą, suskaido jį į žodžius, iš kurių kiekvienas turi skirtingus sintaksinius ir semantinius požymius, juos randa savo duomenų bazėje ir atrinkdamas tinkamiausią variantą, atlieka savus semantinius ir sintaksinius pakeitimus prieš pateikdamas galutinį atpažintos kalbos variantą.[18]

Kalbos atpažinimo tyrimai prasidėjo XX a. šeštajame dešimtmetyje, 1952 metais *BellLaboratories* darbuotojai Davis, Biddulph ir Balashek sukūrė pavienių garsų atpažinimo sistemą skirtą vienam diktoriui. 1956 m. *RCA Laboratories* mokslininkai Olson ir Belar nepriklausomo tyrimo metu bandė atpažinti 10 atskirų vieno diktoriaus skiemenų, kurie sudarė 10 vienaskiemenių žodžių. Abi atpažinimo sistemos buvo paremtos spektrinių rezonansų matavimais. Reikšmingu momentu laikomas 1959 metų Forgie ir Forgie bandymas sukurti balsių atpažinimo sistemą, kuri sugebėjo atpažinti balsius nepriklausomai nuo diktoriaus. Tokie pirminiai bandymai suteikė daug galimybių tolesniems tyrimams ir jau septintajame dešimtmenyje iškilo keletas fundamentalių kalbos atpažinimo idėjų. Tuo laikotarpiu japonų mokslininkai įsitraukė į kalbos atpažinimo sistemų kurimo varžybas ir pradėjo kurti atpažinimo mašinas. Be viso to, tuo metu buvo pradėti trys reikšmingi projektai, kurie įtakojo kalbos atpažinimo tyrimus maždaug dvidešimčiai metų į priekį. Pirmasis buvo Martin ir jo kolegų iš *RCA Laboratories,* jie atliko tyrimus ir pateikė sprendimus dėl laiko skalės nevientisumo kalbėjimo aktuose. Jie pateikė kelis

laiko normalizavimo metodus, kurie padėdavo nustatyti frazės pradžią ir pabaigą, dėka to tęsdamas darbus, įkūrė pirmąją kompaniją, *Threshold Technology*. Beveik tuo pačiu metu tuometinėje Sovietų Sąjungoje, Vintsyuk pasiūlė R. Bellman sukurtą dinaminio programavimo metodą ištarimų laiko suvienodinimui. Trečias reikmingesnis to meto technologinis pasistumėjimas kalbos atpažinimo tyrimuose buvo Reddy darbas ties testiniu kalbos atpažinimu, dinamiškai sekant fonemas. Šie jo tyrimai paskatino tolimesnius *Carnegie Mellon* universiteto tyrimus šioje srityje ir paliko juos vienvaldžius lyderius iki šių laikų.[21]

Bet nors ir šeštajame ir septintajame dešimtmečiuose kalbos atpažinimo studijų raida vystėsi labai sparčiai ir mokslininkai jau tikėjosi, kad po dešimties metų laisvai bendrausime su kompiuteriniais įrenginiais, XI-ajame amžiuje nesame per daug pasistūmėję į priekį. Atsiranda labai daug spragų ir kliučių žmogaus balso atpažinime. Žmogaus balso atpažinimo procesas yra daug sudėtingesnis nei rašytinio teksto (grafinė balso atpažinimo schema pateikiama 1 pav.). Iškilusias problemas lemia fiziologinės ir psichologinės žmogaus savybės, nekalbant apie technologinius netobulumus. Kiekvieno žmogaus balso trakto ilgis ir forma labai skiriasi, vien dėl to kiekvieno mūsų balsai yra skirtingi. Netgi naudojant tą pačią balso įrašymo įrangą tam pačiam žmogui skirtingu metu įrašai skirsis dėl psichologinių priežasčių, nuovargio. Be to, didelę įtaką turi amžius. Technologinės problemos prasideda jau nuo mikrofono – skirtingi įrašymo įrenginiai įrašo skirtingą dažnių spektra turinčius signalus, be to ne visada įvertinamas aplinkos triukšmas.[24]



1 pav. Šnekos atpažinimo sistemos schema

Kalbant apie kalbos atpažinimo problemas, Forsberg M. [8] išskiria nemažai jų, bet paminėsiu pačias reikšmingiausias.

a) Pagrindinis skirtumas tarp žmogaus suvokimo ir kompiuterio, išmokyto suprasti naturaliąją kalbą yra tas, kad bendraudamas žmogus suvokia ne tik tai, ką girdi tuo metu, bet jis

naudoja sukauptą informaciją apie asmenį iš praeities, kūno kalbą. Automatiniame kalbos atpažinime surime tik kalbos signalą be kontekstinės informacijos, kuri palengvintų atpažinimą. Tad kaip prieversti kompiuterį mąstyti plačiai yra vis dar didžiulė problema.

b) Taikant kalbos atpažinimo sistemą natūralioje aplinkoje yra praktiškai neįmanoma išvengti pašalinio triukšmo – kalbančiųjų, laikrodžio tiksėjimo, kitų audio šaltinių aplinkoje. Idealios salygos galimos tik studijoje, todel atpažinimo sistema turi sugebėti filtruoti, o prieš tai atskirti pašalinį triukšmą.
c) Kuriant balso atpažinimo sistemą komercinėms įmonėms ar tokioms, kuriose ji bus naudojama nepriklausomai nuo vartotojo, reikia užtikrinti, kad sistema sugebės funkcionuoti nepriklausomai nuo diktoriaus. Tai reiškia, kad ji negali būti susieta su vieno žmogaus balsu, o esant dideliems skirtumams tarp pavienių balsų, ypač tarp skirtingų lyčių, yra sunku užtikrinti stabilų sistemos darbą.
d) Kalbėjimo greitis ir stilius yra saviti kiekvienam kalbančiajam. Programuotojas negali būti tikras, kad kalbėtojas stengsis prisitaikyti prie bendrinio kalbėjimo stiliaus ir kalbės kitaip negu yra įpratęs. Kiekvienas yra įpratęs kalbėti savo tempu, vartoti savitą žodyną bei tarimą, kirčiavimą.

1.1.1 Šnekos atpažinimo sistemų klasifikacija

Šnekos atpažinimo sistemos skirstomos pagal atpažįstamų žodžių skaičių.

1. Pavienių balso komandų atpažinimas. Kalbos komandos atpažįstamos kaip atskiros komandos nors ir yra ištartos junginyje – tarp kiekvieno ištarimo reikia daryti trumpas pauzes, kad atpažintuvas traktuotų ištarimus kaip atskiras komandas. Galima sakyti, kad diktorius turi palaukti, kol sistema atpažins kiekvieną komandą.
2. Kelių žodžių sekos atpažinimas. Tai siejasi su pavienių komandų atpažinimu, bet pauzės, daromos tarp žodžių yra mažesnės ir programa apjungia ištarimus į vieną seką.
3. Rišlios šnekos atpažinimas. Suteikiama galimybė kalbėti natūraliai, nedarant dirbtinių pauzių ir nereikia laukti, kol sistema atpažins keikvieną komandą. Taikoma diktavimo atpažinimui.
4. Spontaniškos šnekos atpažinimas. Tai labai sudėtingas atpažinimo modelis, kuris suteikia galimybę atskirti kelis žodžius suplaktus į vieną, taip pat mikčiojimą, atrasti reikiamą informaciją ištartame sakinyje ar ieškoti svarbių atitikmenų. Daugelis projektuojamų sistemų vis dar embriono stadijoje.

Kiekviena tokia sistema su kita dalijasi bendru tikslu – kiek įmanoma tiksliau atpažinti diktoriaus ištartą žodį, bent ne prasčiau nei tai daro žmogus. Žinoma tokie aplinkos faktoriai kaip triukšmas, trukdžiai ar kiekvieno kalbančiojo asmeninės savybės, balso ypatumai, apsunkina užduotį, bet atpažinimo tikslumas neturėtų kristi žemiau 90% ribos. Atskirais atvejais, kai yra taikomos specialios salygos, tokios kaip žodyno apimtis, izoliuota aplinka, galima pasiekti net 99% tikslumo ribą.

1.1.2 Šiuolaikiniai šnekos atpažinimo modeliai

Nors vienas svarbesnių vertinimo kriterijų yra procentinė atpažinimo tikslumo išraiška, labai svarbu palyginti kaip tarpusavyje susisieja ir „susišneka“ kompiuteris ir žmogus. Vieni svarbiausių šių palyginimų yra atlikti R. Lippman [13]. Jis atliko tikslumo analizes tikrinant skaičių, raidžių, teksto diktavimo bei laisvo stiliaus atpažinimą. Tie tyrimai parodė, kad žmogaus sugebėjimai bene 10 kartų viršyja kompiuterio sugebėjimus – žmogus lengviau fiksuoja prastai ištartus garsus, nutylėjimą, samplaiką ar nereaguoja į trukdžius.

Paradoksalu atrodo tai, kad bene paprasti uždaviniai išlieka tokie patys sudėtingi, kokie jie buvo prieš 5 metus, ar net sudėtingėja. Buvo spėliota ir pastebėta, kad atsinaujinant galimybėms ir studijoms, 10 skaitmenų atpažinimas vis sudėtingėja, o ne paprastėja. Huang [28] spėja, kad skaičių atpažinimo kokybei prireiks kone 40 metų kol ji pasieks žmogaus sugebėjimo ribą. Taip yra todėl, kad pagrindinė problema yra ne izoliuotų skaitmenų atpažinimas, bet rišlios, spontaniškos kalbos atpažinimas. Nustatyti, ką diktorius pasakė kai jis turi rinktis tik vieną iš dešimties skaitmenų, pavyzdžiui, vykdydamas užsakymą ar duomenų pildymą telefonu, nėra labai sudėtinga, bet atskirti, kas yra sakoma kai įdiktuojamų žodžių eilutė be pauzių yra tariama įvairiais dialektais yra didžiulis uždavinys.

Skaičių atpažinimas tėra maža dalis to, ką yra siekiama įgyvendinti. Akustiškai sudėtingas dėl signalo trumpumo ir fonemų duslumo yra raidžių pavadinimų apažinimas. [14] Dėl šio sudėtingumo svarbesniu objektu tampa tai, kaip ir kodėl žmogus suvokia garsus. Ar tai yra patirtis ar tai jo klausos sistemos unikalumas. Visai įmanoma, kad būtent dėl to mes kitaip komrehuojame garsus ar jų junginius. Dusan savo darbe [6] teigia, kad žmogaus kalbos atpažinimą ir kompiuterinį šnekos atpažinimą jau galima lyginti netgi šešiais lygmenimis. Vienas svarbesnių yra tas, kad žmogus apdoroja informacija milijonais neuronų, jo klausa sugeba susisteminti bendrą informaciją, kuri susideda iš emocinės šnekos dalies, akcento, dialekto ar balso ypatybių. Tuo metu kompiuteris remiasi jam pateikta informacija ir praleidus tam tikrą akcentą, priimtas sprendimas ar įvertinimas gali būti neteisingas. Dėl šios priežasties daugeliu atveju bandoma prilyginti žmogaus šnekos suvokimą arčiau kompiuterinės programos realizacijos. Deja, žmogaus sugebėjimo netgi nuspėti dalinai ištarto žodžio dalį ar užpildyti trūkstamas spragas tiek fonetine tiek prasmine kontekstine prasme yra dar neišpildytas automatinio šnekos

atpažinimo srityje. Kas kartą „bendraudamas“ su kompiuteriu žmogus turi suvokti, kad toji mašina nesuvoks ir netgi suges jei jis bandys bendrauti su ja taip, lyg natūraliai bendrautų su kitu gyvu asmeniu. [24] Šnekos atpažinimo srityje tai yra didžiulė praraja, kurią iki šiol tyrėjams ir mokslininkams yra gana sudėtinga užpildyti.

1.1.3 Šnekos atpažinimo veikla Lietuvoje

Lietuvos tyrėjų vykdyti eksperimentai ir veikla iš pagrindų siejasi su Paslėptų markovo modelių taikymu šnekos atpažinimo technologijoje.

Vienas naujesnių tyrimų, atliktas VGTU absolventų [4], parodė, kad atpažinimo tikslumas spriklauso ne tik nuo žodyno dyžio, bet ir nuo komandų ilgio. Kuo ilgesnės komandos, tuo legviau programai užfiksuoti komandos pradžią ir pabaigą – esmines proceso dedamasias. Be to, esant didesniam žodynui, klaidos tikimybė išauga dėl galimų variacijų skaičiaus. Nors projekto metu gauti rezultatai svyruoja nuo 92,6% iki 95,5% (paklaida 2,9%), toks svyravimas, matuojant tikslumą reiškia labai daug atpažinimo technologijų srityje.

E. Braubarto [5] atlikti tyrimai, taip pat paremti Paslėptų Markovo modelių technologijos taikymu parodė, kad pavienių žodžių atpažinimas spadidėja skirstant juos į katergorijas. Panašus tyrimas, atliktas tais pačiais metais [12], parodė, kad ištisinės kalbos atpažinimas, grįstas fonemomis pagerina atpažinimo tikslumą iki 76%. Šilingo darbe [25] nustatyta, kad dirbant su PMM, Gauso mišinių naudojimas suprastina atpažinimą. Nustatyta, kad naudojant du Gauso mišinius, tik 8% atpažinimo pagerėjimo yra susiję su mišinių kiekiu, o naudojant tris Gauso mišinius, atpažinimas tik suprastėja. Be to, tai sukelia sistemos apkrovą, prailgina proceso trukmę ir reikalauja daugiau išteklių, todėl Gauso mišinių naudojamas, pasak autoriaus, rekomenduojamas tik tais atvejais, kad ištekliai nėra riboti. Kitas susijęs tyrimas buvo atliktas ir aprašytas 2004 metais [7], pritaikant hibridiškumo principą. Tarpusavyje derinti PMM bei neuroniniais tinklais paremti atpažintuvai. Rezultatai iš tiesų įdomūs – naudojamas neuroninių tiklų atpažintuvas yra patikimesnis, priimami sprendimai yra labiau užtikrinti, nėra bandoma surasti visiškų atitikmenų, o tiesiog sulyginti galimus garsinius požymių atitikmenis. Kita vertus, nustatyta, kad apmokymo procesas yra sudėtingesnis nei PMM atveju – reikalauja daugiau skaičiavimų, yra lėtesnis dėl pačios sistemos. Dažnai pasitaikanti problema yra per didelis apmokymas – reikalingų parametrų kiekis yra gana gausus, bet dėl nenaudojamo kryžminio patikrinimo sistema yra perkraunama ir gali užstrigti. Nepaisant šių problemų, sistemų hibrdinis suderinamumas pateikė 91% atpažinimo tikslumą.

1.1.4 Garsynas ir jo pritaikomumas kalbos atpažinime

Kalbos signalų technologijoms žengiant sparčiu žingsniu į priekį, buvo pasiūlyta nemažai praktinio vartojimo sistemų. Tačiau pritaikyti produktus atskiroms rinkoms ar vartojimo segmentams yra labai sunku. Kiekviena programa ar sistema remiasi tam tikru akustiniu – fonetiniu turiniu, kurį surinkti yra labai sunku, nes nėra galimybės įrašyti visus kalboje gyvuojančius akustinius – fonetinius vienetus arba nėra pakankamai specialistų ar diktorių. Garsynas skirtas tam, kad apmokant atpažintuvą būtų kokybiškai paskaičiuoti fonetiniai modeliai, todėl garsyno pateikta akustinė/fonetinė medžiaga turi būti išsami ir atitikti tam tikrus standartus – turi būti atvaizduojama diktorių įvairovė, apmokymo procesas neturi būti per daug sudėtingas.

Kaip teigia Balvočius ir Telksnys [1], garsynai skiriasi trukme, transkribavimo lygmeniu, diktorių skaičiumi, žodyno dydžiu, renkamų garso įrašų turiniu ar pan. Atpažinimui naudojant tam tikrus garsynus didelę įtaką daro ir triukšmai bei kiti trukdžiai esantys garsynuose. Pagal paskirtį garsynus būtų galima išskirti į tokias grupes: garsynai, kurie buvo kurti balso atpažinimo nuo diktoriaus nepriklausančias sistemas ir yra sistemos, kurios balsą atpažįsta, tačiau turi būti pritaikomos (apmokomos) prie diktoriaus. Pagal tipą garsynai skirstomi į garsynus, kuriuose surinkti įrašai yra "perskaitytos šnekos" (kitaip sakant nespontaniškos šnekos įrašai) (pvz.: knygų ištraukos, tam tikri žodžių sąrašai, skaičių sekos ir pan.) ir garsynus, kurių turinį sudaro spontaniškos kalbos įrašai (pvz.: dialogai ir kt.). Garsynų anotavimui ir segmentavimui įprastai naudojami tie patys metodai, kaip ir šnekos atpažinimui: neuroniai tinklai, DLSK (dinaminis laiko skalės kraipymas, angl. — DTW — Dynamic Time Wraping) ir HMM modelis. Džniausiai šie metodai taikomi šnekos technologijų tinklalapiuose aprašytose anotavimo ir segmentavimo sistemose. [20]

Šiomis dienomis, vis labiau tobulėjant įvairioms automatinėms šnekos atpažinimo sistemoms, norima, jog garsynai taip pat atitikitų šiuolaikinių technologijų standartus. Kaip buvo minėta anksčiau, garsynai gali turėti ir papildomą programinę įrangą, bei būti sudaryti ir pripildyti kita svarbia medžiaga. Pasak Balvočiaus ir Telksnio [1] šiuolaikinis garsynas, kuris galėtų būti efektyviai naudojamas šnekos atpažinimo sistemų kūrimui, turėtų turėti tokias savybes:

- garsynas turėtų leisti vartotojui manipuliuoti dideliais duomenų kiekiais (perkelti, atnaujinti, naikinti) ir užtikrinti duomenų korektiškumą bei integralumą. Nuolat didėjant šnekos signalų ir papildomų duomenų kiekiams, svarbu turėti galimybę patogiai ir paprastai valdyti visus duomenis;

- garsynas turėtų turėti efektyvią paieškos sistemą, užtikrinančią tiek šnekos signalų, tiek meta-duomenų ištraukimą. Nuo paieškos sistemos efektyvumo priklauso duomenų ištraukimo greitis ir bendras pasiruošimo, tam tikram tyrimui, greitis.
- garsynas turėtų turėti savybes atitinkančias bendro darbo sistemos principus (daugiavartotojiškumas, duomenų kontrolė ir dalinimasis). Be informacijos dalinimosi ir bendradarbiavimo visas tyrimo progresas taptų žymiai lėtesnis;
- garsynas turėtų turėti standartinę sąsają duomenų lygyje (duomenų importavimui ir eksportavimui į kitas sistemas). Turint standartinę sąsają duomenų lygyje, yra galimas laisvas dalinimasis sukaupta informacija, tarp kitų sistemų, turinčių to pačio standarto sąsajas;
- garsynas turėtų turėti standartinę programinę sąsają, kad sukūrus papildomą funkcionalumą, jis taptų lengvai prieinamas ir kitiems vartotojams;
- garsynas taip pat turėtų turėti ir vartotojo sąsają patogiam duomenų bazės užklausimui ir ataskaitų formavimui. Patogi vartotojo sąsaja leidžia koncentruotis į tyrimą , o ne į duomenų tyrimui paruošimą;

Didžiausią įtaką tolimesnei garsynų raidai, tikriausiai padarė sukurtas **TIMIT** garsynas**.** TIMIT (Texas Instruments in Massachussets Institute of Technology) - yra laikomas klasikinio garsyno pavyzdžiu. TIMIT garsynas remiasi hierarchinės duomenų bazės struktūra, todėl yra aiškus ir lengvai suprantamas. Duomenys šiame garsyne yra sutalpinti į keturių tipų failus, kurie saugo informaciją tiek apie kalbėtojus tiek ir apie anotacijas.[1] TIMIT garsyno duomenis vieno seanso metu perskaitė 630 diktorių (kiekvienas po 10 sakinių), atstovaujantys aštuonis JAV dialektinius regionus. Šis garsynas pasižymi ypač kruopščiai sužymėtomis fonemų ribomis (daugelyje kitų pateikiamos tik žodžių ribos). Žemiau pateikiamas pavyzdys kaip TIMIT garsyne patalpinamų duomenų struktūra.

Verta paminėti, kad norint sukurti tikslią kalbos atpažinimo sistemą būtiną užtikrinti tinkamo garsyno pateikimą.

1.2 Projektas Infobalsas

Balso sąsaja yra kalbos atpažinimo ir/ar sintezės priemonių panaudojimas pritaikant kitas kompiuterines priemones arba savarankiškai. Kitų priemonių pritaikymas leidžia šią paslaugą pritaikyti kompiuteriuose. [9] Naudojant kompiuterinius modelius, aparatinę įrangą, galima sukurti įvairiomis web paslaugomis grįstus programos modelius, kas ir buvo pagrindinis šio projekto tikslas.

Be visa ko, pagrindinė šios sistemos dalis yra lietuvių kalbos atpažinimo modelis, be kurio sistema negyvuos, o tam reikia garsynų, kuriais remiasi atpažinimo bei testavimo procesas. Siekiant

tinkamai paruošti balso sąsają, reikia parinkti ar sukurti tinkamiausią atpažintuvą ar netgi jų derinį. Šiame projekte siekiama ištirti, kokiu būdu ir ar įmanoma suderinti su skirtingus atpažintuvus, kurie vienas kitą papildydami vykdytų daugiadiktorinių lietuviškų medicininės srities balso komandų atpažinimą. Taigi pirmasis, bazinis atpažintuvas, yra kitų, ne lietuvių kalbų atpažintuvas, kuris pritaikytas lietuvių kalbai naudojant daugiakalbio atpažintuvo principus. [24] Antrasis atpažintuvas yra lietuvių kalbai kurta, o toliau tobulinta programinė įranga, paremta PMM principais. Pradinė programos versija kurta dar 2006 m. [25] Vėlesni rezultatai aptarti ir pateikti G. Norkevičiaus ir kt. darbe. [14]

Hibridiškumo principas yra galimybė išnaudoti tiek vieno, tiek kito atpažintuvo privalumus vienu metu. Toks lygiagretinimas naudingas pirmajam atpažintuvui klystant, o antrajam priimant teisingą sprendimą ir atvirkščiai. Nelietuviško atpažintuvo privalumas yra integralumas – jį lengvą pritaikyti prie kitos programinės ar aparatinės įrangos, be to jie palaiko įvairius integruotus modulius, tokius kaip triukšmo slopinimas, signalo aptikimas. Lietuviškasis atpažintuvas naudingas tuo, kad naudoja statistinius atpažinimo modulius, kurie sukurti remiantis daugelio diktorių lietuvių šnekamosios kalbos įrašais.

Nelietuviško atpažintuvo parinkimo metodas paremtas atpažintuvo apmokymu, kurio tikslas rasti akustinius modelius, kurie geriausiai aprašo lietuvių kalbos fonetinę sistemą. Tiriant anglų bei ispanų kalbos fonetinius modelius pagal fonemų sąrašus [17,18], nustatyta, kad anglų kalbos atpažintuvas naudoja 18 balsių ir 27 priebalsių fonetinius modelius, o ispanų kalboje kiek mažiau – 11 balsių ir 20 priebalsių fonetiniai modeliai. Visi šie modeliai pritaikyti įvariems tarimo variantams pagal intonaciją ar tarimo būdus įvariuose fonetiniuose kontekstuose.

Pagrindinis fonetinių vienetų lietuvių kalbai adaptavimas yra transkripcijų sudarymas. Tai buvo bandoma padaryti keliais būdais. Naudoti tiek ispaniški, tiek anglų kalbos sintezatoriai, kurie suvestą tekstą perteikdavo garsinę išraiška, taip pagal skambesį buvo galima nuspręsti, kur ir kaip reikia pakeisti pateiktą transkripciją, kad skambesys atitiktų natūraliai ištartą lietuvišką žodį. Be to, naudoti ir pramoniniai automatiniai generatoriai, kurie pateiktam tekstui pateikdavo galimus tarimo variantus. Pavyzdžiui, pavardei „Davalga“ ispaniškas atpažintuvas siūlo du tarimo variantus D A B A L G A ir D E J B A L G A, o vardui „Jonas“ tik vieną - X O N A S.

Po ispaniško ir angliško atpažintuvų palyginimo [3] pastebėta, kad naudojantis ispanų kalbos atpažintuvu galima daug greičiau surasti transkripcijas lietuviškoms komandoms. Atlikus tyrimą su 100 nesudėtingų lietuviškų pavardžių, pateikiant neoptimizuotas transkripcijas nustatyta, kad ispanų kalbos atpažintuvas duotas komandas atpažįsta 90% tikslumu, kai tuo tarpu anglų kalbos atpažintuvas atpažino tik apie trečdalį pateiktų komandų.

Po šio tyrimo toliau palyginimus vykdyti tik tarp ispanų ir lietuvių kalbos atpažintuvų. Nors kiekviena šių kalbų turi savo skirtybių, pavyzdžiui *š* raidės nebuvimas (kartu su kitais lietuvių kalbos raštmenimis), jų fonetinė sistema yra gerokai artimesnė nei anglų ir lietuvių.

Kalbant konkrečiai apie ligų atpažinimą, jos yra pateiktos ligų sąrašo kataloge TLK-10-AM [27]. Iš viso yra išvardintos 14 179 ligos su kiekvienai jų priskirtu kodu. Pavyzdžiui, ligos „Išeminė kardiomiopatija“ kodas yra I25.5. Turint omenyje, kad tokio kiekio komandų atpažinimo sistemos kūrimo darbai šiuo metu nėra įgyvendinami, buvo apsiribota ties 500-1000 komandų atpažinimo sistemos kūrimo. Remiantis UAB „Softdent“ rinkta informacija apie medicininės terminologijos pavartojimo dažnį, ligų pavadinimai buvo paimti iš tos pačios įmonės tvarkomo tinklapio [29] pagal sergamumo dažnį. Iš viso – 217 ligų pavadinimų.

1.3 Transkribavimas, jo įtaka atpažinimui

Transkribavimas, anot P. Kasparaičio [10], yra vienas pirmųjų sintezės iš teksto etapų. Tai sudėtingas procesas, o reikalingas jis todėl, kad negalima tiksliai perteikti tarimo per raštą ir skyrybą, nes kalboje viena fonema gali turėti kitokią formą ir išraišką. [11]

Rašant lietuvių kalboje, remiamasi tam tikrais rašybos principais:

- *fonologiniu* – rašoma taip kaip tariama, o jei žodžiai skiriasi, jie, be abejo, pažymimi kitomis fonemomis, pvz. *kąsti* (kaip *kanda)* ir *kasti* (kaip *kasa*);
- *morfeminiu* – tas pačias morfemas skirtingose žodžių formose rašome taip pat, nors jos ir tariasi skirtingai, pvz.: tardami žodį *staugti*, girdime *k* vienoje *g*, bet rašome pagal *staugia*;
- *tradicija* – kai kurių žodžių rašyba negali būti paaiškinama taisyklėmis, o tik nusistovėjusiomis tradicijomis, pvz.: nors ir tariame, *j* nerašome žodžiuose *iešmas, ieškoti*.

Transkribuojant lietuvių kalbos žodžius susiduriama su ilgųjų balsių *ą, ę, ė, ų, ū, į* užrašymo sunkumais. Jie žymimi specialiais simboliais, tačiau tai prieinama tik lietuvių kalboje, o siekiant transkribcijas užrašyti užsienio kalboms, tie ženklai nebėra tinkami. Be to, transkribcijos neperteikia kalboje slypinčių intonacijų, pauzių, todėl jos tampa tik sausu kodu kompiuteriui arba tik raštmenimis žmogui. Siekiant suvienodinti trankribavimo simbolius, kuriamos fonetinio transkribavimo sistemos, kurios automatizuoja procesą arba jį palengvina. Dalis yra laisvai prieinamos, tokių sistemų pavyzdžiai – SAMPA (Speech Assesment Method Phonetic Alphabet), Kirshenbaum (dar vadinama ASCII-IPA), Arpabet, X-SAMPA (Extended Speech Assesment Method Phonetic Alphabet). [10]

Dažniausiai viena atpažinimo sistema kuriama vienai kalbai, bet jei tenka ar būtina vieną sistema pritaikyti kitai kalbai, taupant išteklius ir kitais sumetimais, tai padaryti nėra taip jau lengva.

Tiek sistemą, o jei to padaryti neįmanoma, tada kalbą reikia maksimaliai paruošti sąsajai su duota atpažinimo sistema. Kai kuriais atvejais reikalinga gausi transkribuotų žodžių duomenų bazė siekiant užtikrinti sklandų veikimą, todėl automatizuotas duomenų apdorojimas ir paruošimas labai palengvina darbą – būtinas ne tik transkribavimas, bet ir segmentavimas bei anotavimas. Tokios procedūros atliekamos rankiniu būdu užima labai daug laiko, ypač jei transkribuojamas garsynas yra labai didelis, o segmentacija atliekama morfemų lygmeniu. Šiems tikslams įvykdyti naudojamos automatinės sistemos, viena iš jų – nemokamai prieinama, Unix architektūrą palaikanti automatinio, transkribavimo, anotavimo ir segmentavimo programa *Transcribe.* Be jos egzistuoja keletas kitų, pvz., LDC, bet ji remiasi nepatikimais serveriais ar trečiųjų šalių teikiamomis licencijomis, nėra skirtas naudoti ant paprastų vartotojų kompiuterių. Alternatyva, tačiau gana brangiai kainuojanti programinė įranga yra *The Annotator*, sukurta *Entropics*, bet kitas jos trūkumas yra tas, kad ji negali būti lengvai modifikuojama. [2]

Neretai transkribavimo praktika yra gana sudėtinga, anot P. Kasparaičio [10], dažniausiai transkribuojama ir gramatikos sudaromos naudojantis žodynu arba gramatinėmis taisyklėmis, bet pirmuoju metodu reikalingas gana didelės apimties žodynas. Plečiamas transkripcijų sąrašas, pakeičiant tik tam tikras fonemas gali klaidinti sistemą, todėl remianti R. Maskeliūno tyrimu [15], platus tranksripcijų sąrašas nelemia tikslesnio atpažinimo, vietoje to derėtų tikslingai sudaryti jų sąrašus, filtruoti ir atrinkti tas trankripcijas, kurios buvo atpažįstamos dažniausiai. Vienas iš tikslingiausių trankripcijų sudarymo metodų yra *UPS* (*Universal Phone Set*) transkripcijų sudarymas. UPS trankripcijos yra universaliųjų fonemų seka, kurias lengvai supranta kompiuterinės sistemos. Tai SAPI aplinkoje sudaromos transkripcijos. UPS naudoja IPA 1993 Unicode ženklų rinkinį kartu su keliais SAPI ženklais. UPS ženklų sistema sugeneruota pagal didžiųjų ir mažųjų raidžių skirtumus, pavyzdžiui balsiai ir priebalsiai rašomi didžiosiomis ir gali skirtis jų dydis – nuo 1 iki 3 ženklų. Trumpesni imboliai skirti žymėti dažniau pasikartojančias fonemas. UPS simbolių sekos aprašytos ASCII ženklų eilėmis. Anksčiau UPS trankripcijos buvo sudaromos rankiniu būdu, dabar tai galima padaryti pusiau automatiniu būdu, pasitelkiant *Microsoft Speech Server* su *MS Visual Studio* programiniu paketu, kur programa automatiškai sugeneruoja UPS trankripcijas pagal integruotus UPS fonemų sąrašus. Vienas iš jų, ispanų kalbos fonemų lentelė, pateikiamas 3.1 priede.

Taigi apibendrinant, žodžių anotavimas ir tikslingas transkribavimas gali ženkliai pagerinti atpažinimo tikslumą, o apjungus įvairius transkribavimo būdus galima išgauti geresnius rezultatus, bet gramatikų perkrovimas transkripcijomis nebūtinai žada geresnius rezultatus.

1.4 Paslėptieji Markovo modeliai ir HTK paketas

Panašiai, kaip ir bet kurį atpažinimo procesą, kalbos atpažinimą galima išskaidyti į tris pagrindinius etapus: duomenų įvedimą, požymių išskyrimą ir atpažinimą. Kalbos atpažinimo etape nustatoma, kokios fonemos ar žodžiai yra kalbos signale. Tam reikalinga laiko, dažnio ir amplitudinė signalo analizė, kuomet skaičiuojami tam tikrų dažnio juostų energiją charakterizuojantys parametrai.[9]

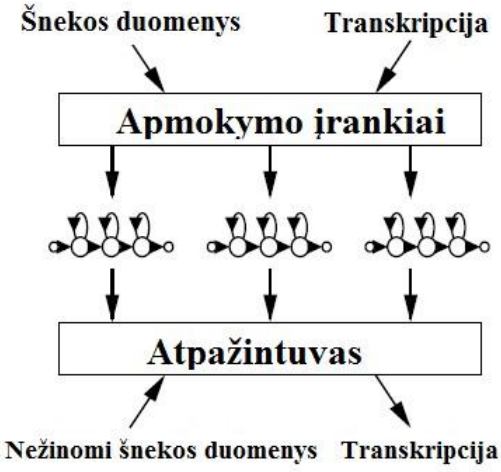
Paslėptųjų Markovo modelių (PMM) taikymo metodika remiasi tikimybiniu modelių sudarymu. Tai statistiniais pavyzdžiais pagrįstas atpažinimo metodas. Šiuo būdu atliekama spektrinė kalbos signalo analizė, kurios metu gaunami požymio vektoriai, kurie aprašo įvairius kalbos garsus. Šie garsų modeliai sujungiami į tinklą, atsižvelgiant į jų tvarką įvairiuose žodžiuose.

PMM naudojami bioinformatikos, automatinio teksto, muzikos generavimo, kalbos atpažinimo, lošimų ir daugelyje kitų sričių. Dėl šio modelio paprastumo jį galima integruoti į įvairius naujus modelius, taip pat įvairiai modifikuoti, priklausomai nuo konkretaus uždavinio.

Daugeliui procesų modeliuoti dėl jų sudėtingumo yra taikomas paslėptasis Markovo modelis. Jis tinka ir šnekos atpažinimo modeliavimui, nes šnekos signalas yra laike kintantis stochastinis (atsitiktinis) procesas. Paslėptasis Markovo modelis apibrėžiamas kaip baigtinė būsenų, susietų perėjimo tikimybėmis, seka, naudojama signalų laikiniam ir spektriniam kitimui modeliuoti. [21]

Vienas iš pagrindinių veiksnių, lėmusių PMM populiarumą sprendžiant šnekos atpažinimo uždavinius, yra mokymosi algoritmų egzistavimas. Mokymui naudojamas iteracinis Baum-Welch algoritmas, su kuriuo turint garsyną, galima surasti modelio parametrus A ir B. [26] Baum-Welch procedūra naudojama Markovo modelio parametrų įvertinimui, atliekant iteracinius tiesioginių-atbulinių (Forward-Backward) tikimybių skaičiavimus. Sintaksė modeliuojama N-gramatikomis, kuriose yra sukaupiamos N paeiliui einančių žodžių statistikos. Atpažinimo procesas grindžiamas Viterbi algoritmu, kai su ištarta fraze dinaminio programavimo būdu lyginami žinomi CD HMM modeliai, surandant panašiausią. [22] Kitaip tariant, Viterbi algoritmas suranda labiausiai tikėtinus kelius Paslėptuose Markovo modeliuose ir išrenka PMM su didžiausia tikimybe.

HTK atpažinimo programų rinkinys yra vienas populiariausių ir plačiausiai naudojamų atpažinimo įrankių. HTK pirmiausiai yra skirtas šnekos apdorojimo priemonių, paremtų paslėptaisiais Markovo modeliais, realizavimui. Yra du pagrindiniai apdorojimo etapai: pirmiausiai, HTK mokymo įrankiai yra naudojami siekiant apytikriai apskaičiuoti paslėptųjų Markovo modelių rinkinio parametrus, naudojant mokymui pateiktus pasakymus (įrašus) ir su jais susijusias transkripcijas; antra, nežinomi pasakymai/įrašai yra transkribuojami naudojant HTK atpažinimo įrankius [32]. Žemiau pateikiama schema, vizualiai apibūdinanti šiuos du svarbiausius apdorojimo etapus.



2 pav. HTK atpažinimo įrankio šnekos apdorojimo schema

Vykdant atpažinimą su HTK paketu, kiekvienam etapui atlikti reikia panaudoti po atskirą programą iš HTK atpažinimo programų rinkinio. Visos programos yra valdomos iš komandinės eilutės per papildomus failus ir eilutės komandas. Žemiau pateikiami pagrindiniai darbo etapai ir jų paaiškinimai su šiuo paketu bei jų įgyvendinimui naudojamų programų pavadinimai:

1. **Žodyno sudarymas.** Kad projektuojama sistema žinotų, kokios žodžių (akustinių vienetų) kombinacijos yra leidžiamos ir kokios ne;
2. **Požymių apskaičiavimas.** Atpažinimo sistemose naudojami ne patys įrašai, o iš jų apskaičiuoti požymiai, vadinasi norint apskaičiuoti požymius pirmiausiai reikia turėti įrašų failus. Požymių skaičiavimui naudojama programa Hcopy;
3. **Modelių apmokymas.** Apmokymas vyksta naudojant Baum-Welch algoritmą, paleidus programą pavadinimu HRest. Svarbus apmokymo etapas – modelių failų kūrimas, kuriuose nurodomi pagrindiniai HMM modelio parametrai. Modelių failai – tai specialaus formato failai, kuriuose bus surašoma atitinkamo modelio parametrai: perėjimų tikimybės ir išėjimų tikimybės. Kadangi HTK operuoja tik tolydiniais HMM modeliais, tai ir išėjimų tikimybės modeliuojamos Gauso skirstiniais. Vadinasi išėjimų tikimybės aprašomos vidurkiais ir standartiniais nuokrypiais;
4. **Atpažinimas.** Atlikus apmokymą ir turint testavimui skirtus įrašus galima atlikti testavimą ir patikrinti kaip gerai vykdomas atpažinimas. Atpažinimui patartina naudoti kitus įrašus negu buvo naudoti apmokymui. Testavimas vykdomas paleidus programą HVite, kuri grindžiama Viterbi algoritmu.

Organizuojant balso komandų atpažinimą gali būti taikomi keli būdai. Vienas iš jų yra modeliuoti kiekvieną garsą pritaikant atskirą Markovo grandinę, po to ieškoma tiksliausiai garsą

atitinkanti grandinių seka – ji turėtų akustiškai prilygti ištartai balso komandai. Antrasis būdas yra naudoti vieną paslėptą Markovo grandinę visai balso komandai. Pirmasis būdas dažniau taikomas nedideliems rišlios šnekos žodynams, nes kiekvieną garsą modeliuoti yra daug sudėtingiau. Be abejo taikyti vienos grandinės modelį ilgam rišliam sakiniui yra dar sudėtingiau, nes tai turėtų būti ištarta be pauzių, o galimų junginių aibė labai išsiplėčia. Tam, kad būtų galima suvaldyti modelių kiekį, modeliuojami monofonai arba trifonai, trijų garsų junginiai, kurie sumažina paieškos ribas. Turint omeny šio projekto ribas, kur bus tiriamos pavienės komandos, kurių kiekis yra ribotas, bus bandoma taikyti vieną paslėptą Markovo grandinę.

2. LIGŲ KODŲ ATPAŽINIMO SISTEMA IR JOS TESTAVIMAS

2.1 Tyrimo metodologija

Siekiant užtikrinti sklandžią darbo eigą, tikslo ir uždavinių pasiekimą ir įvykdymą, reikia nustatyti problemą, sekti ir vykdyti uždavinius bei pasiirnkti tinkamą darbo metodus ir priemones.

Analizės metu taikyti metodai:

1. palyginamasis;
2. aprašomasis;
3. eksperimentinis;
4. kiekybinės analizės;
5. bandymų – klaidų.

Palyginamasis ir aprašomasis metodai buvo taikomi lygiagrečiai – lyginami iki šiol taikyti sprendimo būdai, aprašyti teoriniai standartai. Aprašyti gauti tyrimų rezultatai ir duomenys. Eksperimentų metu atlikti testai, bandymai, kurių metu gauti rezultatai buvo aprašyti analizei skirtoje skiltyje. Buvo eksperimentuojama su įvairiais vartotjų profiliais, keičiant eksperimentų parinktis – Gauso mišinių kiekius, būsenų skaičius. Kai kurių eksperimentų eiga buvo sąmoningai koreguojama vienokia ar kitokia linkme siekiant įžvelgti pokyčius, priklausomybes ar tendencijas.

Eksperimentų rezultatai buvo lyginami tarpusavyje, siekiant geriausių ir blogiausių rezultatų. Visi tyrimai atlikti KTU kalbos signalų tyrimo laboratorijos parengtais ir pritaikytais metodais, programomis ar jų paketais. Testavimai ir analizė atliekami tęsiant tyrimus ties jau įdiktuotu ir sudarytu ligų pavadinimų garsynu, kuris buvo plėstas ir saugotas KTU ITPI kalbos signalų tyrimo laboratorijoje. Garsynas, kaip ir dauguma analizės programų buvo sukurti dirbant ties hibridine atpažinimo technologija balso sąsajai. Toliau šiame skyriuje bus aprašomi įrankiai, kurie naudoti atliekant šį tyrimą.

Hibridinis atpažintuvas. Šio atpažintuvo tikslas yra lygiagrečiai panaudoti du skirtingus atpažintuvus – lietuvišką ir ispanišką. Ispanų kalbos atpažintuvas išsprendžią viena didžiausių lietuviško atpažintuvo problemų – pradžios ir pabaigos aptikimą, mat ispanų atpažintuvas turi efektyvų signalo aptikimo bloką. Hibridinį atpažintuvą sudaro ispanų atpažintuvas, su integruotu signalo aptikimo bloku, lietuvių kalbos atpažintuvas ir sprendimų priėmimo blokas, kuriuo naudojantis realizuojama sprendimo priėmimo taisyklė.

Skaičių pavadinimų garsynas. Atliekant skaičių atpažinimo tyrimą, buvo naudotas garsynas sudarytas dviejų studentų grupių, HKL-6 ir HKL-8, sudarytas iš 30 diktorių įrašų – 23 moteriškosios lyties ir 7 vyriškosios lyties diktorių. Garsynas sudarytas kiekvienam diktoriui ištariant skaitmenų

pavadinimus nuo nulio iki devynių po 20 kartų kiekvieną jų. Kiekvieno diktoriaus įrašai saugomi atskiruose kataloguose, kurių pavadinimai sudaryti iš 7 raidžių. Pirmoji, F arba M žymi diktoriaus lytį, kitos 6 po tris vardo ir pavardės pirmąsias raides, pvz., *FUGNNOV*. Visi diktorių identifikavimo katalogai talpina atskirus kiekvieno skaičiaus pakatalogį. Juose yra:

a) audio failas „*mano.voc*“ (16000Hz, 16 bitų mono audio formatas) ir tas pats failas *.wav* formate, „*mano.wav*“;
b) anotacijų failas, „*mano.zgl*“ su kiekvieno įrašo pradžios ir pabaigos adresais;
c) pagalbinis failas, kuris aprašo garsinio failo turinį ir „*mano.txt*“, kuriame nurodoma, kiek kartų yra ištartas tam tikras skaičiaus pavadinimas;
d) kiti pagalbiniai failai, naudoti garsyno rinkimo ir tikrinimo metu.

Taigi šį garsyną sudaro 6000 skirtingų balso komandų įrašų.

Papildomai naudotas garsyno katalogas, pavadinimu *CORPORA_11*, sudarytas dešimties vyriškosios lyties diktorių. Pastarasis garsynas buvo naudojamas testuotant komandų atpažinimą su ispanų kalbos atpažintuvu naudojant *Microsoft Speech recognizer 8.0 for Windows* sinchroniniame režime, apie kurį plačiau papasakosiu kitame skyriuje.

Vardų garsynas. Pagrindinė fundamentalioji atliekamų eksperimentų dalis yra daugiadiktorinis garsynas, kuris buvo įdiktuotas dvidešimt vieno diktoriaus, 12 moterų ir 9 vyrų. Kiekviena komanda, kaip ir skaičių pavadinimų atveju, buvo ištarta po 20 kartų, taigi iš viso yra 10920 komandų ištarimų. Visi jie yra suskirstyti į 26 katalogus kiekvienam vardui ar žodžiui, kur kiekvienas talpina tokią informaciją:

a) pagalbinį failą *mano.txt*, su informacija apie ištarimų kiekį;
b) audio failą *mano.voc* (16000 Hz, 16 bitų mono audio formatas) ir tas pats *.wav* formato failas *mano.wav*;
c) anotacijų failas *mano.zgl*, kuriame pateikti kiekvieno įrašo pradžios ir pabaigos adresai;
d) pagalbiniai failai, naudoti garsyno rinkimo ir tikrinimo metu.

Kiekvienos komandos įrašų katalogai sudaryti tuo pačiu principu kaip ir skaičių pavadinimų garsynas – pavadinimai iš 7 raidžių, kur pirmoji žymį diktoriaus lytį, o likusios šešios atitinka pirmasias tris vardo ir pavardės raides, pokatalogiai pavadinti kiekvienos komandos vardu.

Vardų atpažinimo atveju taip pat naudotos gramatikos su jose esančiomis transkripcijomis. Jos sudarytos remiantis ta pačia metodologija ir gairėmis, kurios buvo taikytos sudarant skaičių pavadinimų gramatikas – tai aptarta skyrelyje 2.2.2.

Gramatikos, transkripcijos. Kompiuteriui neturint galimybės interpretuoti standartinių raštmenų ir pritaikyti jų žodinių komandų atpažinimui, turi būti paruošiami specialūs transkripcijų failai, gramatikos. (žr. 5 priedą). Duomenų faile yra pateikiami keli transkripcijų variantai ir taikoma komanda „one of“ – atpažintuvas tikrina kiekvieną transkripciją, radus geriausią atitikmenį, pasirenkama viena iš jų (geriausia naudoti tuomet, kai tiriama, kuri tranksripcija buvo atpažįstama dažniausiai). Tyrimo metu bus naudojamos gramatikos vestos naudojant transkripcijas be USP atributų ir su UPS atributais

Ligų kodai. Remiantis dideliu ligų skaičiumi ir ribotu ištekliu kiekiu, visų ligų pavadinimų (jų yra daugiau nei 10000) garsinės duomenų bazės paruošimas būtų labai sudėtingas. Kita vertus, jas klasifikuoti pagal kodą yra daug paprasčiau. Ligos kodą, pagal TLK-10-AM [27] gali sudaryti simbolis, raidė, skaičiai ir ženklai, pavyzdžiui *A17.1* ar *B37.88*. Klasifikacija dar vykdoma naudojant 4 skirtingus simbolius – kryželį, žvaigždutę, atverstą trikampį ar du kartus įstryžai perbrauktą rutuliuką. Kiekvienas simbolis nurodo kodo koduotę ar sukelėją. Bet turint omeny, kad daugumą sistemų kodus atpažįsta pagal raidės ir skaičių kombinacijas, bei siekiant tikslesnių testų rezultatų, bus naudojami tik raidės ir keturių skaitmenų kodai, be taškų. 3.5 priede yra pateikiama ligos kodo sandaros lentelė ir kodo raidžių reikšmės.

HTK paketo optimizavimo *.bat* failai. HTK paketo veikimo principas grįstas keliomis programomis, kurios atlieka savo funkciją atpažinimo sistemos kūrimo metu – HCopy, HRest, HVite. Paleidžiant kiekvieną komandą, naudojant *Far commander* failų ir duomenų valdymo įrankį, prie jų įrašomi parametrai. KTU ITPI buvo parengti *batch* failai, komandų sąrašai, kurie paspartina darbą su HTK paketo programomis ir suteikia daugiau laiko pačiam tyrimui. Tad trumpai apie kiekvieną:

Parse.bat: paleidus failą, automatiškai į komandos eilutę įvedamas tekstas „HParse.exe gram.dict wordnet.txt“ – paleidžiamas HPartse.exe failas, kuris sukuria wordnet failą iš gramatikų failo, kruis naudojamas atpažinimui.

Pozym.bat: į komandos eilutę įrašomas tekstas „HCopy -C CONFIG -S failai.scp“, kuris iškviečia HCopy HTK paketo programą, kuri naudodama tekstinį failą „failai.scp“, kuriame nurodomas virtualus kelias iš garsinio failo ir nurodo naujai sukurto požymių failo saugojimo vietą.

Apmok_be_labels.bat: į komandos eilutę įveda „HRest.exe -T 1 -S trans_austeja.scp hmm_austeja>>kk1“. Ši komanda naudodama HRest programą ir parinktus parametrus iš požymių failų *.mfc* sudaro naujus *kk* failus su numeracija nuo 1 iki 26 (kiekvienam iš vardų), kurie vėliau naudojami atpažinimui.

Pozym_test.bat: įvedamas tekstas „HCopy -C CONFIG -S test.scp“ aktyvuoja HCopy programą ir sukuriamas *test.scp* failas, kuriame saugoma informacija apie garsinių ir požymių failų talpinimo vietą.

Atpaz.bat: iškviečia HVite komadą (HVite -T 1 -S test_mfc.scp -i results -w wordnet.txt dict hmmlist), nukreipią programą į požymių failus, wordnet failą ir atlieka vidinius komandų atpažinimo testavimus.

HResults.bat: šis failas iškviečia programą Hresults (HResults -p -I testref.mlf dict recout.mlf>>rez) kuri sukaičiuoja gautus atpažinimo rezultatus ir apibendrintus rezultatus, su atpažinimo tikslumo procentinėmis išraiškomis pateikia tekstiniame faile.

2.2 Transkripcijų sudarymas

Kiekvienai bet kokiai atpažinimo sistemai reikia paruošti sistemos „gramatiką“. Gramatikos sudarymas yra labai svarbus etapas, kadangi tai yra specialus failas, kuriame nurodoma, kaip tam tikra komanda (žodis) turėtų būti ištarta — tai vadinamosios transkripcijos, ir ką reikėtų išvesti į ekraną įvykus atpažinimui. Pirminės gramatikos sudarymui transkripcijos rašytos intuityviai, nesiremiant jokiais pagalbiniais transkripcijų generavimo įrankiais, beveik visos komandos aprašytos taip, kaip tariamos įprastai, tik nenaudojami lietuviški simboliai. Tokiu būdu siekiama patikrinti, kiek ir kokiu būdu galima pagerinti atpažinimo kokybę. Žemiau pateikiama ištrauka iš pirminės gramatikos failo:

<item><one-of><item> ehiymahntahs</item>
<item> eimantas</item>
</one-of><tag>$._value = "eimantas"</tag></item>

Antroji gramatika sudaroma pildant ją naujomis transkripcijomis, pasinaudojant sintezatoriumi, laisvai prieinamu internete, kurio kūrėjai yra *Smart Link Corporation*, svetainėje http://text-to-speech.imtranslator.net/. Sintezatorius veikia *text-to-speech* principu. Transkripcijos kuriamos nesiremiant jokiu tiksliu metodu, žodis bandomas užrašyti įvairiais būdais, perklausomas, kaip jį ištaria sintezatorius, tobulinamas iki panašiausio ištarimo į užrašymą, o tada panaudojamos gramatikoje kaip alternatyvos. Taip atrodo ištrauka iš patobulintos gramatikos:

```
<item>                                  <one-of>
        <item>kyuh</item>
        <item>kju</item>
        <item>qhju</item>
        <item>khju</item>
        <item>kjjhu</item>
        <item>kjjhju</item>
        <item>kjjhju</item>
        </one-of>
        <tag>$._value = "kju"</tag>
</item>
```

Trečioji gramatika sudarinėjama įtraukiant UPS transkripcijas. Jos generuojamos automatiniu būdu, panaudojant balso serverį ir jame įdiegtą *Microsoft Visual studio 2005* programą. Šiam darbui panaudojama antroji gramatika, su naujomis alternatyviomis transkripcijomis, kurios pakeičiamos UPS atributais. Sistema pateikia vieną arba daugiau galimų variantų, o vartotojas išsirenka norimas sugeneruotas transkripcijas ir jas įtraukia į gramatiką (šiame tyrime įtraukti visi sistemos siūlomi variantai). Žemiau pateikiama ištrauka iš UPS gramatikos:

```
<item>
        <?MS_Grammar_Editor GroupWrap?>
        <token sapi:pron="B A S I S;B A K I S;B A K J S">vacys</token>
        <tag>$._value = "vacys"</tag>
    </item>
    <item>
        <?MS_Grammar_Editor GroupWrap?>
        <token sapi:pron="B A T S I I S;B A T S J J S;B A T S J E J S;B A T S I J S;B A T S J I S">vahtsiys</token>
        <tag>$._value = "vahtsiys"</tag>
    </item>
```

Tyrimo metu bus naudojamos visos gramatikos ir stebimi testavimų pokyčiai, siekiant išgauti kuo aukštesnę atpažinimo kokybę.

2.3 Darbo su HTK paketu principai

Testavimų rezultatų palyginimui bus sudaromas atpažintuvas veikianti Paslėptų Markovo grandinių principu, naudojantis HTK paketo programų rinkiniu. Atlikus šiuos darbus, gauti testų rezultatai bus lyginami su gautais testų su kitakalbiu atpažintuvu metu.

Testavimui naudojami šešių diktorių garso įrašai – *FIEVVIS, FJUSKIN, FUGNBUC, FUGNNOV, FVISSTA, MLINJUR.* Likusiųjų 24 diktorių yrašai naudoti sistemos apmokymui.

Darbo našumui gerinti buvo naudota failų tvarkymo programa *„Far Commander 3“* – ji skirta atlikti duomenų tvarkymo darbus, komandinių eilučių įvedimui. Programų aktyvavimui naudoti *batch* failai, apie kuriuos smulkiau aprašyta metodinėje dalyje, skyriuje 2.1.

Toliau aprašysiu, kaip ir kokia programų ar failų sukūrimo tvarka buvo taikyta procese.

Pirmiausia sukuriamas *gram.dict* failas, kuriame nurodoma komandų tvarka, kuri bus naudojama atpažįstant. Failo turinys atrodo taip:

```
$word=nulis|vienas|du|trys|keturi|penki|sesi|septyni|astuoni|devyni;
($word)
```

Komandų atskyrimui naudojamas statmenas brūkšnys, kuris programos viduje reiškia, kad atpažįstamos atskiros komandos, o ne jų junginiai.

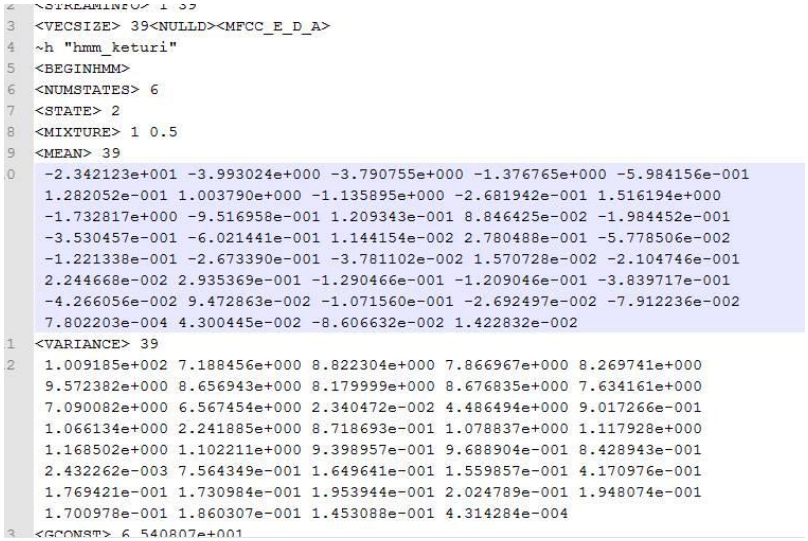
Naudojantis *HParse.exe* sukuriamas failas *wordnet.txt*. Šiam procesui naudojamas *parse.bat* komandinis failas, kuris įrašo į komandų juostą tokį tekstą - *HParse.exe gram.dict wordnet.txt*.

Atpažintuvui sukurti, reikalingas failas *failai.scp*, kuris naudojamas požymiams apskaičiuoti ir sukurti požymiams. Jo viduje nurodoma kompiuterio viduje talpinamų įrašų direktorija ir atitinkama direktorija, kur turėtų būti išsaugoti požymių failai. Failo turinio ištrauka:

C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F11.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F11.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F12.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F12.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F13.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F13.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F14.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F14.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F15.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F15.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F16.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F16.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F17.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F17.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F18.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F18.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0F19.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0F19.mfc
C:\HTK\HTK_SKAICIAI2\FJUSKIN\0G00.wav C:\HTK\HTK_POZYMIAI2\FJUSKIN\0G00.mfc

Šių failų sukūrimas reikalingas dėl atpažintuvo veikimo principo – jis nenaudoja pačių garsinių failų, o tik juose esančius garsų požymius (failai *.mfc*). taigi pats požymių sukūrimas vykdomas naudojant programą *HCopy.exe,* o *batch* failo komandos iškvietimo eilutė atrodo taip: *HCopy -C CONFIG -S failai.scp.*, čia nurodomas failas *CONFIG* su informacija apie tai, kaip turėtų būti skaičiuojami požymiai.

Kuriant modelių failus, kuriuos atpažintuvas naudos apmokymui. Sukuriama 10 modelių failų kiekvienam skaičiui, pvz.: *hmm_nulis, hmm_vienas* ir t.t. Modelių failuose talpinama struktūrizuota informacija apie žodžius, Gauso mišinių kiekį, būsenų skaičių; aprašyti vidurkiai ir dispersijos, perėjimo tikimybės ir kita informacija. 3 paveiksle pateikiama *hmm_keturi* modelių failo turinio ištrauka.



3 pav. *Hmm_keturi* modelių failo turinio ištrauka

Apibūdinsiu raktinius modelių failo parametrus:

<*NUMSTATES*> parametras apibūdina, koks būsenų skaičius bus naudojamas modelių faile. Šiame pavydyje jis atitinka raidžių skaičių, todėl galima sakyti, kad būsenų skaičius yra +0.

Eilutėje <*MIXTURE*> nurodoma, kuris Gauso mišinys naudojamas. Jei naudojama daugiau nei 1, eilutėje <*NUMMIXES*> būtų nurodytas Gauso mišinių skaičius.

Paskutinė būsena vadinama transponente ir yra žymima <*TRANSP*>, o joje sudaroma tikimybių matrica:

0.000000e+000 1.000000e+000 0.000000e+000 0.000000e+000
0.000000e+000 9.719329e-001 2.806702e-002 0.000000e+000
0.000000e+000 0.000000e+000 9.685927e-001 3.140728e-002
0.000000e+000 0.000000e+000 0.000000e+000 0.000000e+000

Matricoje eilučių ir stulpelių skaičius turi sutapti, be to, prie parametro <*TRANSP*> nurodomas skaičius žymintis paskutinės būsenos numerį. Judėjimas atgal negalimas, todėl programoje skaičiavimai juda į priekį, link kitos būsenos, arba lieka toje pačioje. Vidurkiai <*MEAN*> ir dispersijos <VARIANCE> vertės yra nuolatos perskaičiuojami.

Keičiant tyrimų eigą, pridedant Gauso mišinių ar keičiant būsenų skaičių, šie failai buvo nuolatos koreguojami. Tad atlikus reikalingus pakeitimus, buvo vykdomas modelių apmokymas. Tam buvo kuriami *script* failai –*.scp*, kurių sudarymo procesas analogiškas *failai.scp* sudarymo procesui – nurodomos direktorijos, kur talpinami požymių failai ir nurodyti, kur bus saugomi apmokymo failai, tokie kaip *trans_du.scp, trans_trys.scp* ir t.t. vienos iš failo turinio fragmentas:

C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA901.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA902.mfc

C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA903.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA904.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA905.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA906.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA907.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA908.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA909.mfc
C:\HTK\HTK_SKAICIAI2\POZYMIAI2\FVISSTA910.mfc

Po to, kai sukuriami šie failai visų skaičių pavadinimams, galima vykdyti modelių apmokymą. Tam naudojama programa *HRest.exe*. kartu su jos paleidimu, komandinėje eilutėje reikia kas kartą nurodyti tokią komandą - *HRest.exe -T 1 -S trans_nulis.scp hmm_nulis>>kk1* – su kiekvieno skaičiaus pavadinimu. Šis procesas optimizuojamas paleidžiant *batch* failą, kuris į komandos eilutę išveda tokį tekstą:

HRest.exe -T 1 -S trans_nulis.scp hmm_nulis>>kk1
HRest.exe -T 1 -S trans_vienas.scp hmm_vienas>>kk2
HRest.exe -T 1 -S trans_du.scp hmm_du>>kk3
HRest.exe -T 1 -S trans_trys.scp hmm_trys>>kk4
HRest.exe -T 1 -S trans_keturi.scp hmm_keturi>>kk5
HRest.exe -T 1 -S trans_penki.scp hmm_penki>>kk6
HRest.exe -T 1 -S trans_sesi.scp hmm_sesi>>kk7
HRest.exe -T 1 -S trans_septyni.scp hmm_septyni>>kk8
HRest.exe -T 1 -S trans_astuoni.scp hmm_astuoni>>kk9
HRest.exe -T 1 -S trans_devyni.scp hmm_devyni>>kk10

Šiame procese naudojami modelių failai ir *script* tipo failai su atitinkamais parametrais. Tokiu būdu gaunami jau apmokyti pagalbiniai *kk** failai, kurie bus pritaikyti testavimo metu.

Testavimo procese reikalingi nauji požymiai, tad jų failai bus bus saugomi *script* formatu. Sukuriamas *test.scp* failas su direktorijų nuorodomis, kaip *failai.scp*. Naudojant *Hcopy.exe* programą, sukuriami nauji, testavimui skirtų diktorių garso įrašų, požymių failai.sukurtus požymių failų pavadinimus surašome į *test_mfc.scp* failą.

Modelių failų pavadinimai surašomi į *hmmlist* failą, o tiriami žodžiai surašomi į *dikt.txt* – jo turinys bus atvaizduojamas atpažinus komandą.

Po šių veiksmų atlikimo, vykdomas testavimas – paleidžiama *Hvite.exe* programa su *batch* faile nurodoma komandine eilute: *HVite -T 1 -S test_mfc.scp -i results -w wordnet.txt dict hmmlist*. Sukuriamas *results* failas, kuri pervadinamas *recout.mlf* tam, kad pasitelkus *batch* faile nurodytą komandinę eilutę, rezultatai būtų automatiškai suskaičiuoti ir išvesti į tekstinį failą *rez*, kur jau gauti testų rezultatai apibendrinami ir naudojami apibendrinimui, palyginimui.

2.4 Kitakalbio atpažintuvo tyrimai

Remiantis analitinėje dalyje sukaupta informacija, pradiniai tyrimai atliekami naudojant kitakalbį ispanų kalbos atpažintuvą. Atlikta eilė testų, kuriais remiantis buvo siekiama išsiaiškinti tiek vartotojo profilio įtaka rezultatams, tiek apmokymo nauda bei transkripcijų sudarymo principai. Ligos kodas susideda iš raidės ir 3-4 skaitmenų, todėl balsu įvedamos komandos bus atpžįstamos remiantis dviemis gramatikomis ir garsynais. Remiantis tuo reikalingi du atskiri tyrimai, kurie nustatytu, koks geriausias kiekvienos raidės ir skaičiaus atpažinimo būdas.

2.4.1 Skaičių pavadinimų atpažinimo tyrimas

Šiame skyriuje aprašomi rezultatai, kurie gauti atliekant skaičių atpažinimo tyrimus ir eksperimentus su paruoštu garsynu.

Skaičių pavadinimų atpažinimo testavimas sinchroninėje veiksenoje. Atliekant komandų testavimus bus naudojamos sinchroninė ir asinchroninė veiksenos. Pirmoji yra pilnai automatizuota – tiriamosios duomenų eilutės į testavimo srautą yra pateikiamos po vieną, tai reiškia, kad kiekviena komanda yra tikrinama atskirai nuo kitų, taip perteikiant procrso eigą.

Testams atlikti buvo naudojama KTU ITPI sukurta programa skirta būtent sinchroninės veiksenos testavimams, ***cWavoDalis.exe,*** kuri yra tiesiogiai susieta su *CORPORA_11* katalogu. Dėl tokios sąsajos ir neesančios prieigos prie programos modifikavimo failų, tris kartus kečiant pirmųjų dešimties diktorių failus kitais dešimt, buvo atliktas pilnas komandų testavimas, kas kart iš naujo išsaugant gautą rezultatų failą. Sinchroninės veiksenos rezultatai pateikiami 1 lentelėje.

1 lentelė. Skaičių atpažinimo tiksumas sinchronine veiksena

Skaičių pavadinimai	Atpažinimo tikslumas	
	Teisingai atpažinta komandų	Išraiška, %
Nulis	424	70,67
Vienas	580	96,67
Du	416	69,33
Trys	594	99,00
Keturi	514	85,67
Penki	594	99,00
Šeši	600	100,00
Septyni	595	99,17
Aštuoni	598	99,67
Devyni	432	72,00
Iš viso:	5347	89,12

Iš lentelėje pateiktų duomenų matome, kad visų komandų atpažinimo vidurkis yra beveik 90%, kas rodo tikrai ne prastą atpažinimo tikslumą. Geriausiai atpažinta komanda yra *šeši*, jos atpažinimo tikslumas yra 100%. Keturios komandos, tai *trys, penki, septyni* ir *aštuoni* priartėjo prie 100% ribos ir jų atpažinimo tikslumas yra lygus arba viršyja 99%. Prasčiausiai atpažinta komanda yra *du* – vos 69,33% tikslumu.

Siekiant kokybiškesnių rezultatų, atlikti papildomi bandymai sinchroninėje veiksenoje.

Skaičių pavadinimų atpažinimo testavimas asinchroninėje veiksenoje. Priešingai nei sinchroninėje veiksenoje, ši nėra automatizuota. Reikalingi visi įvesties duomenys, be to, visi veiksmai atliekami rankiniu būdu – komandiniai failai yra paleidžiami po viena, reikia laukti rezultatų ir juos savarankiškai reziumuoti.

Tam, kad pradėtume testavimą, reikia specialiai paruošti failus. Pirmiausia sukuriamas *mano.voc* failas, kuriame talpinama informacija apie visus vieno diktoriaus komandos ištarimus, toliau buvo sukurti *.wav* failai naudojant *Voc to Wav.exe* programėlę, kuri automatiškai konvertuoja *.voc* failus į garsinius *.wav* failus. Testui atlikti buvo naudota *Rec_UPS.exe* programa. Į ja užkraunami testiniai failai, programos langelyje atsirada visų komandų sąrašas. Išvesties faile pateikiamos atpažintos transkripcijos ir rankiniu būdu reikia suskaičiuoti, kiek komandų buvo atpažinta teisingai. Jei komanda neatpažįstama – jos eilutė būna tuščia, kitais atejais parenkama ne ta komanda, pavyzdžiui, sumaišoma komanda ir pateikiama kito skaičiaus transkripcija. Asinchroninio režimo testavimo rezultatai pateikiami 2 lentelėje.

2 lentelė. Skaičių atpažinimo tiksumas asinchronine veiksena

Skaičių pavadinimai	Atpažinimo tikslumas	
	Teisingai atpažinta komandų	Išraiška, %
Nulis	545	90,83
Vienas	580	96,67
Du	516	86,00
Trys	587	97,83
Keturi	469	78,17
Penki	588	98,00
Šeši	595	99,17
Septyni	600	100,00
Aštuoni	598	99,67
Devyni	544	90,67
Iš viso:	5622	93,70

Iš lentelėje pateiktų duomenų matome, kad bendras atpažinimo tikslumas lygus 93,7 %, tai yra beveik 4,5 % daugiau nei sinchroninės veiksenos testo metu. Geriausiai atpažįstama komanda tapo

septyni, kitos, geriausiai atpažįstamos yra *šeši* ir *aštuoni*, perkopė 99 % ribą, bet jos yra tik 2, lyginant su sinchroninio testo rezultatais, kur tokią ribą pasiekė net 4 komandos iš dešimties. Favoritės išliko tos pačios – komandos *try* ir *penki* liko geriausiųjų penketuke, nors atpažinimo tikslumas sumažėjo. Prasčiausia rezultatą pasiekė komanda *keturi*, atpažinimo tikslumas siekia 78,17 %. Tad, kaip matome, pasikeitė prasčiausioji komanda, be to pakilo minimali atpažinimo tikslumo riba. Asinchroninio testo metu, prieš tai sinchroninėje veiksenoje surinkusi vos 69 %, komanda *du* pasiekė 86 % atpažinimo tikslumą.

2.4.2 Lietuviškų vardų atpažinimo tyrimas

Antroji svarbiausioji ligos kodo dalis yra raidė. Dėl fonetinės sandaros, neaiškios komandos pradžios ir pabaigos, tik raidžių tarimas yra sunkiai fiksuojamas, todėl pasirinkta kiekvienai alfabeto raidei priskirti po vardą ar žodį (žr. 3.2 priedą) ir vietoje raidės tarti būtent jį. Tolimesniuose skyriuose bus aptarti atliktų tyrimų rezultatai.

Balso ir apmokymo įtaka atpažinimo su kitakalbiu atpažintuvu tikslumui. Šio tyrimo metu bandoma nustatyti, ar diktoriaus, apmokiusio testavimo profilį, lytis turi įtakos atpažinimo tikslumui. Be to, į tyrimą įjungta ir UPS transkripcijų panauda atpažinime. Tirsime, ar UPS transkripcijų panaudojimas turi įtakos atpažinimo tikslumui. Tyrimai vykdyti su ispanų kalbos atpažintuvu.

Sukurti trys profiliai, specialiai šiam testui – *vyr_zod, mot_zod, Default_zod*. Testuojant su jais, bus siekiama nustatyti apmokymo svarbą izoliuotų komandų atpažinimui. *Vyr_zod* profilis yra apmokytas vyriškos lyties diktoriaus, atitinkamai *mot_zod* – moteriškosios lyties diktoriaus. *Default_zod* profilis nebuvo visiškai apmokomas, todėl testas turėtų parodyti ir tai, ar apmokymas tikrai reikalingas, o ne tik apmokytojo lyties reikšmingumą. Papildomas testas, profilio *vyr_UPS* testas turėtų pateikti duomenų apie UPS atributų svarbą.

Atlikus automatizuotus testus su pateiktomis vardų ir žodžių komandomis sinchronine veiksena (pasirinkimas pagal nutylėjimą, nes testas veikia automatiniu rėžimu), gauti rezultatai susumuoti lentelėje, kuri pateikta 3.6 priede. Žemiau pateikiami reziumuoti rezultatai. Įvykdytas rezultatų, o tiksliau diktorių rūšiavimas pagal 10 geriausiai komandas atpažįstančių diktorių rezultatus kiekvienam pofiliui (žr. 3 lentelę).

3 lentelė. Geriausių diktorių išsidėstymas pagal atpažinimo tikslumą

	vyr_zod	Atpaž., %	mot_zod	Atpaž., %	Default_zod	Atpaž., %	vyr_UPS	Atpaž., %
1	MANDMAR	99,7	MDARJEG	98,2	MSARNEM	99,7	MDARJEG	97,2
2	MSARNEM	99,7	FIVEVAL	98	MDARJEG	99,5	MJURBIZ	97
3	MJURBIZ	99,2	FEGLZAJ	97,9	MANDMAR	99,1	MJUOCES	96,6
4	FGRETUB	98,8	MSARNEM	97,7	FEGLZAJ	99	FLAUKLU	96,6
5	MZYGSVE	98,8	FGRETUB	96,3	FGINBAR	99	MANDMAR	96,1
6	MDARJEG	98,8	MANDMAR	96	MJURBIZ	97,8	FIVEVAL	96,1
7	FGINBAR	98,6	FJULBAL	95,7	FGRETUB	96,9	MZYGSVE	96
8	FLAUKLU	97,8	FGINBAR	93	FIVEVAL	96,9	MSARNEM	95,8
9	FEGLZAJ	97,6	FLAUKLU	92	FSEVBUT	96,3	FEGLZAJ	95,1
10	MJUOCES	97,1	FGINTRA	90,2	MJUOCES	96,1	FGINBAR	95,1

Pateiktoje lentelėje matome, kad rezultatai pasiskirstę labai apylygiai ir nevisiškai taip, kaip buvo tikėtasi. Testuojant vyriškąjį profilį *vyr_zod* tikėtasi, kad vyriškosios lyties diktorių komandos bus atpažįstamos geriausiai – iš dalies taip ir buvo. Pirmieji trys diktoriai yra vyriškosios lyties ir jų atpažinimo tikslumas svyruoja tarp 99,2 ir 99,7 nuošimčių. Visgi bendraja prasme rezultatai pasiskirstę tolygiai – tarp 10 geriausiai atpažįstamų diktorių šeši yra vyriškosios lyties.

Moters balsu apmokyto profilio tyrimo rezultatai rodo, kad geriausia buvo atpažįstamos vyriškosios giminės diktoriaus komandos – *MDARJEG,* bet jo atpažinimo tikslumas tesiekia 98,2% ir tai tėra vos 1,1% daugiau nei prasčiausias vyro balsu apmokyto profilio rezultatas. Likusieji rezultatai ir diktoriai pasiskirstę labai panašiai kaip ir prieš tai buvusio testo metu – iš dešimties geriausiųjų yra 6 moteriškosios lyties atstovės.

Lyginant rezultatus gautus testuojant su neapmokytu profiliu nustatyta, kad pirmieji geriausi diktorių rezultatai priklauso vyriškosios giminės diktoriams, o aukščiausias atpažinimo tikslumas siekia 99,7% - tiek pat kiek pasiekta testuojant su vyro balsu apmokytu profiliu. Bendras rezultatų pasiskirstymas yra lygus – moterų ir vyrų diktorių skaičius tarp 10 geriausių yra lygus – 5 ir 5.

Testo metu naudojant UPS transkripcijas pastebėta, kad aukščiausią atpažinimo tikslumą pasiekė vyriškosios lyties diktoriai – jie yra pirmieji trys, o atpažinimas svyruoja tarp 96,6 ir 97,2 nuošimčių. Diktorių lyčių pasiskirstymas taip pat gana tolygus – 6 vyriškosios lyties diktoriai iš dešimties geriausiųjų.

Iš gautų rezultatų galima daryti prielaidą, kad atpažinimo tikslumas mažai susijęs su tuo, kieno balsu ir ar iš viso buvo apmokytas profilis. Aukščiausi rezultatai pasiekti vyrų diktorių, nepriklausomai

nuo profilio ir nuo to, kas jį ir ar iš viso apmokė. UPS transkripcijų pateikimas pasitarnavo vyriškosios lyties diktorių labui, nes geriausi rezultatai gauti taip pat jų. Be to, verta paminėti, kad bendras visų profilių atpažinimo tikslumo vidurkis taip pat labai panašus, mažiausias rezultatas pasiektas testuojant su UPS tranksripcijomis papildytu profiliu (94,12%), o aukščiausias rezutatas priklauso vyrišku balsu apmokytam profiliui *vyr_zod*, jo rezultatų vidurkis yra 96,44%. Likusiųjų profilių rezutatai atsiduria intervale tarp šių dviejų dydžių.

Išsiaiškinus, kokių diktorių įrašai yra atpažįstami geriausiai, idomu nustatyt, kurios komandos buvo atpažįstamos geriausiai ir blogiausiai. Šiam uždaviniui įgyventindi sudaryta nauja lentelė (žr. 3.7 priedą). Rezutatų palyginimui pasirinktas mažesnis duomenų kiekis – 10 geriausių komandų pagal profilį ir atpažinimo tikslumą (žr. 4 lentelę).

4 lentelė. Geriausiai atpažįstamos komandos pagal profilį

	vyr_zod	**Atpaž., %**	**mot_zod**	**Atpaž., %**	**Default_zod**	**Atpaž., %**	**vyr_UPS**	**Atpaž., %**
1	Oskaras	100	Karolis	100	Karolis	100	Hansas	100
2	Patrikas	99,9	Izaokas	99,5	Oskaras	100	Vacys	100
3	Donatas	99,9	Martynas	99,5	Patrikas	99,9	Donatas	99,9
4	Cecilija	99,9	Oskaras	99	Sandra	99,8	Martynas	99,9
5	Sandra	99,6	Patrikas	99	Teodoras	99,8	Patrikas	99,8
6	Gražvydas	99,3	Hansas	98,9	Laima	99,3	Sandra	99,8
7	Austėja	99,3	Eimantas	98,8	Nojus	99,1	Austėja	99,5
8	Teodoras	99,1	Donatas	98,2	Hansas	98,9	Cecilija	99,3
9	Hansas	99	Faustas	98,2	Eimantas	98,9	Laima	99,2
10	Boleslovas	98,9	Sandra	98,2	Gražvydas	98,9	Izaokas	99,2

Iš pateiktos lentelės duomenų galima iš karto pasakyti, kad bendras atpažinimo tikslumas yra tikrai aukštas – visų profilių mažiausias atpažinimo tikslumas, imant geriausiųjų komandų dešimtuką, yra net 98,2%, o aukščiausias, be abejo, 100% atpažinimo tikslumas.

Lyginant vyrišku balsu apmokyto profilio su žodinėmis transkripcijomis rezultatus galima matyti, kad atpažinimas nenukrenta žemiau 98,9% lyginant 10 geriausiai atpažintų komandų. Geriausiai atpažinti vardai yra *Oskaras, Patrikas, Donatas*.

Analizuojant rezultatus gautus su moters balsu apmokytu profiliu su žodinėmis transkripcijomis matome, kad atpažinimo tikslumas mažai skiriasi nuo prieš tai aptarto profilio. Skirtumas tas, kad geriausiųjų vietas užėmė kitos komandos – *Karolis, Izaokas, Martynas*. Nei vienos iš šių komandų nėra pirmųjų geriausiųjų pozicijose *vyr_zod* profilio rezultatų suvestinėje. Negana to, prieš tai geriausiai atpažintos komandos nusirito į žemesnes pozicijas, bet buvo atpažintos vos vienu ar dviem nuošimčiais prasčiau.

Testuojant profiliu, kuris nebuvo specialiai apmokytas, gauti rezultatai mažai skiriasi nuo prieš tai gautų – pavienių komandų atpažinimo tikslumas šokteli iki 100%, o prasčiausiai atpažintos komandos atpažinimo tikslumas yra 98,9%. Geriausiųjų komandų trejetukas – *Karolis, Oskaras, Patrikas* ir bent viena iš jų taip pat buvo atpažinta geriausiai bent viename iš prieš tai aptartų profilių.

Profilio, apmokyto vyrišku balsu ir testuojant su UPS transkripcijomis geriausiai atpažįstamos komandos yra *Hansas, Vacys, Donatas*. Tik komanda *Donatas* yra prieš tai patekusi į geriausiai atpažintų komandų trejetuką, o atpažinimo tikslumas taip pat siekia 99,9 - 100%.

Visų keturių profilių atpažinimo rezultatai yra labai panašūs, labai aukšti, bet komandų eiliškumo pasiskirstymas nėra visiškai tolygus – nėra fiksuojamas nuoseklumas, kinta geriausiai atpažįstamos komandos dešimtuke bei pirmosiose jo vietose. Todėl negalima teigti, kad profilio apmokymą galima tiesiogiai susieti su komandų atpažinimu, juolab kažkurių fonetinių savybių, kurios padėtų komandoms likti aukščiausiose pozicijose kiekviename profilyje.

2.5 Tyrimai su HTK paketu

Tolesniam darbui, kuris bus vykdomas naudojant HTK paketą, buvo atlikti testavimai būtent su juo. Pasirinkti kriterijai ir parametrai, pagal kuriuos buvo tikrinamas atpažinimo tikslumas. Balsų komandų modeliavimui naudotos ištisinės Markovo grandinės.

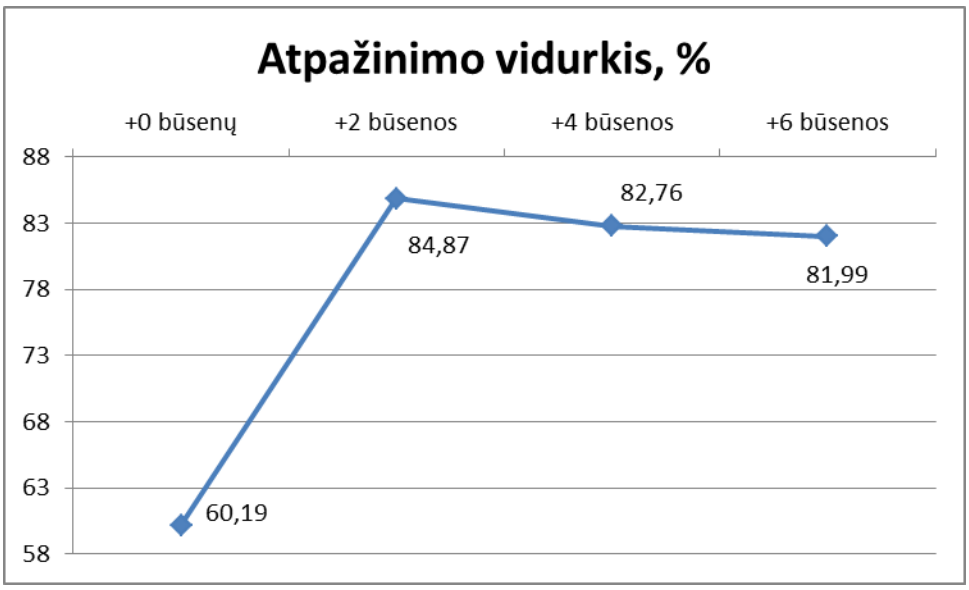
2.5.1 Lietuviškų vardų atpažinimo tyrimas

Šių testų metu buvo tikrinama atpažinimo priklausomybė nuo pasirinktų mišinių skaičiaus naudojant dvi būsenų skaičių grupes - +2 ir +4. Būsenų skaičius skaičiuojamas prie testuojamo žodžio raidžių skaičiaus, t.y. jau esamų būsenų (nuo 3 iki 10) pridedant pasirinktą būsenų skaičių (+2, +4 ir t.t.). mažiausią kiekį būsenų turėjo komanda Qju (3), o daugiausia komandos *Zacharijus* ir *Boleslovas*. Papildomi tyrimai atlikti siekiant išsiaiškinti, kaip susijęs atpažinimo tikslumas vien tik su būsenų skaičiumi. Naudojant HTK paketo programas, atlikti vardų garsyno testavimai. Testuojama buvo parenkant būsenų skaičių kaip pagrindinį parametrą. Kaip minėta prieš tai, būsenos žymimos kaip komandos raidžių skaičius +0 (nepridedant papildomų būsenų), +2, +4 ar +6. Atlikus testavimus, gauti rezultatai pareikti 5 lentelėje.

5 lentelė. Atpažinimo tikslumo priklausomybė nuo būsenų skaičiaus

Atpažinimo tikslumo priklausomybė nuo būsenų skaičius, %				
Vardas	**Būsenų skaičius atitinka raidžių sk.**			
	+0 būsenų	**+2 būsenos**	**+4 būsenos**	**+6 būsenos**
Austėja	100	100	100	100
Boleslovas	100	100	98,3	98,3
Cecilija	100	100	100	100
Donatas	66,7	100	48,3	40
Eimantas	98,3	100	95	100
Fausta	26,7	98,3	100	100
Gražvydas	76,7	50	41,7	98,3
Hansas	43,3	100	100	100
Izaokas	100	100	100	8,3
Jonas	11,7	83,3	83,3	100
Karolis	58,3	100	100	11,7
Laima	8,3	75	50	100
Martynas	85	100	100	100
Nojus	10	75	100	100
Oskaras	71,7	100	100	16,7
Patrikas	90	91,7	50	96,7
Kju	0	1,7	95	95
Ričardas	31,7	91,7	31,7	80
Sandra	6,7	100	100	100
Teodoras	96,7	100	98,3	100
Ulijona	100	100	98,3	70
Vacys	18,3	70	100	100
Wašington	68,3	70	70	100
Xsas	0	0	60	90
Ygrekas	96,7	100	31,7	26,7
Zacharijus	100	100	100	100
Vidurkis, %	**60,19**	**84,87**	**82,76**	**81,99**

Iš lentelėje pateiktų rezultatų matome, kad geriausi atpažinimo rezultatai gauti nustačius +2 papildomas būsenas prie esamo komandos raidžių skaičiaus – pasiektas 84,87% atpažinimo tikslumas. Antroji vieta tenka +4 būsenoms – 82,76%. Skirtumas nedidelis, vos 2,11%. Apibendrinant visus rezultatus, atpažinimo tikslumas nėra aukštas, todėl kiti tyrimai bus atliekami prie būsenų skaičiaus parametro prijungiant Gauso mišinių skaičių.Vizualus nuo būsenų priklausantis atpažinimo tikslumo kitimas pateikiamas 4 paveiksle.



4 pav. Atpažinimo tikslumo pagal būsenų skaičių diagrama

Geriausiai atpažįstamos komandos buvo *Zacharij*, *Cicilija* ir *Austėja* – jų atpažinimo tikslumas lygus 100%. Arčiausiai 100% ribos priartėjo komandos *Boleslovas*, *Teodoras* ir *Eimantas* – jų atpažinimo tikslumas svyravo nuo 98,33% iki 99,15%. Geriausiųjų komandų sąrašas, lyginant su prieš tai aptarto testo rezultatais, ne daug ir pasikeitė, naujomis buvo pakeistos tik kelios komandos. Kita vertus prasčiausiai atpažinta komanda liko ta pati – *Xsas*. Šios komandos atpažinimo tikslumas, skaičiuojant visų testų rezultatų vidurkį siekia vos 37,5%. Be to, verta paminėti, kad šių testų metu daug daugiau komandų atpažinimo tikslumas nesiekė 90% ribos – net 18. Iš to galima daryti prielaidą, kad mišinių skaičius, ypač taikant 10 mišinių parametrą, turi didelės įtakos tiek bendram komandų atpažinimo tikslumui, tiek pavienių komandų atpažinimui.

Kiekvienoje grupėje buvo tikrinamas atpažinimo tikslumas atliekant testus su 2, 3, 4, 6 ir 10 Gauso mišinių. Didinant mišinių skaičių, pakeičiamos atpažintuvo sąlygos, nuo kurių atpažinimo tikslumas gali kisti, bet nebūtinai į geresniąją pusę. Kuo daugiau komandų, variacijų, tuo didesnė tikimybė, kad tai įtakos atpažinimo tikslumą. Tyrimo metu buvo naudoti visi 21 diktoriai, iš kurių 18 apmokymui, o likusieji trys – testavimui.

Iš viso buvo atlikta dešimt testų su HTK paketu, tikrinant atpažinimo tikslumo priklausomybę nuo mišinių skaičiaus. Lentelėje (žr. 6 lentelę) pateikiami visų testų rezultatai ir kiekvienos komandos atpažinimo tikslumas procentine išraiška.

6 lentelė. Atpažinimo tikslumo priklausomybė nuo Gauso mišinių skaičiaus

Atpažinimo tikslumo priklausomybė nuo Gauso mišinių skaičiaus, %										
	Būsenų skaičius atitinka raidžių sk. + 2					Būsenų skaičius atitinka raidžių sk. + 4				
Vardas	2 mišiniai	3 mišiniai	4 mišiniai	6 mišiniai	10 mišinių	2 mišiniai	3 mišiniai	4 mišiniai	6 mišiniai	10 mišinių
Austėja	100	100	100	100	100	100	100	100	100	100
Boleslovas	100	100	100	100	100	100	100	100	98,3	100
Cecilija	100	100	100	100	100	100	100	100	100	100
Donatas	100	100	100	100	100	100	100	100	95	100
Eimantas	100	100	100	100	100	100	100	100	98,3	100
Fausta	100	100	100	96,7	100	100	100	100	100	100
Gražvydas	100	100	100	100	100	100	100	100	100	100
Hansas	100	100	100	100	100	100	100	100	100	100
Izaokas	98,3	98,3	100	100	100	100	100	100	96,7	100
Jonas	98,3	100	51,7	38,3	100	91,7	41,7	86,7	100	98,3
Karolis	96,7	75	100	100	100	100	100	100	100	100
Laima	96,7	91,7	90	81,7	100	100	100	100	100	100
Martynas	100	100	100	100	100	100	68,3	100	100	100
Nojus	98,3	100	73,3	96,7	100	100	100	100	100	100
Oskaras	85	66,7	95	100	100	100	56,7	100	61,7	100
Patrikas	100	100	100	100	100	100	78,3	100	86,7	100
Kju	83,3	91,7	88,3	88,3	86,7	85	95	86,7	95	95
Ričardas	100	100	100	100	100	100	75	100	78,3	100
Sandra	100	100	100	100	100	100	100	100	100	100
Teodoras	100	100	100	100	100	100	100	100	100	100
Ulijona	100	100	98,3	100	100	100	100	100	100	100
Vacys	85	91,7	98,3	1,7	91,7	66,7	100	98,3	100	100
Wašington	100	80	100	100	100	100	90	100	98,3	100
Xsas	70	8,3	58,3	38,3	100	91,7	98,3	95	85	100
Ygrekas	100	91,7	100	100	100	100	96,7	100	95	100
Zacharijus	100	100	100	100	100	100	100	100	98,3	100
Vidurkis, %	96,6	92,12	94,36	90,06	99,17	97,5	92,31	98,72	95,64	99,74

Pateiktos lentelės duomenys rodo, kad tiksliausiai atpažintuvas veikia parinkus 10 mišinių tiek vienu, tiek kitu atveju – ir su +2 ir su +4 būsenomis. Tokie rezultatai nebuvo prognozuoti, ir visiškai prieštarauja pirmame skyriuje aptartiems Šilingo [15] darbo rezultatams, kur didesnis mišinių skaičius

lėmė prastesnius rezultatus. Buvo tikėtasi, kad geriausi rezultatai bus pasiekti naudojant 2 arba 4 mišinius, kiek su +2, tiek su +4 būsenomis. Taigi aukščiausias pasiektas tikslumas yra 99,74%, jis pasiektas naudojant 10 mišinių su +4 būsenomis. Antrasis pagal tikslumą nustatytas parametrų junginys yra +2 būsenos ir 10 Gauso mišinių – pasiektas 99,17% tikslumas. Taigi matome, kad prie būsenų parametro pridėjus Gauso mišinių parametrą, atpažinimo tikslumas ženkliai šoktelėjo.

Geriausiai atpažįstamos komandos, kurių atpažinimo tikslumo vidurkis skaičiuojant bendrus testo rezultatus yra 100% yra *Teodoras, Sandra, Hansas, Gražvydas, Cecilija* ir *Austėja*. 95% ribą viršiujusios, pasiekusios 99,8% tikslumą komandos yra *Zacharijus, Ulijona, Eimantas* ir *Boleslovas*.

Prasčiausiai atpažintos komandos, kurių atpažinimas nesiekė 90% ribos yra *Jonas, Vacys, Oskaras, Kju.* Jų atpažinimo tikslumas svyruoja nuo 80,67% iki 89,5%. Prasčiausiai visų tyrimų metu atpažinta komanda yra *Xsas* – vos 74,49% tikslumas. Toks prastas rezultatas pasiektas dėl komandos trumpumo bei duslios, neapibrėžtos pradžios – fonetiškai šis garsas sunkiai fiksuojamas įrašymo įrangos.

2.5.2 Skaičių pavadinimų atpažinimo tyrimas

Parengus atpažintuvą naudojantis HTK paketo programomis, buvo atliekami skaičių pavadinimų atpažinimo tyrimai.

Atpažinimo tikslumo priklausomybė nuo būsenų skaičiaus. Pirmiausia tirta atpažinimo tikslumo priklausomybė nuo būsenų skaičiaus. Testai atlikti naudojant raidžių skaičių pridedant prie 1, 2, 4, 10 ir 16 būsenų. Testų rezultatai pateikiami 7 lentelėje.

7 lentelė. Skaičių pavadinimų atpažinimo tikslumo priklausomybė nuo būsenų skaičiaus

Komanda	Būsenų skaičius atitinka raidžių skaičių													
	+1							+2						
	FIEVVIS	FJUSKIN	FUGNBUC	FUGNNOV	FVISSTA	MLINJUR	Vidurkis, %	FIEVVIS	FJUSKIN	FUGNBUC	FUGNNOV	FVISSTA	MLINJUR	Vidurkis, %
NULIS	14	20	11	2	1	19	55.8	9	20	11	19	4	20	69.2
VIENAS	20	20	20	20	20	20	100.0	16	13	20	17	20	13	82.5
DU	0	0	0	0	0	0	0.0	0	0	7	1	1	0	7.5
TRYS	2	6	1	15	2	2	23.3	4	12	10	2	11	12	42.5
KETURI	15	18	18	19	20	19	90.8	20	20	18	20	20	19	97.5
PENKI	19	10	18	5	19	10	67.5	20	17	18	4	19	9	72.5
ŠEŠI	2	20	18	19	20	18	80.8	3	20	20	20	20	19	85.0
SEPTYNI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0

AŠTUONI	17	20	20	20	20	19	96.7	16	20	20	20	19	20	95.8
DEVYNI	18	20	16	20	19	18	92.5	20	17	14	20	19	18	90.0
Bendras vidurkis, %							70.8							74.3

Komanda	Būsenų skaičius atitinka raidžių skaičių													
	+4							+10						
	FIEVVIS	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	**Vidurkis, %**	**FIEVVIS**	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	**Vidurkis, %**
NULIS	20	20	20	20	20	20	100.0	20	20	18	20	19	20	97.5
VIENAS	19	20	20	20	20	12	92.5	19	20	20	20	20	20	99.2
DU	6	6	18	15	19	13	64.2	19	20	20	20	20	20	99.2
TRYS	20	18	18	20	17	15	90.0	19	20	20	20	20	20	99.2
KETURI	20	20	17	20	20	20	97.5	19	19	19	19	19	19	95.0
PENKI	20	19	19	20	20	20	98.3	20	20	20	17	20	20	97.5
ŠEŠI	15	20	20	20	20	20	95.8	16	20	20	20	20	20	96.7
SEPTYNI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
AŠTUONI	17	20	20	20	20	20	97.5	15	20	20	20	19	20	95.0
DEVYNI	20	19	16	20	18	20	94.2	20	19	15	20	20	16	91.7
Bendras vidurkis, %							93.0							97.1

Būsenų skaičius atitinka raidžių skaičių

Komanda	+16						
	FIEVVIS	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	**Vidurkis, %**
NULIS	19	20	15	19	12	20	87.5
VIENAS	20	20	20	20	19	19	98.3
DU	19	20	19	20	20	20	98.3
TRYS	20	18	20	17	20	19	95.0
KETURI	20	20	19	18	19	19	95.8
PENKI	20	19	20	18	18	20	95.8
ŠEŠI	17	19	20	20	19	20	95.8
SEPTYNI	20	20	19	20	20	20	99.2
AŠTUONI	19	19	18	20	20	20	96.7
DEVYNI	19	20	15	20	20	19	94.2
Bendras vidurkis, %							95.7

Lentelėje pateikti visų testų duomenys. Iš jų galima matyti, kad rezultatai pasiskirstę labai nevienodai. Pradėsiu nuo rezultatų, gautų atliekant testavimus su papildoma viena būsena. Vidutinis kiekvienos komandos atpažinimo tikslumas tesiekia 70,8%. Geriausiai atpažintos komandos, 100% tikslumo yra *vienas* ir *septyni*. Be jų, daugiau nei 90% tikslumu atpažintos komandos yra *keturi, aštuoni*

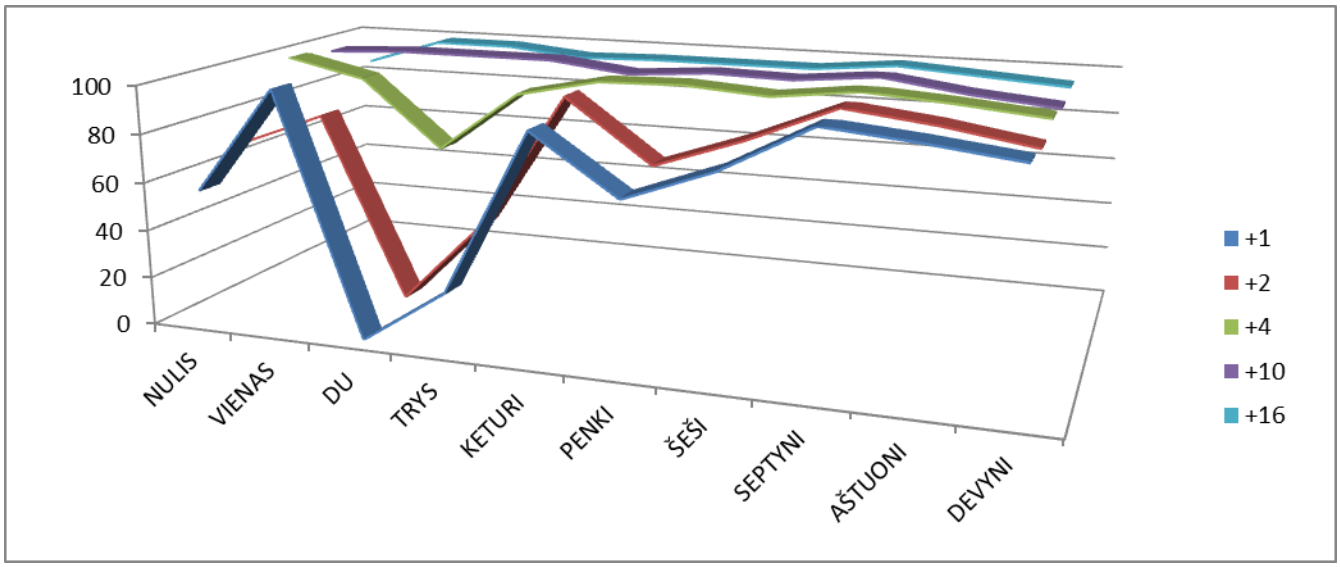
ir *devyni*. Nei karto neatpažinta komanda yra *du*. Tokį reiškinį galima aiškinti komandos trumpumu – ankstesniuose tyrimuose buvo pastebėta, kad trumpesnės komandos, ypač tos, kurių pradžioje yra duslusis priebalsis, atpažįstamos prasčiau. Kiek geriau, bet tik 23,3% tikslumu atpažinta komanda *trys*.

Prie raidžių skaičiaus pridėjus dvi papildomas būsenas matome nedidelį pagerėjimą atpažinimo tikslume – 74,3%. Tai rodo, kad net šiektiek didesnis būsenų skaičius sąlygoja geresnius atpažinimo rezultatus. Geriausiai atpažįstama komanda taip pat *septyni*, kaip ir testo su viena papildoma būsena rezultatuose. Komandos *aštuoni, devyni, keturi* taip pat yra atpažįstamos geriausiai – daugiau nei 90% tikslumu. Prasčiausia vis dar liko komanda *du* (7,5%), o komanda *trys* atpažįstama ženkliai geriau, beveik du kartus padidėjo atpažinimo tikslumas – 42,5%.

Dvigubai padidinus būsenų skaičių, iki 4, atpažinimo tikslumas ženkliai padidėjo, jis siekia net 93%. Toks rezultatas tikrai tenkinantis. Neliko komandų, kurių atpažinimas nesiektų 50%, o iki šiol prasčiausia komanda buvusi *du* šio testo metu pasiekė 64,2% tikslumą, na o visos kitos – atpažįstamos daugiau nei 90% tikslumu. Geriausia komanda liko *septyni* (100%) kaip *nulis* (100%).

Geriausi iki šiol rezultatai pasiekti naudojant 10 papildomų būsenų – 97,1% tikslumas. Nuo pradinio rezultato tai svarus 27,7% pagerėjimas. Visos komandos, net ir *du* atpažįstamos aukštesniu nei 90% tikslumu. Įdomu tai, kad komanda *septyni* vis dar atpažįstama geriausiai - 100% tikslumu ir toks rezultatas pastebimas visų testų metu. Šio testo metu, prasčiausia komanda tapo *devyni*, nors šios pozicojos nebuvo užėmusi visų 4 testų metu. O komandos *du* atpažinimas išaugo iki 99,2%. Nors pasiekti geriausi rezultatai, atliktas dar vienas testas su papildomų 16 būsenų.

Paskutiniojo testo metu rezultatai šiek tiek suprastėjo testo su papildomų 10 būsenų atžvilgiu – atpažinimo tikslumas sumažėjo iki 95,7%. Nei viena komanda nepasiekė 100% atpažinimo tikslumo – komandos *septyni* atpažinimas nukrito iki 99,2%, kas yra geriausias pasiektas rezultatas šio testo metu. O prasčiausiai atpažinta komanda tapo *nulis* – 87,5%. Žemiau pateiktoje diagramoje (žr. 4 paveikslą) galima matyti bendrą komandų atpažinimo kitimą priklausomai nuo būsenų skaičiaus. Kaip matome, komandos *du* atpažinimo tikslumo pagerėjimas yra ryškiausias – kaip jau minėta – pagerėjo nuo 0% iki 99,2% - tai beveik 100% pagerėjimas. Kita komanda, kurios atpažinimo tikslumas ženkliai pagerėjo yra *penki* – nuo 67,5% iki 98,3%.



5 pav. Skaičių pavadinimų atpažinimo tikslumo priklausomybė nuo būsenų skaičiaus

Apibendrinant rezultatus galima pasakyti, kad palaipsniui didinant būsenų skaičių galima pasiekti geresnius rezultatus, bet egzistuoja tam tikra riba, kurią pasiekus jie nesikeičia arba pradeda blogėti dėl potencialiai per didelio apkrovimo.

Skaičių pavadinimų atpažinimo tikslumo priklausomybė nuo būsenų ir Gauso mišinių skaičiaus. Atlikus pirminius testus esant kintamam mišinių skaičiui, pabandysime išsiaiškinti, ar Gauso mišinių skaičius turi įtakos atpažinimo rezultatams, jei jie kombinuojami kartu su papildomomis būsenomis. Atliekti testai keičiant Gauso mišinių skaičių – bus taikomos 2, 3, 6 ir 10 mišinių kombinacijos su 2 papildomom būsenom. Testai atliekami naudojant tų pačių šešių diktorių įrašus. Testų rezultatai pateikiami 8 lentelėje.

8 lentelė. Skaičių pavadinimų atpažinimo tikslumo priklausomybė nuo būsenų ir Gauso mišinių skaičiaus

Būsenų skaičius atitinka raidžių skaičių +2														
Gauso mišinių skaičius	2 mišiniai							3 mišiniai						
Komanda	**FIEVVIS**	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	Vidurkis, %	**FIEVVIS**	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	Vidurkis, %
NULIS	20	19	19	19	7	20	86.7	20	20	20	20	20	20	100.0
VIENAS	17	17	20	19	20	18	92.5	20	20	20	20	20	20	100.0
DU	1	0	3	4	2	4	11.7	8	11	18	17	18	20	76.7
TRYS	19	19	19	19	20	19	95.8	19	19	19	19	19	20	95.8
KETURI	19	19	20	20	20	20	98.3	20	20	20	20	20	20	100.0

PENKI	20	20	20	18	15	17	91.7	20	20	20	20	20	20	100.0
ŠEŠI	19	20	19	20	20	20	98.3	20	20	20	20	19	20	99.2
SEPTYNI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
AŠTUONI	12	18	20	20	20	19	90.8	20	20	20	20	20	20	100.0
DEVYNI	15	19	16	20	19	20	90.8	20	20	20	20	19	20	99.2
	Bendras vidurkis, %						**85.7**	**Bendras vidurkis, %**						**97.1**
Būsenų skaičius atitinka raidžių skaičių +4														
	6 mišiniai							10 mišinių						
Komanda	**FIEVVIS**	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	Vidurkis, %	**FIEVVIS**	**FJUSKIN**	**FUGNBUC**	**FUGNNOV**	**FVISSTA**	**MLINJUR**	Vidurkis, %
NULIS	20	20	20	20	20	20	100.0	20	20	20	19	20	20	99.2
VIENAS	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
DU	20	20	20	18	20	19	97.5	16	19	20	19	20	19	94.2
TRYS	19	19	19	19	19	19	95.0	19	19	19	19	19	19	95.0
KETURI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
PENKI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
ŠEŠI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
SEPTYNI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
AŠTUONI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
DEVYNI	20	20	20	20	20	20	100.0	20	20	20	20	20	20	100.0
	Bendras vidurkis, %						**99.3**	**Bendras vidurkis, %**						**98.8**

Duotojoje lentelėja matom gana tolygų duomenų pasiskirstymą. Bendras atpažinumo vidurkis svyruoja nuo 85,7% iki 99,3%. Matomas ženklus pagerėjimas pirmųjų testavimų atžvilgiu, kai nebuvo naudoti Gauso mišiniai. Pavyzdžiui lyginant rezultatus, kurie gauti naudojant tik 6 mišinius su gautais testuojant komandas su dviem papildomom būsenom matome jau gana ženklų bendro vidurkio pagerėjimą – jis padidėjo 11,4%. Visų pavienių komandų atpažinimas svyruoja apie 90%, o tuo tarpu komandos *du* pagerėjimas yra 11,7%. Stebėtina, bet geriausiai atpažįstama komanda išliko *septyni* (100%), bet jų padaugėjo kitų testų metu.

Padidinus Gauso mišinių skaičių iki 3 matome ženklų pagerėjimą pirmojo testo atžvilgiu – atpažinimo tikslumas išaugo 11,4%. Net 6 komandų iš 10 atpažinimo tikslumas siekia 100%, bet komando *du* atpažinimas vis dar išlieka prasčiausias, nors ir ženkliai pagerėjęs – nuo 11,7% iki 76,7%, tai net 65% pagerėjimas. Šio testo rezultatai prilygsta aukščiausiam rezultatui, kuris buvo gautas testuojant skaičių pavadinimus nenaudojant Gauso mišinių.

Kitas testas atliktas padidinus gauso mišinių skaičių iki 6. Atpažinimas išaugo 2,2% - iki 99,3%, bet po tokios ribos kiekvinas procentas ar net dešimtoji jo dalis yra labia svarbūs. Komandos *du*

atpažinimas pasiekė aukščiausią tašką bandymų su Gauso mišiniais metu – 97,5%, lyginant su aukščiausiu rezultatu testuojant be Gauso mišinių, jis šiek tiek sumažėjęs nuo 99,2%, bet nepaisant to, tikslumas yra labia aukštas. Septynios komandos iš 10 pasiekė 100% atpažinimo tikslumą.

Dar kartą padidinus mišinių skaičių rezultatai suprastėjo – pasiektas 98,8% tikslumas, kurį sumažino komandos *nulis* ir *du* atpažinimo tikslumo pasikeitimas į neigiamą pusę. Tai gali reikšti, kad gali tolimesnis mišinių skaičiaus didinimas gali pateikti tik blogesnius rezultatus, todėl ši riba nebuvo peržengiama. Kitame skyrelyje bus lyginami geriausi gauti rezultatai testuojant su kitakalbiu atpažintuvu ir su HTK paketo atpažintuvo gautais rezultatais.

2.6 Dviejų atpažintuvų atpažinimo rezultatų palyginimas

Kiekvienas iš atpažintuvų turi vienokių ar kitokių savybių, kurios turi įtakos ir naudos tikslesniam atpažinimui. Kaip buvo minėta skyriuje 1.2, ispanų kalbos atpažintuvas ypatingas tuo, kad jo algoritmai puikiai fiksuoja signalo pradžią ir pabaigą, o lietuviškas, paremtas Paslėptomis Markovo grandinėmis, naudingas sprendimo priėmimo bloku.

Apibendrinant abiejų tyrimų rezultatus (žr. 9 lentelę) matome, kad lietuviškas atpažintuvas, kurtas naudojant HTK paketo programas yra daug pranašesnis už kitakalbį.

9 lentelė. Geriausių skaičių pavadinimų ir vardų atpažinimo rezultatų palyginimas

Lietuviški vardai		**Skaičių pavadinimai**	
Kitakalbis atpažintuvas	Lietuviškas atpažintuvas	Kitakalbis atpažintuvas	Lietuviškas atpažintuvas
96,91%	99,17%	89,2%	98,8%
97,09%	99,74%	93,7%	99,3%

Žinoma, lyginami rezultatai nėra visiškai lygiaverčiai, tačiau tiek testuojant su kitakalbiu, tiek su lietuvišku atpažintuvu buvo bandomos įvairios priemonės, kurios pagerintų atpažinimo tikslumą, tad paimti tik geriausi gauti rezultatai. Lentelėje matome, kad lietuviškų vardų atpažinimo tyrime geriausias rezultatas su lietuvišku atpažintuvu lenkią ispaniškojo rezultatą 2,65%. Kaip jau minėjau, peržengus 97% ribą, kiekviena procento dalis yra labai svarbi ir reiškia progresą.

Skaičių pavadinimų atpažinime taip pat pirmauja lietuviškas atpažintuvas, bet ženkliai didesniu skirtumu – rezultatas geresnis net 6,4% ir priartina atpažintuvą prie 100% tikslaus atpažinimo. Potencialiai sujungus atpažintuvus būtų galima pasiekti šiek tiek geresnius rezultatus. Pastebima, kad skirtingų komandų, skirtingų testų metu atpažinimas skiriasi bei svyruoja į vieną ar kitą pusę – nėra pastovaus gerėjimo nuosekliai. Pakeitus vienus, ar kitus parametrus, pavyzdžiui, mišinių skaičių ar būsenų skaičių, ar kitakalbio atpažintuvo testavimo atveju pakeitus diktorių ar tiesiog profilį, matome

vienų komandų atpažinimo pagerėjimą, o kitų suprastėjimą. Toks kitimas nėra blogai kol pasiekiamas aukštas atpažinimo tikslumo balas, nes svyravimai, kokie jie bebūtų, jei bandymai tikslingi, pasiekia tą aukščiausią, tinkamą ribą, bet siekiant tą svyravimą sustabdyti, reikia nustatyti, kas jį sukelia. Galbūt dviejų atpažintuvų sujungimas į vieną, išnaudojant vieno ir kito savybes pagerintų tikslumą ir subalansuotų atpažinimo sprendimų priėmimą prasčiau atpažįstamų komandų atžvilgiu.

2.7 Demonstracinės programos

Atlikus analizę, nustačius, kuris atpažintuvas veikia geriau nuspręsta pabandyti realizuoti ir vizauliai pateikti, kaip veikia pats ligos kodo atpažinimas, jo įvedimas balsu į programos langą bei jo paieška internete. Be to, norint perteikti platesnį pritaikomumą, sudaryta programa, kuri atpįstą komandą, bet pateikia daugiau informacijos apie ją per sąsają su duomenų baze.

2.7.1 Ligų kodų atpažinimo su kitakalbiu atpažintuvu ir sąsajos su internetu programa

Demonstracinė programa sukurta naudojant *Microsoft Viasual Studio 2013* programinį paketą, ji parašyta *C#* kalba. Detaliau apžvelgsiu programinį kodą ir pateiksiu vizualius programos langų pavyzdžius. Visas programos kodas pateiktas 3.8 priede.

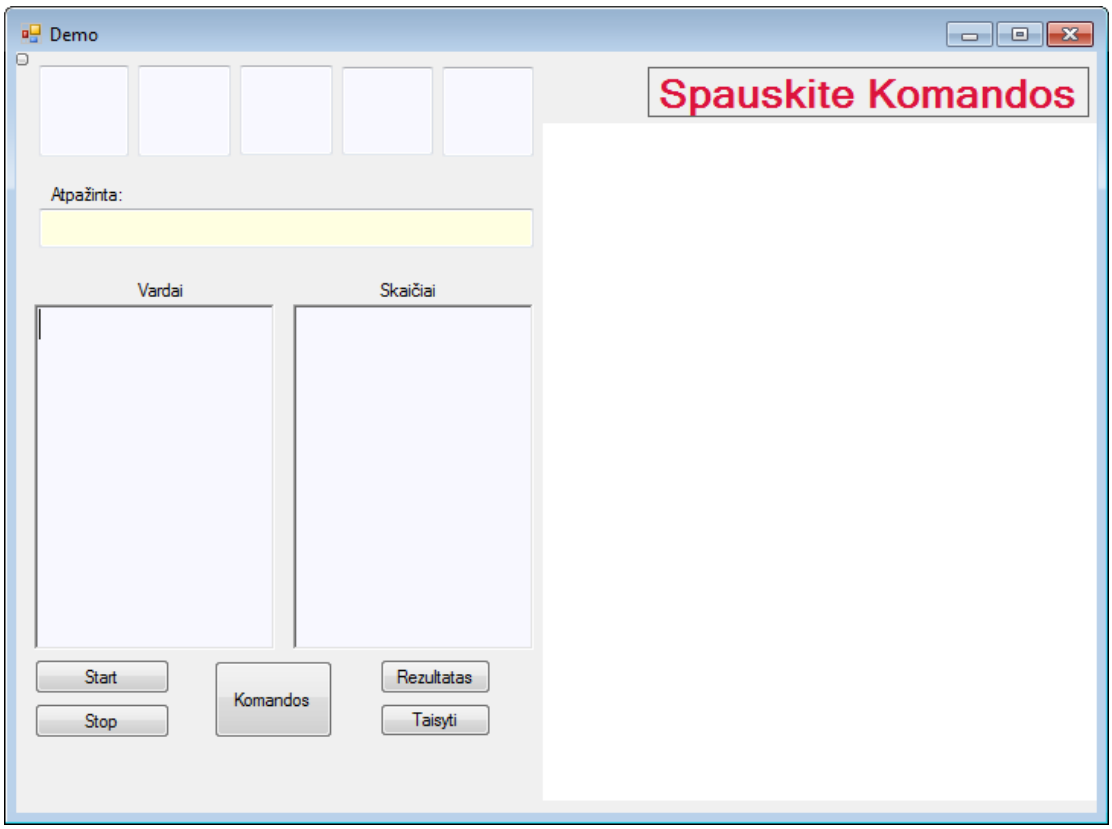
Paleidus programą, matomi keli langai ir mygtukai (žr. 6 paveikslą). Į viršuje esančius 5 langelus yra išvedamos atpažintos komandos – raidė, skaičiai ir taško simbolis. Eilutėje po jais išvedamas ligos pavadinimas, kuris susietas pagal kodą. Dar žemiau esantys du langai talpina vardų ir skaičių sąrašus, kaip pagalbą diktuojant. Dešinėje pusėje esanti tuščia erdvė užsipildo interneto puslapio vaizdu su informacija apie ligą, kuri yra atpažinta įdiktavus jos kodą.

Programa susideda iš kelių esminių dalių, tad trumpai paaiškinsiu jos kodą.

Pirmiausia yra paleidžiamas atpažintuvas ir aprašomi keli kintamieji:

```
private SpeechRecognitionEngine recognizer = new SpeechRecognitionEngine();
        string kodas = "",komanda="", kodas_old="";

                int num = 0;
```

Čia kodas yra pilna raidžių ir skaičių kombinacija; komanda yra kiekvienas įdiktavimas; kodas_old – saugoma kodo dalis, tai reiškia, kad diktuojant padarius klaidą, galima ištrinti vieną komandą einančia iš eilės atgaline tvarka ir įvesti naują.



6 pav. Ligų kodų atpažinimo su kitakalbiu atpažintuvu ir sąsajos su internetu programos langas

Kiekvienas iš programos lange esančių mygtukų atlieka savo funkciją. Pavyzdžiui mygtukas *Komandos* suaktyvina kelias funkcijas – gramatikos failo užkrovimą (gramatika, kurią naudoja programa yra jungtinė skaičių ir vardų, nes atpažintuvas vienu metu dirba tik su viena gramatika), suaktyvina *.rtf* failų įkėlimą į duotus komandų langus ir jį paspaudus keičiamas tekstas informaciniame lange virš naršyklės ploto.

```
private void Vardai_Click(object sender, EventArgs e)
        {
            Grammar gramatika = new Grammar("vardai.grxml", "Rule");
            recognizer.LoadGrammar(gramatika);
            label2.Text = "Spauskite Start";
            richTextBox1.Clear();
            richTextBox1.LoadFile("vardai.rtf");
            richTextBox2.Clear();
            richTextBox2.LoadFile("skaiciai.rtf");
                  }
```

Mygtuko *Start* paspaudimas įjungia atpažintuvą ir nurodo jį dirbti asinchroniniu režimu. Mygtukas *Stop* išvalo visus tekstinius laukus ir sustabdo gramatikos naudojimą. Mygtuko *Taisyti* funkcija yra ištrinti paskutinę įdiktuotą komandą ir leisti įdiktuoti ją iš naujo. Paspaudus mygtuką *Rezultatas*, vykdomi loginiai veiksmai – tikrinama, ar įvesta skaičių ir raidžių kombinacija atitinka reikiamą, jei taip – išvedama komanda į rezultatų langą ir paleidžiama naršyklė su atitinkamos ligos puslapiu:

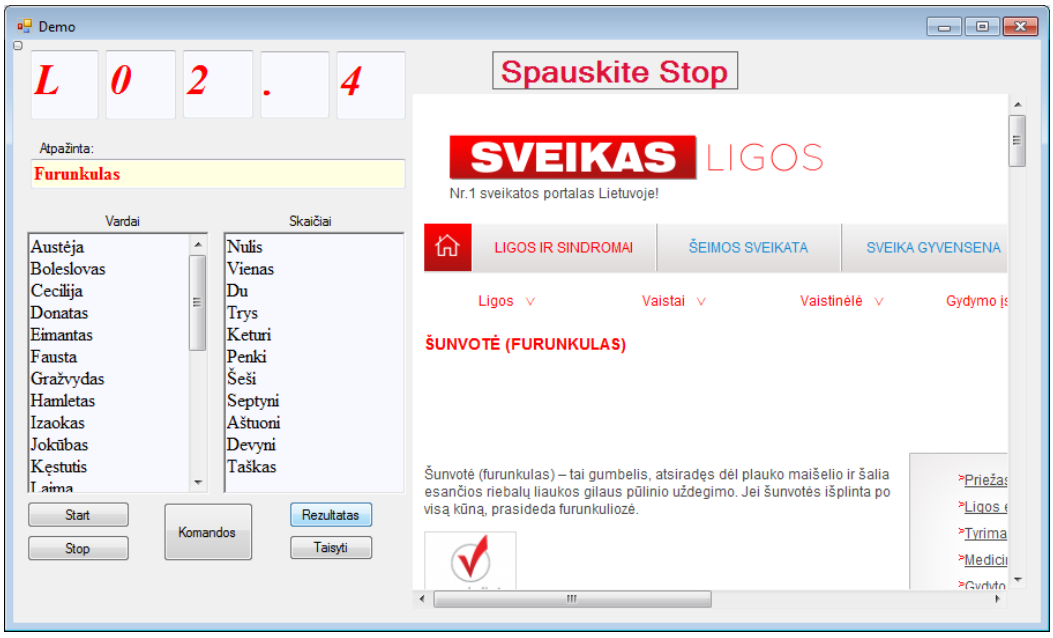
```
private void Liga_Click(object sender, EventArgs e)
        {
            label2.Text = "Spauskite Stop";
            if (kodas == "H65.9")
            {
                textBox6.Text = "Ausies uždegimas";
                string adresas =
"http://ligos.sveikas.lt/lt/ligos/ausunosiesgerkles_ligos/isorines_ausies_uzdegimas_otitas";
                webBrowser1.Navigate(textBox6.Text);
                webBrowser1.Navigate(adresas);

                    };
```

Pagrindinė programos funkcija – atpažinimas – aprašyta kode žemiau:

```
void recognizer_SpeechRecognized(object sender, SpeechRecognizedEventArgs e)
        {
            num++;
            string word = e.Result.Text.ToString();
        //    textBox1.Text = word;
           string komanda = e.Result.Semantics.Value.ToString();
            kodas_old = kodas;  kodas = kodas + komanda;
            if (num == 1) { textBox1.Text = komanda; }
            if (num == 2) textBox2.Text = komanda;
            if (num == 3) textBox3.Text = komanda;
            if (num == 4) textBox4.Text = komanda;
            if (num == 5) { textBox5.Text = komanda; label2.Text = "Spauskite Rezultatas"; }
          }
```

Ištarimas, apdorotas atpažintuvo verčiamas į simbolių eilutę, įsijungia skaitliukas – skaičiuoja kiek komandų ištarta ir prireikus vieną išvalyti, grįžta prie pradinės komandos. Tikrinamas eiliškumas ir kiekviena komanda išvedama į vieną iš 5 langelių.

Atlikus nurodytus veiksmus, programos langas užsipildo informacija (žr. 9 paveikslą).

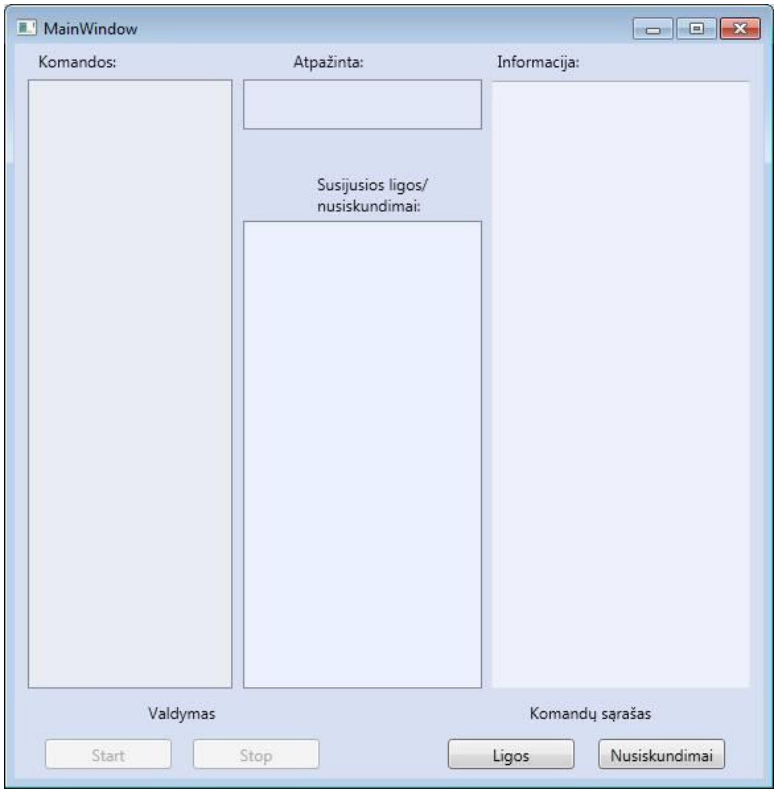


7 pav. Ligos kodo atpažinimo programos langas po atpažinimo ir su išvestimi į naršyklę

Programos valdymas ir vizualizacija gana paprasti ir šabloniški, bet tai tik bandomoji versija skirta pademonstruoti, kaip tinkamai adaptavus atpažinimo sistemą galima pasiekti patogumą ir gana greitą duomenų įvedimą. Diktavimų metu, testuojant programą pastebėta, kad ilgos dirbtinės pauzės nėra reikalingos įdiktuojant kodą, buvo naudotas integruotas nešiojamojo kompiuterio mikrofonas, kurio pozicija nuo diktoriaus nutolusi per 40-50cm. Toks atstumas neturėjo neigiamos įtakos atpa-inimui, jei komandos buvo tariamos aiškiai.

2.7.2 Demonstracinė sąsajos su duomenu baze programa

Ši programa sukurta naudojantis *Microsoft Visual Studio 2013* programų paketu, parašyta *C#* programavimo kalba ir naudojantis KTU ITPI resursais. Programos veikimo principas yra toks – jei programos lange (8 pav.) pasirenkame *ligų* sąrašą, ištarta komanda, naudojantis duomenų baze ir jos įrašais, yra susiejama su aprašymu, kuris išvedamas į vieną iš programos langų. Taip pat yra pateikiamas susijusių nusiskundimų sąrašas. Jei pasirenkame *nusiskundimų* sąrašą, tuomet pagrindiniame lange yra išvedamos dažniausiai susijusios ligos, o šalutiniame kitos, potencialios ligos.



8 pav. Atpažintuvo programos susietos su duomenų baze pavyzdys

Programa sudaryta iš 4 langų – viename iš jų išvedamas komandų sąrašas, kurias galima įdiktuoti ir kurios yra atpažįstamos. Mažiausiajame lange išvedama atpažinta komanda; lange, esančiame po juo išvedami duomenys susiję su liga, tai gali būti nusiskundimai, o jei diktuojame

nusiskundimus – susijusios ligos. Dešiniajame lange išvedama pagrindinė informacija apie ligą, o jei įdiktuojame nusiskundimus, kelias pačias dažniausias ligas. Navigaciniai mygtukai paleidžia arba sustabdo atpažintuvą, o dešinėje pusėje esantys mygtukai leidžia pasirinkti komandų sąrašą, kurį norėsime įdiktuoti. Toliau trumpai aptarsiu esmines programos kodo dalis, o visą kodą galima rasti 3.9 priede.

Pirmiausia yra sukuriamas atpažintuvas:

public SpeechRecognitionEngine recognizer = new SpeechRecognitionEngine();

Dėl paprastesnės navigacijos, jis pavadintas *rec*. Toliau yra pasirenkamas įrašymo įrenginys – pagal nutylėjimą tai bet kuris pagrindinis mikrofonas:

rec.SetInputToDefaultAudioDevice();

Tame pačiame žingsnyje nustatoma pradžios ir pabaigos mygtukų veikimo padėtis – jie yra neveiklūs.

```
button_Stop.IsEnabled = false;
button_Start.IsEnabled = false;
InitializeComponent();
```

Kitas programos veiksmas yra nustatyti kas įdiktuota – liga ar nusiskundimas, tai nustatoma pagal mygtuko paspaudiną ir *x* reikšmę. Nustačius kad tai liga, kreipiamasi į duomenų bazę ieškant atitikmens, susietą su nusiskundimais ir informacija apie ligą.

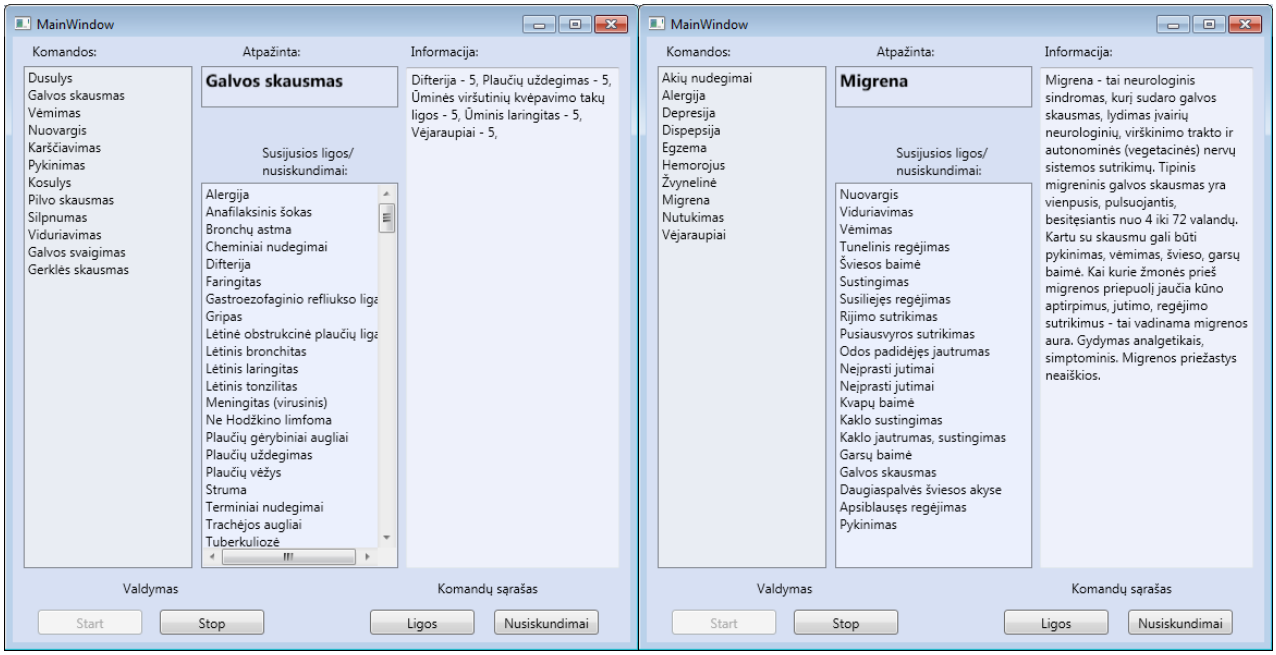
```
var data = new medicinine_dbDataSetTableAdapters.Nusiskundimai_LigosTableAdapter();
var data2 = new medicinine_dbDataSetTableAdapters.LigosTableAdapter();
var nusiskund = from p in data.GetData().
```

Jei pateikta komanda yra nusiskundimas, kas taip pat nustatoma pagal *x* kintamojo reikšmę, kreipiamasi į atitinkamą duomenų bazės sritį, kurioje saugoma informacija apie nusiskundimą, ligas susijusias su jais. Su *Start, Stop* mygtukų paspaudimais keičiama jų būsena, po pakeitimų visada išvalomi komandiniai langai. Be to kas kartą prieš užkraunant kokį nors tekstą, tie langai yra išvalomi.

Be to, užuot naudojus *.rtf* failus, dėl mažos programos apimties, išvedamojo teksto eilutės įrašytos tiesiai programoje – tai komandų sąrašas pateikiamas diktoriui.

```
listBox_gramatika.Items.Clear();
listBox_gramatika.Items.Add("Akių nudegimai");
listBox_gramatika.Items.Add("Alergija");
listBox_gramatika.Items.Add("Depresija");
listBox_gramatika.Items.Add("Dispepsija");
listBox_gramatika.Items.Add("Egzema");
listBox_gramatika.Items.Add("Hemorojus");
```

Žemiau pateikiamas pavyzdys, kokia informacija yra išvedama į programos langus tiek diktuojant ligų pavadinimus, tiek nusiskundimus. (žr. 9 pav.)



9 pav. Išvedamojo teksto pavyzdžiai programoje susietoje su duomenų baze

Kaip matome paveiksluose, informacija, kuri yra išvedami yra tvarkinga ir graži, o atpažintuvas veikia sklandžiai, be laiko delsimų ir kokybiškai.

Paruošti demonstraciniai programų vaizdo įrašai bus pateikti kaip priedas optinėje laikmenoje kartu su magistro projekto failu. Jų peržiūrai pakaks standartinio *MS Windows* aplinkoje veikiančio vaizdo įrašų grotuvo.

IŠVADOS IR REZULTATAI

1. Atlikus literatūros šaltinių analizę hibridinei atpažinimo technologijai testuoti pasirinkti kitakalbis ispanų kalbos atpažintuvas ir lietuviškas atpažintuvas paremtas Paslėptais Markovo modeliais. Ispaniškas atpažintuvas pasirinktas dėl fonetinės kalbos struktūros artimos lietuvių kalbai bei puikaus signalo aptikimo modulio, o lietuviškasis dėl spendimo priėmimo savybių bei nesudėtingo paruošimo bei apmokymo.
2. Atlikti skaičių pavadinimų testavimai su ispanų kalbos atpažintuvu tiek sinchroninėje, tiek asinchroninėje veiksenose ir atrinkti geriausi atpažintuvų rezultatai. Geriausias skaičių pavadinimų atpažinimo rezultatas yra 93,7%, jis pasiektas testus atliekant asinchroninėje veiksenoje ir yra 4,58% aukštesnis už rezultatus gautus testuojant sinchroninėje veiksenoje. Geriausiai asinchroninėje veiksenoje buvo atpažįstama komanda *septyni* (100%), o nuo jos nedaug atsiliko komandos *šeši* ir *aštuoni* – pasiektas beveik 100% atpažinimo tikslumas.
3. Lietuviškų vardų ir komandų pavadinimų atpažinimo tikslumas lygintas pagal diktoriaus profilį ir geriausi rezultatai gauti testuojant su vyrišku balsu apmokytu profiliu su žodinėmis transkripcijomis – 96,44%. Prasčiausią rezultatą pasiekė profilis, apmoytas vyrišku balsu ir su UPS transkripcijomis – 94,12%. Geriausiai atpažįstami lietuviški vardai buvo *Karolis* ir *Oskaras* – 98,73% ir 99,05% tikslumas atitinkamai.
4. Tiriant komandų atpažinimo tikslumą su lietuvišku atpažintuvu pastebėtas ženklus rezultatų pagerėjimas tiek skaičių pavadinimų, tiek vardų atpažinimo atvejais. Pirmieji atlikti bandymai tiriant priklausomybę nuo būsenų skaičiaus parodė, kad geriausi rezultatai gaunami naudojant raidžių skaičiu su 10 papildomų būsenų. Atpažinimo tikslumas testuojant vardus šoktelėjo iki 84,87% lyginant su prasčiausiu rezultatu gautu nenaudojant papildomų būsenų – 60,19%. Geriausi skaičių pavadinimų atpažinimo rezultatai taip pat gauti testuojant su 10 papildomų būsenų, lyginant su prasčiausiais 70,8% be papildomų būsenų.
5. Atlikus papildomus testus su Gauso mišiniais ir 2 papildomom būsenom, geriausi rezultatai testuojant skaičių pavadinimų atpažinimą gauti su 2 papildomom būsenom ir 6 Gauso mišiniais – 99,3%. Prasčiausi rezultatai – su 2 papildomom būsenom ir 2 mišiniais – 85,7%. Lietuviški vardai geriausiai atpažįstami pritaikant 4 papildomas būsenas ir 10 Gauso mišinių – atpažinimo tikslumas siekia 99,74%, o prasčiausias rezultatas gautas su 2 papildomom būsenom ir 6 Gauso mišiniais.

LITERATŪROS SĄRAŠAS

1. Balvočius B., Telksnys L. (2003). Lietuvių kalbos kompiuteriniai tyrimai (IX sekcija): Garsynų duomenų modeliai ir programinės įrangos architektūros /Vytauto Didžiojo universitetas, Matematikos informatikos institutas.
2. Barras C., Geoffrois E., Wuand Z., Liberman M. (1998). Transcriber: a Free Tool for Segmenting, Labeling and Transcribing Speech.
3. Bartišiūtė G., Ratkevičius K.. (2012) Speech Server based Lithuanian Voice Commands Recognition . Electronics and Electrical Engineering. Kaunas : Technologija. ISSN 1392-1215. 2012, Vol. 18, NO. 10, p.53-56.
4. Bartoševič, D., Kamarauskas, J. (2011) Nepriklausomas nuo kalbėtojo balso komandų atpažinimo tyrimas. 2011 metų teminės konferencijos straipsnių rinkinys. VGTU, Vilnius. Interaktyvus [žiūrėta 2015-03-13]. Prieiga internete: [http://dspace.vgtu.lt/ bitstream/1/737/1/ DBartosevic.pdf]
5. Braubartas, E. (2008). Lietuvių kalbos atskirai tariamų žodžių, skirtstant juos į kategorijas, atpažinimo tyrimas. Jaunųjų mokslininkų darbai. Nr. 5, 6 p. Šiauliai.
6. Dusan, S.; Rabiner, L. R. On Integrating Insights from Human Speech Perception into Automatic Speech Recognition. In proceedings of Interspeech 2005, September, 4 – 8, Lisbon, Portugal, 2005, p. 1233 – 1236.
7. Filipovic, M.; Lipeika, A. Development of HMM/Neural Network-based Medium-Vocabulary Isolated-Word Lithuanian Speech Recognition System. Informatica, 2004, vol. 15, no. 4, p. 465-474.
8. Fosberg M. (2003). Why is Speech Recognition Difficult? Department of Computing Science, Chalmers University of Technology.
9. Kasparaitis P., Dumbliauskas T., Rudžionis A. (2003) Lietuviško sintezatoriaus SAPI sąsaja. // Automatika ir valdymo technologijos - 2003. Kaunas, Technologija, 2003, pp. 45-48.
10. Kasparaitis P. (2005). Kompiuterinė lingvistika. Kirčiavimas. Transkribavimas. Interaktyvus [žiūrėta 2015-01-13] Prieiga internete: http://www.mif.vu.lt/~pijus/CL/KircTr.pdf
11. Kazlauskienė A., Raškinis G., Vičiūnas A. (2010). Automatinis lietuvių kalbos žodžių skiemenavimas, kirčiavimas, transkribavimas. Vytauto Didžiojo universitetas, Kaunas. p. 82-86.
12. Laurinčiukaitė, S. (2008). Lietuvių šnekos atpažinimo akustinis modeliavimas. Daktaro disertacija. VGTU, Vilnius. 2008, p. 108

13. Lippmann, R. P. Recognition by Humans and Machines: Miles to Go Before We Sleep. Speech Communication, April 1996, vol. 18, p. 247-248.
14. Loizou, P.; Spanias, A. High performance alphabet recognition. IEEE Trans. on Speech and Audio Processing, November 1996, vol.4, no 6, p. 430-445.
15. Maskeliūnas R. (2009). Lietuviškų balso komandų atpažinimas daugybinių transkripcijų pagrindu: daktaro disertacija: technologijos mokslai, informatikos inžinerija – 07T / Kauno technologijos universitetas. Kaunas. 157-159 p.
16. Norkevičius G., Raškinis G., Kazlauskienė A., Knowledge-based grapheme-tophoneme conversion of Lithuanian words // In Proceedings of the 10th International Conference on Speech and Computer , SPECOM - Patras, Greece, 2005, p. 235-238.
17. Phoneme Table for English (United States). Interaktyvus [žiūrėta 2014-10-09]. Prieiga per internetą: http://msdn.microsoft.com/enus/library/bb813894.aspx
18. Phoneme Table for Spanish (United States). Interaktyvus [žiūrėta 2014-10-09]. Prieiga per internetą: http://msdn.microsoft.com/enus/library/bb856944.aspx\
19. Rabiner R., Juang B. H. (1993). Fundamentals of Speech Recognition. Prentice-Hall, Inc. Upper Saddle River, NJ, USA. ISBN:0-13-015157-2. p. 3-5.
20. Rabiner R., Juang B.H. (2004). Automatic Speech Recognition‑A Brief History of the Technology Development // Elsevier Encyclopedia of Language and Linguistics (2005).
21. Raškinis. A., Raškinis G., Kazlauskienė A. (2003) „VDU bendrinės lietuvių šnekos universalus anotuotas garsynas“. Konferencijos „Informacinės technologijos 2003“ pranešimų medžiagoje, IX–(28–34).
22. Ringelienė, Ž., Filipovič, M. (2011). Žodžių atpažinimo, grįsto paslėptaisiais Markovo modeliais, vizualizavimo ir analizės programinė įranga. ISSN 1392-0561. Interaktyvus [žiūrėta 2014-01-14] Prieiga per internetą: http://www.leidykla.eu/fileadmin/Informacijos_mokslai/2011-56/63-72.pdf
23. Rudžionis, A., Ratkevičius, K., Rudžionis, V., Kasparaitis, P., Šalna, B. (2001). Balso technologijų taikymo lietuvių kalbai analizė ir perspektyvinių veiklos krypčių pagrindimas: ataskaita / Kauno technologijos universitetas, Vilniaus universitetas, Teismo ekspertizės centras. Kaunas, Vilnius. 84 p. Interaktyvus [žūrėta 2014-01-19]. Prieiga per internetą: http://www.likit.lt/all/balso_tech/balsotech.zip
24. Ruzgys G., Kamarauskas J. (2011). Nuo teksto nepriklausomas albančiojo atpažinimas // 14-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis“ 2011 metų teminės konferencijos straipsnių rinkinys. ISBN 978-9955-28-835-0

25. Schultz T., Kirchhoff K. (2006) Multilingual Speech Processing. Academic Press, Apr 21, 2006 - 508 pages.
26. Šilingas, D.; Telksnys, L. (2004) Specifics of Hidden Markov Model for Larger Vocabulary Continous Speech Recognition. Informatica, 2004, vol. 15, no.(1), p. 93-110.
27. TLK-10-AM sisteminis ligų sąrašas. Interaktyvus [žiūrėta 2014 – 11 – 19]. Prieiga per internetą: http://ebook.vlk.lt/e.vadovas/index.jsp;
28. Vaičiūnas, A. (2006). Lietuviu kalbos statistinių modelių ir jų taikymo šnekos atpažinimui tyrimas, kai naudojami labai dideli žodynai: daktaro disertacija: fiziniai mokslai, informatika 09P / Vytauto Didžiojo universitetas, Kaunas. Interaktyvus [žiūrėta 2015-01-19]. Prieiga per internetą: http://donelaitis.vdu.lt/disertacijos/Vaiciunas_2006.pdf
29. „VisiVaistai“, Informacinė sistema. Interaktyvus [žiūrėta 2015-03-21]. Prieiga per internetą: www.visivaistai.lt.
30. X. Huang, A. Acero, F. Alleva, M. Hwang, L. Jiang, M. Mahajan. (1996). Making speech recognition usable. Microsoft Corporation, Readmond, WA 98052, USA.
31. X. Huang, L. Deng. (2004). Challenges in Adopting Speech Recognition. Communications of the ACM. Vol. 47, No. 1. 71 p.
32. Young, S. ir kt. (2006). The HTK Book (for HTK Version 3.2.1), 2 p. Interaktyvus [žiūrėta 2015 04 25]. Prieiga per internetą: http://nesl.ee.ucla.edu/projects/ibadge/docs/ASR/htk/htkbook.pdf

3. PRIEDAI

3.1 PRIEDAS. UPS fonemų lentelė ispanų kalbai, naudota UPS transkripcijų sudarymui

Balsės		
UPS Label	**Example Word**	**UPS Transcription of Example Word**
A	v**a**c**a**	B A K A
A J	**ai**re	A J DX E
A W	c**au**sa	K A W S A
E	p**e**ro	P E DX O
E J	Ab**ei**jón	A B E J X O N
E W	**Eu**ropa	E W DX O P A
I	p**i**co	P I K O
O	t**o**r**o**	T O DX O
O J	s**oi**s	S O J S
O W	S**ou**to	S O W T O
U	d**u**ro	D U DX O

Priebalsės		
UPS Label	**Example Word**	**UPS Transcription of Example Word**
B	**v**ino ; ca**br**a	B I N O ; K A B DX A
CH	mu**ch**o	M U CH O
CJ	**y**egüa	CJ E G W A
D	na**d**a	N A D A
DX	pu**r**o	P U DX O
F	**f**ácil	F A S I L
G	**g**ata	G A T A
J	v**i**udo	B J U D O
K	**c**asa	K A S A
L	**l**argo	L A DX G O
LJ	caba**ll**o	K A B A LJ O
M	**m**asa	M A S A
N	**n**oche	N O CH E
NJ	a**ñ**o	A NJ O
P	**p**adre	P A D DX E
RR	to**rr**e	T O RR E
S	**c**ielo ; **i**sla	S J E L O ; I S L A
T	an**t**es	A N T E S
W	s**u**ave	S W A B E
X	mu**j**er	M U X E DX

Prozodija			
UPS Label	**Description**	**Example Word**	**UPS Transcription of Example Word**
S1	primary stress	señor	S E I . S1 NJ O DX
S2	secondary stress	señorita	S2 S E I . NJ O . S1 DX I . T A
.	syllable break	Isla	S1 I S . L A
_!	intonation - exclamation		
_&	intonation - continue		
_,	intonation - incompletion rise		
_.	intonation – fall		
_?	intonation - question rise		
_s	silence		

3.2 PRIEDAS. Vardų sąrašas atpažinimui.

1. Austėja
2. Boleslovas
3. Cecilija
4. Donatas
5. Eimantas
6. Fausta
7. Gražvydas
8. Hansas
9. Izaokas
10. Jonas
11. Karolis
12. Laima
13. Martynas
14. Nojus
15. Oskaras
16. Patrikas
17. Kju
18. Ričardas
19. Sandra
20. Teodoras
21. Ulijona
22. Vacys
23. Wašington
24. Xsas
25. Ygrekas
26. Zacharijus

3.3 PRIEDAS. Diktorių sąrašas

Nr.	Moterys	Vyrai
1	FAGNRUM	MDARJEG
2	FEGLZAJ	MJURBIZ
3	FGINBAR	MLAUBAR
4	FIVEVAL	MROKKUO
5	FJULBAL	MSARNEM
6	FGINPAS	MANDMAR
7	FGINTRA	MJUOCES
8	FGRETUB	MKASRAT
9	FINDBEN	MZYGSVE
10	FLAUKLU	
11	FMILRAU	
12	FSEVBUT	

3.4 PRIEDAS. Vardų ir skaičių gramatikos

3.4.1 Skaičiai

```
<grammar xmlns:sapi="http://schemas.microsoft.com/Speech/2002/06/SRGSExtensions" xml:lang="es-ES" tag-format="semantics-ms/1.0" version="1.0" mode="voice" xmlns="http://www.w3.org/2001/06/grammar" sapi:alphabet="x-microsoft-ups">

    <rule id="Rule" scope="public">
        <one-of>

            <item>
                <one-of>
                <item>nulejs</item>
                <item>mulis</item>
                <item>nuhlejhs</item>
                <item>nuhlejs</item>
                </one-of>
                <tag>$._value = "0"</tag>
            </item>
            <item>
                <one-of>
                <item>vienas</item>
                <item>vihehnahs</item>
                </one-of>
                <tag>$._value = "1"</tag>
            </item>
            <item>
                <one-of>
                <item>doo</item>
                <item>dou</item>
                <item>douh</item>
                <item>duo</item>
```

<item>dua</item>
</one-of>
<tag>$._value = "2"</tag>
</item>
<item>
<one-of>
<item>tryis</item>
<item>trris</item>
<item>trriis</item>
<item>trriis</item>
<item>trys</item>
<item>triys</item>
<item>tris</item>
</one-of>
<tag>$._value = "3"</tag>
</item>
<item>
<one-of>
<item>kewturih</item>
<item>keaturih</item>
<item>keaturii</item>
<item>keturej</item>
<item>kevturii</item>
</one-of>
<tag>$._value = "4"</tag>
</item>
<item>
<one-of>
<item>penkii</item>
<item>penkiih</item>
<item>penki</item>
</one-of>
<tag>$._value = "5"</tag>

```
                    </item>
                    <item>
                        <one-of>
                        <item>sheshii</item>
                        <item>chechii</item>
                        </one-of>
                        <tag>$._value = "6"</tag>
                    </item>
<item>
                        <one-of>
                        <item>tashkas</item>
                        <item>taskas</item>
                        <item>thaskhahs</item>
                        <item>tashkhahs</item>
                        <item>tashkahs</item>
                        </one-of>
                        <tag>$._value = "."</tag>
                    </item>
                    <item>
                        <one-of>
                        <item>septinii</item>
                        <item>septiinii</item>
                        <item>ceptinii</item>
                        <item>septinej</item>
                        </one-of>
                        <tag>$._value = "7"</tag>
                    </item>
                    <item>
                        <one-of>
                        <item>astuonii</item>
                        <item>ashtuhohnii</item>
                        <item>ashtuhohniih</item>
                        </one-of>
```

```
                    <tag>$._value = "8"</tag>
                </item>
                <item>
                    <one-of>
                    <item>deviinii</item>
                    <item>debini</item>
                    <item>dewini</item>
                    <item>deviiniih</item>
                    <item>debjinese</item>
                    </one-of>
                    <tag>$._value = "9"</tag>
                </item>




            </one-of>
        </rule>
    </grammar>
```

3.4.2 Vardai

```
<grammar xmlns:sapi="http://schemas.microsoft.com/Speech/2002/06/SRGSExtensions"
xml:lang="es-ES" tag-format="semantics-ms/1.0" version="1.0" mode="voice"
xmlns="http://www.w3.org/2001/06/grammar" sapi:alphabet="x-microsoft-ups">
    <rule id="Rule" scope="public">
        <one-of>
<item>
                    <one-of>
                    <item>ahwstehyeh</item>
                    <item>austeja</item>
                    </one-of>
                    <tag>$._value = "austeja"</tag>
                </item>
<item>
                    <one-of>
                    <item>baolehslaovahs</item>
                    <item>boleslovas</item>
                    </one-of>
                    <tag>$._value = "boleslovas"</tag>
```

```
                </item>
<item>
                        <one-of>
                        <item>tsehtsihlihyeh</item>
                        <item>cecilija</item>
                        </one-of>
                        <tag>$._value = "cecilija"</tag>
                </item>
<item>
                        <one-of>
                        <item>daonaatahs</item>
                        <item>donatas</item>
                        </one-of>
                        <tag>$._value = "donatas"</tag>
                </item>
<item>
                        <one-of>
                        <item>ehiymahntahs</item>
                        <item>eimantas</item>
<item>eihmantas</item>
                        <item>eigmantas</item>
                        <item>eihmanhtas</item>
                        <item>eihmanhtahss</item>
                        <item>aeihmanthahs</item>
                        </one-of>
                        <tag>$._value = "eimantas"</tag>

                </item>
<item>
                        <one-of>
                        <item>fahwstah</item>
                        <item>fausta</item>
                        </one-of>
                        <tag>$._value = "fausta"</tag>
                </item>
<item>
                        <one-of>
                        <item>graazhviydahs</item>
                        <item>grazvydas</item>
                        </one-of>
                        <tag>$._value = "grazvydas"</tag>
                </item>
<item>
                        <one-of>
                        <item>haansahs</item>
                        <item>hansas</item>
                        </one-of>
                        <tag>$._value = "hansas"</tag>
                </item>
```

```
<item>
                    <one-of>
                    <item>ihzahaokahs</item>
                    <item>izaokas</item>
                    </one-of>
                    <tag>$._value = "izaokas"</tag>
</item>
<item>
                    <one-of>
                    <item>yaonahs</item>
                    <item>jonas</item>
                    </one-of>
                    <tag>$._value = "jonas"</tag>
</item>
<item>
                    <one-of>
                    <item>kahraolihs</item>
                    <item>karolis</item>
                    </one-of>
                    <tag>$._value = "karolis"</tag>
</item>
<item>
                    <one-of>
                    <item>laaiymah</item>
                    <item>laima</item>
                    </one-of>
                    <tag>$._value = "laima"</tag>
</item>
<item>
                    <one-of>
                    <item>mahrtiynahs</item>
                    <item>martynas</item>
                    </one-of>
                    <tag>$._value = "martynas"</tag>
</item>
<item>
                    <one-of>
                    <item>naoyuhs</item>
                    <item>nojus</item>
                    </one-of>
                    <tag>$._value = "nojus"</tag>
</item>
<item>
                    <one-of>
                    <item>aoskahrahs</item>
                    <item>oskaras</item>
                    </one-of>
                    <tag>$._value = "oskaras"</tag>
</item>
```

<item>
<one-of>
<item>paatrihkahs</item>
<item>patrikas</item>
</one-of>
<tag>$._value = "patrikas"</tag>
</item>
<item>
<one-of>
<item>kyuh</item>
<item>kju</item>
<item>qhju</item>
<item>khju</item>
<item>kjjhu</item>
<item>kjjhju</item>
<item>kjjhju</item>
</one-of>
<tag>$._value = "kju"</tag>
</item>
<item>
<one-of>
<item>rihchahrdahs</item>
<item>ricardas</item>
</one-of>
<tag>$._value = "ricardas"</tag>
</item>
<item>
<one-of>
<item>saandrah</item>
<item>sandra</item>
</one-of>
<tag>$._value = "sandra"</tag>
</item>
<item>
<one-of>
<item>tehaodaorahs</item>
<item>teodoras</item>
</one-of>
<tag>$._value = "teodoras"</tag>
</item>
<item>
<one-of>
<item>uhlihyaonah</item>
<item>ulijona</item>
<item>ulihjonah</item>
<item>uhlihjonah</item>
<item>ullihjonah</item>
<item>ullihjona</item>
<item>hulihjonah</item>

```
                        </one-of>
                        <tag>$._value = "ulijona"</tag>
</item>
<item>
                        <one-of>
                        <item>vahtsiys</item>
                        <item>vacys</item>
                        </one-of>
                        <tag>$._value = "vacys"</tag>
</item>
<item>
                        <one-of>
                        <item>vahshihngktaon</item>
                        <item>vasington</item>
                        </one-of>
                        <tag>$._value = "vasington"</tag>
                    </item>
<item>
                        <one-of>
                        <item>ihksahs</item>
                        <item>iksas</item>
                        </one-of>
                        <tag>$._value = "iksas"</tag>
</item>
<item>
                        <one-of>
                        <item>iygrehkahs</item>
                        <item>ygrekas</item>
<item>igrekas</item>
                        <item>iygrehkas</item>
                        <item>yigrejhkas</item>
                        <item>igrahkas</item>
                        <item>iygrahqas</item>
                        </one-of>
                        <tag>$._value = "ygrekas"</tag>
</item>
<item>
                        <one-of>
                        <item>zahthaarihyuhs</item>
                        <item>zacharijus</item>
<item>zajarijus</item>
                        <item>zagharijus</item>
                        <item>zsagharhijus</item>
                        <item>zakarhijus</item>
                        <item>zthajarijus</item>
                        </one-of>
                        <tag>$._value = "zacharijus"</tag>
</item>
```

```
            </one-of>
        </rule>
    </grammar>
```

3.5 PRIEDAS. Ligos kodo raidės reikšmių lentelė

Skyrius	**Skyriaus pavadinimas**	**Prieš kodą rašoma raidė**	**Tipas**
1 skyrius	*Kai kurios infekcinės ir parazitinės ligos*	A, B	Ypatingas
2 skyrius	*Navikai*	C, D	Ypatingas
3 skyrius	*Kraujo ir kraujodaros organų ligos bei tam tikri sutrikimai susiję su imuniniais mechanizmais*	D	Pagal sritį
4 skyrius	*Endokrininės, mitybos ir medžiagų apykaitos ligos*	E	Pagal sritį
5 skyrius	*Psichikos ir elgesio sutrikimai*	F	Pagal sritį
6 skyrius	*Nervų sistemos ligos*	G	Pagal sritį
7 skyrius	*Akies ir jos priklausinių ligos*	H	Pagal sritį
8 skyrius	*Ausies, nosies, burnos ir gerklės ligos*	H	Pagal sritį
9 skyrius	*Kraujotakos sistemos ligos*	I	Pagal sritį
10 skyrius	*Kvėpavimo sistemos ligos*	J	Pagal sritį
11 skyrius	*Virškinimo sistemos ligos*	K	Pagal sritį
12 skyrius	*Odos ir poodžio ligos*	L	Pagal sritį
13 skyrius	*Jungiamojo audinio ir skeleto bei raumenų sistemos ligos*	M	Pagal sritį
14 skyrius	*Urogenitalinės sistemos ligos*	N	Pagal sritį
15 skyrius	*Nėštumas, gimdymas ir laikotarpis po gimdymo*	O	Ypatingas
16 skyrius	*Tam tikros perinatalinio periodo ligos*	P	Ypatingas
17 skyrius	*Įgimtos formavimosi ydos, deformacijos ir chromosomų anomalijos*	Q	Ypatingas
18 skyrius	*Simptomai, požymiai ir nenormalūs klinikiniai bei laboratoriniai radiniai neklasifikuojami kitur*	R	Netaikoma
19 skyrius	*Traumos, apsinuodijimai ir kiti išorinių priežasčių padariniai*	S, T	Ypatingas
20 skyrius	Išorinės *sergamumo priežastys*	U, V, W, X, Y	Netaikoma
21 skyrius	*Sveikatos būklę veikiantys faktoriai ir kontaktai*	Z	Netaikoma

	su sveikatos tarnyba		
22 skyrius	*Ypatingų tikslų kodai*	U	Netaikoma

3.6 PRIEDAS. Komandų atpažinimo testų rezultatai pagal profilius

Diktorius	Atpažinimo tikslumas, %			
	vyr_zod	mot_zod	Default_zod	vyr_UPS
MDARJEG	98,8	98,2	99,5	97,2
MJURBIZ	99,2	90,2	97,8	97
MJUOCES	97,6	92	96,9	96,6
FLAUKLU	98,6	95,7	96,3	96,6
MANDMAR	99,7	96,3	99,1	96,1
FIVEVAL	97,1	98	96,9	96,1
MZYGSVE	98,8	93	96,1	96
MSARNEM	99,7	97,7	99,7	95,8
FEGLZAJ	97,8	97,9	99	95,1
FGINBAR	98,8	96	99	95,1
FGRETUB	99,1	96,9	97,3	94,2
FSEVBUT	94,2	92,6	96,9	94,2
FGINPAS	95	94,7	95	93,5
MROKKUO	94,7	87	94,5	93,3
MLAUBAR	96,2	92,1	95,5	92,9
MKASRAT	95,1	92,9	91,2	92,3
FJULBAL	97,5	96,1	96,2	92,2
FGINTRA	94,9	95,3	95,6	91
FAGNRUM	88,9	90	91,8	91
FINDBEN	90,1	92,9	88	90,2
FMILRAU	93,5	93	93,8	90,1
Vidurkis:	96,44	94,21	96,00	94,12

3.7 PRIEDAS. Geriausiai atpažįstamos komandos pagal profilį ar traskripcijas

Nr.	Vardas	Atpažinimas, %			
		mot_zod	Default_zod	vyr_zod	vyr_UPS
1	Austėja	90,7	98,3	99,3	99,5
2	Boleslovas	96	97,6	98,9	98,6

3	Cecilija	96,4	97,6	99,9	99,3
4	Donatas	98,2	98,1	99,9	99,9
5	Eimantas	98,8	98,9	96,5	97,2
6	Faustas	98,2	98,5	98,1	98,6
7	Gražvydas	95	98,9	99,3	98,7
8	Hansas	98,9	98,9	99	100
9	Izaokas	99,5	95,1	98,6	99,2
10	Jonas	94	95,5	97,1	96
11	Karolis	100	100	97,8	97,1
12	Laima	97,1	99,3	98,6	99,2
13	Martynas	99,5	98,1	97,7	99,9
14	Nojus	97,9	99,1	97,9	98,2
15	Oskaras	99	100	100	97,2
16	Patrikas	99	99,9	99,9	99,8
17	Qju	60	70,9	85	88,3
18	Ričiardas	92	95,1	89,8	97,1
19	Sandra	98,2	99,8	99,6	99,8
20	Teodoras	96,1	99,8	99,1	97,6
21	Ulijona	92,4	94,1	96	90,9
22	Vacys	97,9	95,2	93	100
23	Wašington	93,9	96,6	98,1	97,1
24	Xsas	97	97	94,2	98,2
25	Ygrekas	93,3	95,6	95,1	96,9
26	Zacharijus	94,9	94	96	75,3
Vidurkis, %:		95,15	96,61	97,09	96,91

3.8 PRIEDAS. Ligų kodų atpažinimo su kitakalbiu atpažintuvu ir sąsajos su internetu programos kodas

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Speech.Recognition;
using System.Speech.Recognition.SrgsGrammar;
using System.Diagnostics;


namespace Demo
{
```

```
    public partial class Demo : Form
    {
        private SpeechRecognitionEngine recognizer = new SpeechRecognitionEngine();
        string kodas = "",komanda="", kodas_old="";
        int num = 0;

        public Demo()
        {
            InitializeComponent();
            label2.Text = "Spauskite Komandos";

        }

        private void Start_Click(object sender, EventArgs e)
        {
    recognizer.SetInputToDefaultAudioDevice();
    label2.Text = "Sakykite";
            recognizer.SpeechRecognized += new
EventHandler<SpeechRecognizedEventArgs>(recognizer_SpeechRecognized);
            recognizer.RecognizeAsync(RecognizeMode.Multiple);
            return;
        }
      //  label2.Text = "Spauskite Skaiciai";
        void recognizer_SpeechRecognized(object sender, SpeechRecognizedEventArgs e)
        {
            num++;
            string word = e.Result.Text.ToString();
        //    textBox1.Text = word;
           string komanda = e.Result.Semantics.Value.ToString();
            kodas_old = kodas;  kodas = kodas + komanda;
            if (num == 1) { textBox1.Text = komanda; }
            if (num == 2) textBox2.Text = komanda;
            if (num == 3) textBox3.Text = komanda;
            if (num == 4) textBox4.Text = komanda;
            if (num == 5) { textBox5.Text = komanda; label2.Text = "Spauskite Rezultatas"; }
          //  webBrowser1.Navigate(komanda);
      //      SendKeys.Send(komanda);
        }

        private void Vardai_Click(object sender, EventArgs e)
        {
            Grammar gramatika = new Grammar("vardai.grxml", "Rule");
            recognizer.LoadGrammar(gramatika);
            label2.Text = "Spauskite Start";
            richTextBox1.Clear();
            richTextBox1.LoadFile("vardai.rtf");
            richTextBox2.Clear();
            richTextBox2.LoadFile("skaiciai.rtf");

        }

        private void Stop_Click(object sender, EventArgs e)
        {

       //     recognizer.RecognizeAsyncStop();
            richTextBox1.Clear();
            richTextBox2.Clear();
            textBox1.Clear();
            textBox2.Clear();
            textBox3.Clear();
            textBox4.Clear();
```

```
            textBox5.Clear();
            textBox6.Clear();
            label2.Text = "Ačiū";
            recognizer.UnloadAllGrammars();


        }

        private void Skaiciai_Click(object sender, EventArgs e)
        {
            Grammar gramatika = new Grammar("skaiciai.grxml", "Rule");
            recognizer.LoadGrammar(gramatika);
            label2.Text = "Sakykite";

        }
        private void Liga_Click(object sender, EventArgs e)
        {
            label2.Text = "Spauskite Stop";
            if (kodas == "H65.9")
            {
                textBox6.Text = "Ausies uždegimas";
                string adresas =
"http://ligos.sveikas.lt/lt/ligos/ausunosiesgerkles_ligos/isorines_ausies_uzdegimas_otitas";
                webBrowser1.Navigate(textBox6.Text);
                webBrowser1.Navigate(adresas);
            };
            if (kodas == "L02.4")
            {
                textBox6.Text = "Furunkulas";
                string adresas =
"http://ligos.sveikas.lt/lt/ligos/odos_ligos/sunvote_furunkulas";
                webBrowser1.Navigate(textBox6.Text);
                webBrowser1.Navigate(adresas);
            };
            if (kodas == "K71.3")
            {
                textBox6.Text = "Hepatitas";
                string adresas = "http://www.sveikoskepenys.lt/kepenu-ligos-ir-pazeidimo-
formos/hepatitas/";
                webBrowser1.Navigate(textBox6.Text);
                webBrowser1.Navigate(adresas);
            }
        }

        private void button1_Click(object sender, EventArgs e)
        {

            switch (num)
            {
                case 1: textBox1.Clear(); label2.Text = "Sakykite";
                    break;
                case 2: textBox2.Clear();
                    break;
                case 3: textBox3.Clear();
                    break;
                case 4: textBox4.Clear();
                    break;
                case 5: textBox5.Clear(); label2.Text = "Sakykite";
                    break;
            }
            num--; kodas = kodas_old;
```

```
        }
        private void Narsykle_Click(object sender, EventArgs e)
        {
            string adresas = "http://www.google.lt";
            webBrowser1.Navigate(textBox6.Text);

            webBrowser1.Navigate(adresas);
        }
        private void webBrowser1_DocumentCompleted(object sender,
WebBrowserDocumentCompletedEventArgs e)
        {

        }


        }
    }
```

3.9 PRIEDAS. Su duomenų baze susieto atpažintuvas programos kodas

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.Speech.Recognition;
using System.Speech.Recognition.SrgsGrammar;
using System.Windows.Forms;
using System.Diagnostics;


namespace MSI
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public SpeechRecognitionEngine rec = new SpeechRecognitionEngine();
        public int x,y = 0;
        public MainWindow()
        {
            InitializeComponent();
        }


        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            button_Stop.IsEnabled = false;
            button_Start.IsEnabled = false;
            InitializeComponent();
```

```
            rec.SetInputToDefaultAudioDevice();

            rec.SpeechRecognized += new
EventHandler<SpeechRecognizedEventArgs>(rec_SpeechRecognized);
            rec.RecognizeCompleted += new
EventHandler<RecognizeCompletedEventArgs>(rec_RecognizeCompleted);
        }

        private void button_Start_Click(object sender, RoutedEventArgs e)
        {
            button_Start.IsEnabled = false;
            button_Stop.IsEnabled = true;


            rec.RecognizeAsync(RecognizeMode.Multiple);
        }

        void rec_RecognizeCompleted(object sender, RecognizeCompletedEventArgs e)
        {
          //  System.Windows.Forms.MessageBox.Show("suveike");
        }

        void rec_SpeechRecognized(object sender, SpeechRecognizedEventArgs e)
        {
            //throw new NotImplementedException();
            listBox_atpazinta.Items.Clear();
            listBox_atpazinta.Items.Add(e.Result.Semantics.Value);




                //DB-Ligos
                if (x == 2)
                {
                    var data = new
medicinine_dbDataSetTableAdapters.Nusiskundimai_LigosTableAdapter();
                    var data2 = new medicinine_dbDataSetTableAdapters.LigosTableAdapter();
                    var nusiskund = from p in data.GetData()
                                    where p.Disease.ToString() ==
e.Result.Semantics.Value.ToString() // e.Result.Text
                                    select new
                                    {
                                        Nusiskund = p.Nusiskundimai,
                                    };
                    listBox_atrinka.Items.Clear();
                    foreach (var s in nusiskund)
                    {
                        listBox_atrinka.Items.Add(s.Nusiskund.ToString());

                    }

                    var info = from p in data2.GetData()
                               where p.Disease.ToString() ==
e.Result.Semantics.Value.ToString()
                               select p.Info;
                    textBox_info.Clear();
                    foreach (string s in info)
                    {
                        textBox_info.Text = s;
```

```
                    }
                }


                if (x == 3)
                {
                    var data = new
medicinine_dbDataSetTableAdapters.Nusiskundimai_LigosTableAdapter();


                    var ligos = from p in data.GetData()
                                where p.Nusiskundimai == e.Result.Semantics.Value.ToString()
// e.Result.Text
                                select new
                                {
                                    ligos = p.Disease,
                                };
                    foreach (var s in ligos)
                    {
                        listBox_atrinka.Items.Add(s.ligos.ToString());
                    }
                }
                if ((listBox_atrinka.Items.Count > 5)&&(x==3)) //atrenka besikartojancius
                {
                    string[] itemm = new string[listBox_atrinka.Items.Count];

                    for (int i = 0; i < listBox_atrinka.Items.Count; i++)
                    {
                        itemm[i] = listBox_atrinka.Items[i].ToString();
                    }


                    var ranking = (from item in itemm
                                   group item by item into r
                                   orderby r.Count() descending
                                   select new { name = r.Key, rank = r.Count() }).Take(5);
                    textBox_info.Clear();
                    foreach (var s in ranking)
                    {
                       textBox_info.Text= (textBox_info.Text+ s.name + " - " + s.rank+", ");
                    }

            }

        }

        private void button_Stop_Click(object sender, RoutedEventArgs e)
        {
          //  rec.RecognizeAsync();
            button_Start.IsEnabled = true;
            button_Stop.IsEnabled = false;

            rec.RecognizeAsyncCancel();

            listBox_atrinka.Items.Clear();
            listBox_atpazinta.Items.Clear();
            textBox_info.Clear();
        }


        private void button_Ligos_Click(object sender, RoutedEventArgs e)
```

```
        {
            rec.UnloadAllGrammars();
            Grammar gramatika = new Grammar("ligos.grxml", "Rule");
            rec.LoadGrammar(gramatika);

            listBox_atpazinta.Items.Clear();
            listBox_atrinka.Items.Clear();

            button_Start.IsEnabled = true;

            listBox_gramatika.Items.Clear();
            listBox_gramatika.Items.Add("Akių nudegimai");
            listBox_gramatika.Items.Add("Alergija");
            listBox_gramatika.Items.Add("Depresija");
            listBox_gramatika.Items.Add("Dispepsija");
            listBox_gramatika.Items.Add("Egzema");
            listBox_gramatika.Items.Add("Hemorojus");
            listBox_gramatika.Items.Add("Žvynelinė");
            listBox_gramatika.Items.Add("Migrena");
            listBox_gramatika.Items.Add("Nutukimas");
            listBox_gramatika.Items.Add("Vėjaraupiai");

            x = 2;
        }
        private void button_Nusiskund_Click(object sender, RoutedEventArgs e)
        {
            rec.UnloadAllGrammars();
            Grammar gramatika = new Grammar("nusisk.grxml", "Rule");
            rec.LoadGrammar(gramatika);

            listBox_atpazinta.Items.Clear();
            listBox_atrinka.Items.Clear();

            button_Start.IsEnabled = true;

            listBox_gramatika.Items.Clear();
            listBox_gramatika.Items.Add("Dusulys");
            listBox_gramatika.Items.Add("Galvos skausmas");
            listBox_gramatika.Items.Add("Vėmimas");
            listBox_gramatika.Items.Add("Nuovargis");
            listBox_gramatika.Items.Add("Karščiavimas");
            listBox_gramatika.Items.Add("Pykinimas");
            listBox_gramatika.Items.Add("Kosulys");
            listBox_gramatika.Items.Add("Pilvo skausmas");
            listBox_gramatika.Items.Add("Silpnumas");
            listBox_gramatika.Items.Add("Viduriavimas");
            listBox_gramatika.Items.Add("Galvos svaigimas");
            listBox_gramatika.Items.Add("Gerklės skausmas");


            x = 3;


        }
    }
}
```