ŠIAULIŲ UNIVERSITETAS
TECHNOLOGIJOS FAKULTETAS
ELEKTRONIKOS KATEDRA

Ernestas Braubartas

LIETUVIŲ KALBOS ATPAŽINIMAS, PANAUDOJANT JULIUS PROGRAMINĘ ĮRANGĄ

Magistro darbas

Vadovas

doc. dr. G. Daunys

ŠIAULIAI, 2008

ŠIAULIŲ UNIVERSITETAS
TECHNOLOGIJOS FAKULTETAS
ELEKTRONIKOS KATEDRA

TVIRTINU

Katedros vedėjas doc. dr. G. Daunys

2008

LIETUVIŲ KALBOS ATPAŽINIMAS, PANAUDOJANT JULIUS PROGRAMINĘ ĮRANGĄ

Magistro darbas

Recenzentas

(parašas) dr. D. Dervinis
2008

Vadovas

(parašas) doc.dr. G. Daunys
2008

Atliko

RM6 gr. stud.
(parašas) E. Braubartas
2008

ŠIAULIAI, 2008

Braubartas E. Speech Recognition of Lithuanian Using Julius Software: Master thesis of signal technology/research advisor Doc. Dr. G. Daunys; Šiauliai University, Faculty of Technology, Department of Electronics. – Šiauliai, 2008. – 48p.

SUMMARY

The theme of Master project of signal technology is actual, because not enough usual information introduction ways. Therefore information search and processing, complicated devices and programs control would be more handily if computers and devices understood human speech. Similar systems are designing for many years in the world. However Lithuanian speech recognition systems are still developing in nowadays.

The thesis treats of isolated Lithuanian words recognition dividing them into category and using Hidden Markov Models. The idea of research is to explore categorization of Lithuanian words influence on the accuracy of recognition. The recognition of single words and word groups is under research too.

Acoustic model is constructed by using HTK toolkit which is based on Hidden Markov Models. Categorization of words is described with Backus-Naur form. Experiments are made with Julius software speech recognition system Julian witch performs words category based recognition.

Best results are got trying to recognize single words set into categories. The accuracy rate of recognition reaches 91 %. While trying to recognize uncategorized word sequences – the accuracy rate of recognition reaches only 51 %. The accuracy rate of Microsoft Office Word 2003 control menu recognition reaches 82 %.

SANTRUMPŲ IR TERMINŲ ŽODYNAS

BNF – Bakus Naur forma

HTK – Hidden Markov Toolkit

LPC – tiesinės prognozės kodavimas

MLF – pagrindinis žymėjimo failas

MFCC – dažnių skalės kepstro koeficientai

PMM – Paslėptieji Markovo modeliai

ILIUSTRACIJŲ SĄRAŠAS

1.1 pav. Atskirai tariamų žodžių atpažinimas 12
1.2 pav. Viterbi algoritmas panaudotas atskirai tariamų žodžių atpažinime 17
2.1 pav. Tarimo žodyno sudarymo schema 26
2.2 pav. Fonemų transkripcijų sudarymo schema 30
2.3 pav. Kalbos įrašų kodavimo schema 31
2.4 pav. Paslėptųjų Markovo modelių apmokymo schema 33
3.1 pav. Žodžių atpažinimo tikslumas 43

TURINYS

ĮVADAS 7
1. KALBOS ATPAŽINIMO APŽVALGA 9
1.1. Skaitmeninio garso signalo konvertavimas 9
1.2. Fonemų išskyrimas 10
1.3. Paslėptieji Markovo modeliai 12
1.3.1. Baum-Welsh algoritmas 14
1.3.2. Viterbi algoritmas 15
1.3.3. Sklidimo pirmyn-atgal algoritmas 17
1.4. Skaičiavimų mažinimas ir tikslumo didinimas 19
1.5. Akustiniai kalbos atpažinimo metodai 20
1.6. Dirbtinio intelekto metodai 21
1.7. Kalbos atpažinimo tyrimų analizė 23
2. LIETUVIŲ KALBOS ATSKIRAI TARIAMŲ ŽODŽIŲ ATPAŽINIMAS 25
2.1. Akustinis modelis 25
2.1.1. Tarimo žodynas 25
2.1.2. Transkripcijų kūrimas 29
2.1.3. Kalbos įrašų kodavimas 31
2.1.4. Paslėptųjų Markovo modelių apmokymas 32
2.2. Žodžių skirstymas į kategorijas 35
2.3. Julius programinė įranga 40
3. TYRIMO REZULTATAI 42
IŠVADOS 45
LITERATŪRA 46
PRIEDAS 48

ĮVADAS

Sparčiai tobulėjant ir sudėtingėjant informacinėms technologijoms bei didėjant informacijos kiekiui nebeužtenka įprastų informacijos įvedimo priemonių, tokių kaip pelė ar klaviatūra. Žmonėms tarpusavio patogiausia ir greičiausia bendravimo forma yra kalba. Todėl ieškoti ir apdoroti informaciją, valdyti sudėtingus įrenginius ir programas daug patogiau būtų jei kompiuteriai ir įvairūs įrenginiai suprastų žmogaus kalbą. Greitėjant gyvenimo tempui žmogus vienu metu priverstas atlikinėti keletą darbų. Šis uždavinys labai palengvėtu, jei dalį darbo, atliekamo rankomis perkeltumėm balsui. Kadangi dabar žmogus didžiąją informacijos dalį kaupia kompiuteryje, balsą galima panaudoti patogiai ir greitai suvedant informaciją į jį, taip padarant kitas duomenų įvedimo priemones tik kaip papildomas. Balsu valdant kompiuterį ir kitus įrenginius žmogus galėtų laisviau judėti, būtų daug paprastesnis pats valdymo procesas. Pasaulyje panašios sistemos kuriamos jau daugelį metų. Kadangi kiekvienas žmogus kalba skirtingai, sukurti kalbos atpažinimo sistemas sudėtinga. Taip pat daug problemų sukelia aplinkinis triukšmas. Tačiau šiuo metu pasiekta neblogų rezultatų. Todėl netolimoje ateityje kiekvienas galės dirbti su kompiuteriu kalbėdamas. Deja šios sistemos pritaikytos tik plačiai vartojamoms kalboms, tokioms kaip anglų, japonų, prancūzų, vokiečių. Lietuvių kalba turi sudėtingą struktūrą, dėl kurios netinka kitoms kalboms sukurti akustiniai modeliai. Todėl svarbu kurti lietuvių kalbos atpažinimo sistemas, kad šios kalbos vartojimas nebūtų išstumtas modernių technologijų srityje. Šiuo metu lietuvių kalbos atpažinimo sistemos yra dar tik kūrimo stadijoje.

Srityse, kuriose nereikia rišlios kalbos atpažinimo, tikslinga žodžius skirstyti į kategorijas, taip sumažinant atpažįstamų žodžių kiekį ir tuo pačiu labai padidinant atpažinimo tikslumą. Toks kalbos atpažinimo būdas, tinkamas naudoti įrenginių valdymui, formų pildymui, įvairių programų komandų ir meniu valdymui.

Šio darbo tikslas – ištirti lietuvių kalbos žodžių skirstymo į kategorijas įtaką atpažinimo tikslumui.

Darbo uždaviniai: sukurti lietuvių kalbos akustinį modelį, sudaryti žodžių kategorijų rinkinius, palyginti žodžių atpažinimo tikslumą atliekant eksperimentus su žodžių kategorijom ir be jų, palyginti žodžių grupių ir pavienių žodžių atpažinimą.

Darbas bus atliktas naudojant vienas iš populiariausių kalbos atpažinimo būdų – paslėptųjų Markovo modelių (PMM) taikymo metodiką, kuri remiasi tikimybinių modeliu 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 [1].

Paslėptųjų Markovo modelių apmokymas bus atliktas *HTK* paketo įrankiais [2]. Žodžių kategorijos bus aprašomos *Backus-Naur* formatu [3]. Eksperimentai bus atliekami ir rezultatai gaunami naudojant, *Julius* programinės įrangos įrankius bei šio paketo, žodžių kategorijų pagrindu veikiančią, *Julian* kalbos atpažinimo sistemą [4].

1. KALBOS ATPAŽINIMO APŽVALGA

Kalbos atpažinimas – tai iš garso plokštės ateinančio skaitmeninio garso signalo konvertavimas į atpažintą kalbą. Kalbos atpažinimo dalys:

1. Skaitmeninio garso signalo konvertavimas į spektrą, tinkamą išskirti reikiamą informaciją kalbos atpažinimui.
2. Fonemų išskyrimas.
3. Kalbos atpažinimo metodo panaudojimas.
4. Ištartų fonemų susiejimas su žodžiais.

1.1. Skaitmeninio garso signalo konvertavimas

Atliekant kalbos atpažinimą pirmiausia iš garso plokštės ateinantis skaitmeninis garso signalas konvertuojamas į tokį formatą, iš kurio galima gauti daugiau naudingos informacijos. Skaitmeninis garso signalas – tai amplitudžių srautas suskaidytas 16000 kartų per sekundę. Ši informacija nelabai naudinga kalbos atpažinimui, nes per daug sunku nustatyti, kokius nors požymius, kuriuos būtų galima susieti su tuo kas buvo pasakyta.

Tam kad padaryti požymių atpažinimą lengvesnį, skaitmeninis garso signalas transformuojamas į spektrą. Ši transformacija atliekama naudojant Greitąją Furjė transformaciją. Spektre galima nustatyti garso dažnines komponentes. Iš dažninių komponenčių galima nustatyti požymius, susiejant juos su ištartomis fonemomis.

Greitoji Furjė transformacija garso duomenis analizuoja kas 0,01 sekundės ir konvertuoja į spektrą. Kiekvienas 0,01 sekundės rezultatas yra dažnio dedamosios amplitudės stulpelis, aprašantis kaip garsas girdimas 0,01 sekundės. Kalbos atpažinimo variklis turi duomenų bazę iš keleto tūkstančių tokių stulpelių, kurie atspindi įvairius žmogaus balso sukeliamus garsus. Garsas atpažįstamas sulyginant jį su panašiausiu įrašu duomenų bazėje pagal požymio skaičių. Iš tikrųjų yra keletas sugeneruotų požymio skaičių kiekvienai 0,01 sekundės daliai.

Kalbos atpažinimo pradžioje gaunamas 16000 verčių per sekundę skaitmeninio garso signalo srautas. O panaudojus Greitąją Furjė transformaciją ir turint garso šablonų duomenų bazę gaunamas 100 požymio skaičių per sekundę. Turint šiuos duomenis galima aprašyti garsus ir atpažinti ištartus žodžius [5].

1.2. Fonemų išskyrimas

Idealiu atveju būtų galima susieti kiekvieną požymio skaičių su fonema. Jei garso signalo dalies požymio skaičius būtų *#52*, tai galėtų reikšti, kad žmogus ištarė garsą *„h“*. Požymis *#53* galėtų reikšti garsą *„f“* ir panašiai. Jei būtų taip paprasta, tai būtų lengvą suprasti, kurias fonemas žmogus taria. Tačiau dėl keleto priežasčių taip nėra:

1. Kiekvieną kartą žmogus tą patį žodį ištaria skirtingai. Žmogus neištaria tiksliai tokio pat garso tai pačiai fonemai.
2. Mikrofono triukšmas ir žmogų supanti aplinka kartais būna priežastis to, kad atpažinimo sistema priskiria kitokį požymio skaičių, negu tada kai žmogus kalba tyliame kambaryje su aukštos kokybės mikrofonu.
3. Fonemos garsas priklauso nuo šalia esančių fonemų.

Triukšmas ir kitos problemos išsprendžiamos leidžiant požymio skaičių naudoti daugiau nei vienai fonemai ir panaudojant statistinius modelius, kad išsiaiškinti kurios fonemos yra tariamos. Tai galima padaryti, nes fonema trunka santykinai ilgą laiką, nuo 50 iki 100 požymio skaičių ir gali būti, kad vienas ar daugiau garsų vyraus tuo metu. Taigi įmanoma numatyti kuri fonema buvo tarta.

Kad kalbos atpažinimo sistema suprastu, kaip fonema skamba, mokymo įrankis apdorojo šimtus fonemų įrašų. Jis analizuoja kiekvieną 0,01 sekundės šių įrašų ir nustato požymio skaičių. Iš to jis sužino statistiką, kokia tikimybė atsirasti konkretaus požymio skaičiaus konkrečioje fonemoje. Taigi fonemai *„h“* bet kurioje 0,01 sekundėje požymio *#52* pasirodymo tikimybė gali būti 55%, požymio *#189* tikimybė gali būti 30%, požymio #53 tikimybė gali būti 15%. O raidės *„f“* kiekviena 0,01 sekundės gali turėti požymio *#52* tikimybę 10%, požymio *#189* tikimybę 10% ir požymio *#53* tikimybę 80%.

Tikimybių analizė mokymo metu yra vėliau panaudojama kalbos atpažinimo metu. Šeši požymio skaičiai girdimi atpažinimo metu gali būti: *52, 52, 189, 53, 52, 52*. Atpažinimo sistema suskaičiuoja, kad turimas garsas yra *„h“* tikimybę, ir bet kurios kitos tarkim *„f“* tikimybę. Kad garsas bus *„h“* tikimybė yra *80% + 80% + 30% + 15% + 80% + 80% / 7= 60,8%*. Kad garsas bus *„f“* tikimybė yra *10% + 10% +10% + 10% + 80% + 10% + 10% / 7 = 20%*. Taigi pagal šiuos duomenis *„h“* tikimybė yra didesnė.

Kalbos atpažinimo sistemai reikia žinoti, kada viena fonema baigiasi, o kada prasideda kita. Kalbos atpažinimo varikliai tai atskiria pasinaudodami paslėptaisiais Markovo modeliais. Sakykime, kad atpažinimo sistema girdi žodį su *„h“* fonema, kuri eina po *„e“* fonemos. *„e“* fonema turi 75% tikimybę požymiui *#82* kiekvienai 0,01 sekundės, požymis *#98* turi 15% tikimybę, o požymis *#52* turi 10% tikimybę. Požymis *#52* patenka ir į *„h“*. Duomenys galėtu atrodyti taip: *52,*

52, 189, 53, 52, 52, 82, 52, 82, ir t.t. Žiūrint į šiuos požymius galima matyti, kad požymiai *#52* yra susigrupavę pradžioje, o požymiai *#82* susigrupavę pabaigoje. Tad riba tarp fonemų yra kažkur per vidurį. Atpažinimo sistema ją nustato naudodama paslėptuosius Markovo modelius.

Kalbos atpažinimo sistema gali nustatyti kada kalba prasideda, o kada baigiasi, nes ji turi tylos fonemą. Kiekvienas požymio skaičius turi pasirodymo tyloje tikimybę, kaip ir bet kuri kita fonema.

Fonemos skambesys keičiasi priklausomai nuo to kokia fonema yra prieš ją ir po jos. Taip pat fonemos šiek tiek persidengia. Dalis vienos fonemos yra kitoje fonemoje. Kalbos atpažinimo varikliai išsprendžia šią problemą sukuriant tribalsius, kurie yra fonemos apsuptos kitomis fonemomis. Tribalsių yra labai daug, todėl kad mažiau apkrauti kompiuterį ir padidinti atpažinimo greitį panašiai skambantys tribalsiai yra grupuojami.

Fonemos garsas nėra pastovus. *„t“* garsas pradžioje yra tylus, po to pasidaro staigus didelio dažnio triukšmo protrūkis, kuris pamažu nutyla. Tai išsprendžiama suskaidant kiekvieną fonemą į keletą segmentų. Kada kuris segmentas pasibaigia ir kada prasideda sužinoma taip pat kaip ir atskiriant fonemų pradžią ir pabaigą.

Kalbos atpažinimo sistema spėlioja tam tikrą skaičių skirtingų būsenų iš karto. Kiekviena būsena turi fonema ir prieš tai buvusių fonemų sąrašą. Spėjama būsena turinti didžiausią balą naudojama kaip galutinis atpažinimo variantas.

Kai kalbos atpažinimo sistema pradeda klausytis, ji turi vieną spėjamą būseną. Ji mano kad žmogus nekalba ir atpažinimo sistema girdi tylos fonemą. Kas kiekviena 0,01 sekundės sistema spėja kad žmogus pradeda kalbėti ir prideda naują būsena kiekvienai fonemai sukurdama 50 naujų būsenų kiekvienai priskiriant tikimybę. Po pirmosios 0,01 sekundės atpažinimo sistema turi 51 spėjamą būseną.

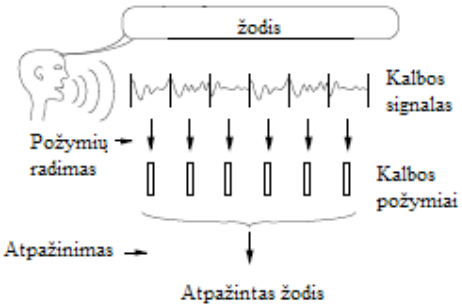
Po 0,01 sekundės gaunamas kitas požymio skaičius. Esamo požymio tikimybės perskaičiuojamos su naujuoju požymiu. Tada kiekviena fonema turi perėjimo tikimybę į dar kitą fonemą. Taigi sukuriamos 2550 naujų būsenų. Kiekvienos būsenos tikimybės yra pirmos 0,01 sekundės tikimybės padaugintos iš antros 0,01 sekundės tikimybių. Po 0,02 sekundės atpažinimo sistema turi 2601 spėjamų būsenų.

Tas pats veiksmas kartojamas kas kiekvieną 0,01 sekundės. Kiekvieno naujo spėjimo tikimybės yra pirminio spėjimo tikimybės padaugintos iš tikimybių gautų iš 0,01 sekundės dalių. Pabaigoje spėjimas su didžiausia tikimybe panaudojamas kaip atpažinimo rezultatas [6].

Jei spėjimo tikimybė daug mažesnė už didžiausią tikimybę tada spėjimas atmetamas.

1.3. Paslėptieji Markovo modeliai

Atliekant atskirai tariamų žodžių atpažinimą, surandami ištarto kalbos signalo požymiai. Po to fiksuotame žodyne surandama kalbos signalą atitinkanti fonemų seka (1.1 pav.).



1.1 pav. Atskirai tariamų žodžių atpažinimas

Kiekvieną ištartą žodį galima atvaizduoti, kaip garso signalo požymių seką *O*:

$$O = o_1, o_2, ..., o_t; \tag{1.1}$$

čia o_t yra garso signalo požymis išskirtas *t* laiku. Atskirai tariamų žodžių atpažinimas gali būti aprašytas taip:

$$\arg\max_i \{P(w_i \mid O)\}; \tag{1.2}$$

čia w_i žodyno žodis. Kadangi galimų požymio sekų *O* yra labai daug. Šios tikimybės $P(w_i|O)$ tiesiogiai įvertinti neįmanoma. Ji skaičiuojama naudojant *Bayes* formule ir užrašoma taip:

$$P(w_i \mid O) = \frac{P(O \mid w_i)P(w_i)}{P(O)}; \tag{1.3}$$

čia $P(w_i)$ yra *a priori* tikimybė, kad bus ištartas žodis w_i, $P(O|w_i)$ yra tikimybė, kad tariant žodžius w_i, bus stebima garso signalo požymių seka *O*, $P(O)$ yra tikimybė, kad bus stebima požymių seka *O*. Kadangi $P(O)$ tikimybė duotam garso signalui yra nekintanti, tai kalbos atpažinimas gali būti aprašytas taip:

$$\arg\max_i \{P(O \mid w_i)P(w_i)\}; \tag{1.4}$$

čia $P(O|w_i)$ vadinamas akustiniu modeliu, o $P(w_i)$ – kalbos modeliu. Skaičiuojamos tikimybės visiems galimiems modeliams ir parenkamas tas žodis, kurio modelio tikimybė yra didžiausia [2].

Kalbos atpažinimo akustiniam modeliui modeliuoti naudojami paslėptieji Markovo modeliai. Tai statistinis atpažinimo metodas. Šiuo metodu signalas gali būti aprašomas parametriniu atsitiktiniu procesu, kurio parametrai gali būti tiksliai surasti apibrėžtu būdu.

PMM parametrai:

1. N – modelio būsenų skaičius. Būsenos yra surištos taip, kad iš kiekvienos būsenos būtų galima patekti į bet kurią kitą ar net tą pačią būseną. Tačiau įmanomi ir kitokie sąryšiai. Atskiros būsenos žymimos $S=S_1, S_2, ..., S_N$, o būsena laiko momentu t – q_t.

2. M – skaičius skirtingų stebimų simbolių vienoje būsenoje. Stebimi simboliai atitinka fizikinę modeliuojamos sistemos gaunamą informaciją. Atskiri simboliai žymimi $V = v_1, v_2, ..., v_M$.

3. Būsenų perėjimo tikimybės $A = a_{ij}$, kur

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i] \tag{1.5}$$

Tuo atveju, kai iš bet kurios būsenos galima patekti į bet kurią kitą būseną, $a_{ij} > 0$ visiems i ir j. Kitais atvejais, $a_{ij} = 0$ vienai ar daugiau i ir j porų.

4. Būsenoje j stebimų simbolių tikimybės pasiskirstymas, $B = \{b_j(k)\}$, kur

$$b_j(k) = P[v_k, kai\ t | q_t = S], \qquad \begin{matrix} 1 \le j \le N \\ 1 \le k \le M \end{matrix} \tag{1.6}$$

5. Pradinis būsenų pasiskirstymas $\pi = \{ \pi_i \}$, kur

$$q\ \pi_i = P[q_1 = S_i], \qquad 1 \le i \le N \tag{1.7}$$

Turint tinkamas N, M, A, B ir π reikšmes, PMM panaudojamas generuoti stebėjimų seką (1.1), kurioje kiekvienas stebėjimas O_t yra vienas iš V simbolių, o T yra stebėjimų skaičius sekoje. Tokiu būdu:

1. Pagal pradinį būsenų pasiskirstymą π pasirenkama pradinė būsena $q_1 = S_i$.
2. Nustatoma $t = 1$.
3. Pagal simbolių tikimybės pasiskirstymą būsenoje S_i, tai yra $b_i(k)$, pasirenkama $O_t = v_k$.

4. Pagal būsenų perėjimo tikimybės pasiskirstymą S_i, tai yra a_{ij}, pereinama į naują būseną $q_{t+1} = S_j$.
5. Nustatoma $t = t + 1$, jeigu $t < T$, grįžtama į 3 žingsnį. Kitu atveju procedūra baigiama.

Tokia procedūra gali būti naudojama tiek stebėjimų generavimui, tiek modeliavimui, kaip duota seka buvo sugeneruota tinkamu PMM [7].

Taigi PMM reikia dviejų modelio parametrų N ir M, stebimų simbolių bei trijų tikimybinių dydžių A, B ir π. Modelį galima aprašyti taip:

$$\lambda = (A, B, \pi) \tag{1.8}$$

1.3.1. Baum-Welsh algoritmas

Tam kad nustatyti PMM parametrus, pirmiausia būtina atlikti apytikslį spėjimą kokie jie galėtų būti. Kai tai padaroma, panaudojant *Baum-Welsh* algoritmą randami tikslesni parametrai. Pagrindinė problema yra apskaičiuoti PMM reikšmes. Jos aprašomos taip:

$$b_j(O_t) = \frac{1}{\sqrt{(2\pi)^n \left|\sum j\right|}} e^{-\frac{1}{2}(O_t-\mu_j)\sum_j^{-1}(O_t-\mu_j)} \tag{1.9}$$

Jeigu yra tik viena PMM būsena j, šis parametrų skaičiavimas yra paprastas. Tikimybės maksimumas apskaičiuojamas iš μ_j ir Σ_j kaip vidurkis:

$$\hat{\mu}_j = \frac{1}{T}\sum_{t=1}^{T} O_t \tag{1.10}$$

$$\hat{\Sigma}_j = \frac{1}{T}\sum_{t=1}^{T}(O_t - \mu_j)(O_t - \mu_j) \tag{1.11}$$

Bet praktikoje yra daug būsenų ir nėra tiesioginio stebėjimo būsenų pavertimo į atskiras būsenas, nes pagrindinė būsenos seka yra nežinoma. Kad apskaičiuoti, pirmiausia apmokomos stebėjimo būsenos lygiai padalinamos iš modelio būsenų ir tada paskaičiuojamos kiekvienos būsenos pradinės reikšmės (1.10) (1.11). Tada randamas būsenų sekos tikimybės maksimumas pasinaudojant *Viterbi* algoritmu ir surenkamos stebėjimo būsenos. Vėl perskaičiuojamos tikslesnės pradinės reikšmės (1.10) (1.11). Šie veiksmai kartojami iki kol atsakymas nebesikeičia.

Kol kiekvienos stebėjimo sekos pilna tikimybė yra pagrįsta visų įmanomų būsenų sekų suma, kiekviena stebėjimo būsena O_t įneša į maksimumo tikimybės skaičiavimą parametrų

reikšmes kiekvienai j reikšmei. Todėl jei $L_j(t)$ reiškia būsenos j tikimybę t laiku, tada galima suskaičiuoti svorinius vidurkius:

$$\hat{\mu}_j = \frac{\Sigma_{t=1}^{T} L_j(t) O_t}{\Sigma_{t=1}^{T} L_j(t)} \tag{1.12}$$

$$\hat{\Sigma}_j = \frac{\Sigma_{t=1}^{T} L_j(t)(O_t - \mu_j)(O_t - \mu)}{\Sigma_{t=1}^{T} L_j(t)} \tag{1.13}$$

Tai *Baum-Welsh* formulė apskaičiuoti PMM vidurkius ir kovariaciją.

Baum-Welsh algoritmas susideda iš šių dalių:

1. Kiekvienam parametrui suskaičiuojami svoriniai vidurkiai.
2. Suskaičiuojamos sklidimo pirmyn-atgal tikimybės kiekvienai j būsenai t laiku.
3. Kiekvienai j būsenai t laiku perskaičiuojama suma, panaudojant tikimybę $L_j(t)$ ir stebimą būseną O_t.
4. Panaudojamos galutinės susumuotos reikšmės apskaičiuoti naujas parametro vertes.
5. Jei $P = P(O|M)$ reikšmė mažesnė nei prieš tai buvusi, tada skaičiavimai baigiami. Jei didesnė – tada skaičiavimai pakartojami ir gaunamos naujos parametrų reikšmės [2].

1.3.2. Viterbi algoritmas

Norint rasti vienintelę geriausią būsenų seką, tai yra maksimizuoti $P(Q|O, \lambda)$, reikia naudoti *Viterbi* algoritmą. Jis yra pagrįstas dinaminiais programavimo metodais.

Norint rasti vienintelę ir geriausią būsenų seką $Q = q_1 q_2 ... q_T$, kai duota stebėjimų seka $O = O_1 O_2 ... O_T$, reikia aprašyti dydį:

$$\delta_t(i) = \max_{q_1, q_2, ..., q_{t-1}} P[q_1 q_2 ... q_t = i, O_1 O_2 ... O_t \mid \lambda]; \tag{1.14}$$

tai yra didžiausią taškų skaičių išilgai vienos būsenų sekos laiko momentu *t*, kuri įvertina pirmus *t* stebėjimus ir baigiasi būsenoje S_i. Tada gauname:

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij}\right] \cdot b_j(O_{t+1}) \tag{1.15}$$

Tam, kad būtų galima atkurti būsenų seką, reikia sekti argumentą, kuris maksimizuoja lygtį prie visų *t* ir *j* reikšmių. Tai yra atliekama naudojantis masyvu $\psi_t(j)$ [8].

Pirmiausia priskiriamos pradinės reikšmės:

$$\delta_1(i) = \pi_i b_i(O_1), \qquad 1 \le \mathrm{i} \le \mathrm{N} \tag{1.16}$$
$$\psi_1(i) = 0$$

Po to atliekama rekursija:

$$\delta_t(j) = \max_{1 \le i \le N}\left[\delta_{t-1}(i) a_{ij}\right] b_j(O_t), \qquad 2 \le t \le T \tag{1.17}$$
$$1 \le j \le N$$

$$\psi_1(i) = \arg\max_{1 \le i \le N}\left[\delta_{t-1}(i) a_{ij}\right] \qquad 2 \le t \le T \tag{1.18}$$
$$1 \le j \le N$$

Kadangi taip skaičiuojant tikimybės artėja prie nulio, suskaičiuojamos logaritminės tikimybės [2]:

$$\psi_j(t) = \max_i\left\{\psi_i(t-1) + \log\left(a_{ij}\right)\right\} + \log\left(b_j(O_t)\right) \tag{1.19}$$

Surandama labiausiai tikėtina būsena q_t laiko momentu *t*:

$$P = \max_{1 \le i \le N}\left[\delta_T(i)\right] \tag{1.20}$$

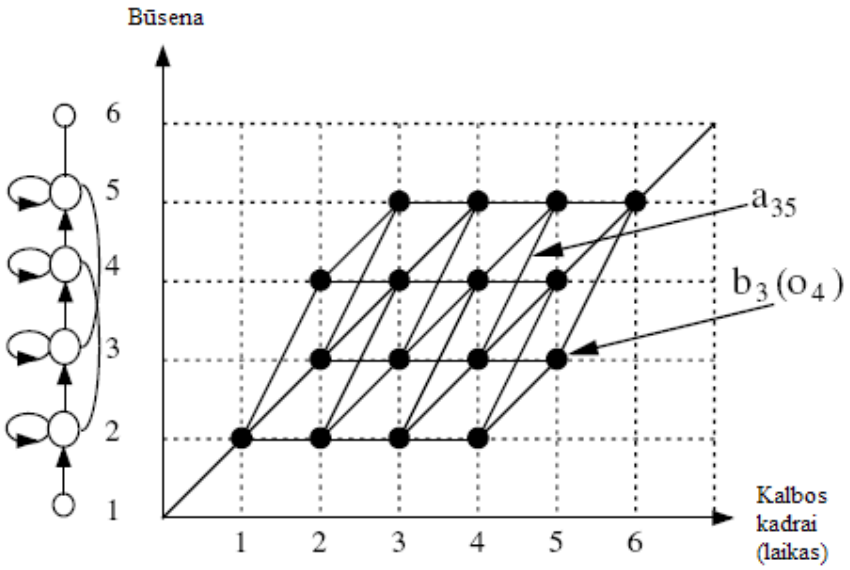
$$q_T = \arg\max_{1 \le i \le N}\left[\delta_T(i)\right] \tag{1.21}$$

Atkuriama būsenų seka [6]:

$$q_t = \psi_{t+1}(q_{t+1}), \qquad t = T-1, T-2, \ldots, 1 \tag{1.22}$$

Viterbi algoritmas gali būti pavaizduotas (1.2 pav.), kaip geriausias kelias per matricą, kur vertikali ašis yra PMM būsenos, o horizontali ašis yra kalbos kadrai, tai yra laiko ašis. Kiekvienas didelis taškas tai yra stebima logaritminė tikimybė tam tikru laiko momentu, o kiekviena linija tarp tašku vaizduoja logaritminę pereinamąją tikimybę. Bet kurio kelio logaritminė

tikimybė suskaičiuojama susumuojant pereinamąsias tikimybes ir to kelio išėjimo tikimybes. Keliai didėja stulpelis po stulpelio, iš kairės į dešinę [2].



1.2 pav. Viterbi algoritmas panaudotas atskirai tariamų žodžių atpažinime

1.3.3. Sklidimo pirmyn-atgal algoritmas

Tiesiogiai skaičiuojant tikimybę $P(O|\lambda)$ reikia atlikti $2T \cdot N^T$ operacijų, nes kiekvienu laiko momentu $t = 1, 2, ..., T$ galima pereiti į N galimų būsenų ir kiekvienai tokiai būsenai reikia $2T$ skaičiavimų. Toks skaičiavimas yra labai neefektyvus net ir mažoms N ir T reikšmėms. Todėl naudojamas sklidimo pirmyn-atgal algoritmas.

Sklidimo pirmyn kintamasis $\alpha_t(i)$ aprašomas taip:

$$\alpha_t(i) = P(O_1 O_2 O_3 ... O_t, q_t = S_i \mid \lambda) \tag{1.23}$$

Tai tikimybė dalinės stebėjimo sekos $O_1O_2 ... O_t$ iki laiko momento t ir būsenos S_t laiko momentu t, kai duotas modelis λ.

Skaičiuojant tikimybę pirmiausia priskiriamos pradinės reikšmės sklidimo pirmyn tikimybės, kaip jungtinę tikimybę būsenos S_i ir pradinio stebėjimo O_1.

$$\alpha_t(i) = \pi_i b_i(O_1), \qquad 1 \le i \le N \tag{1.24}$$

Laiko momentu $t+1$ iš N galimų būsenų galima patekti į būseną S_j. Taip kaip $\alpha_t(i)$ yra tikimybė jungtinio įvykio, kad seka $O_1O_2 \ldots O_t$ yra stebima ir būsena laiko momentu t yra S_i, tada sandauga $\alpha_t(i)a_{ij}$ yra tikimybė jungtinio įvykio, kad seka $O_1O_2 \ldots O_t$ yra stebima ir į būseną S_j laiko momentu $t+1$ patenkama iš būsenos S_i laiko momentu t. Sumuojant šią sandaugą per visas N įmanomas būsenas S_i, $1 \le i \le N$, laiko momentu t, gauname tikimybę būsenos S_j, laiko momentu $t+1$, su visais ankstesniais daliniais stebėjimais. Kai S_j yra žinoma, tada $\alpha_{t+1}(j)$ yra gaunama ankstesnės sumos rezultatą dauginant iš tikimybės $bj(O_{t+1})$.

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \le t \le T-1, \quad 1 \le j \le N \tag{1.25}$$

Ši lygtis skaičiuojama visoms S_j, $1 \le j \le N$, būsenoms duotu laiko momentu t. Po to skaičiavimai kartojami visiems $t = 1, 2, \ldots, T-1$.

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{1.26}$$

Gaunama norima $P(O|\lambda)$ sumuojant sklidimo pirmyn kintamuosius $\alpha_T(i)$. Tai gaunama iš lygties:

$$\alpha_T(i) = P(O_1O_2 \ldots O_T, q_T = S_i \mid \lambda) \tag{1.27}$$

Todėl $P(O|\lambda)$ yra tiesiog suma visų $\alpha_T(i)$.

$\alpha_T(i)$, kai $1 \le t \le T$ *ir* $1 \le j \le N$, paskaičiavimui reikia gerokai mažiau, apie N^2T operacijų, vietoj $2TN^T$, kurių reikalauja tiesioginis šios tikimybės radimas.

Tokiu pačiu būdu aprašome sklidimo atgal kintamąjį $\beta_t(i)$:

$$\beta_t(i) = P(O_1O_2O_3 \ldots O_t, q_t = S_i \mid \lambda) \tag{1.28}$$

Tikimybė dalinės stebėjimo sekos nuo laiko momento $t + 1$ iki galo duotai būsenai S_i laiko momentu t ir modeliui λ.

Priskiriamos pradinės reikšmės $\beta_t(i)$ prilyginant 1 visiems i:

$$\beta_t(i) = 1, \qquad 1 \le i \le N \tag{1.29}$$

Būnant būsenoje S_i laiko momentu t ir norint įvertinti stebėjimo seką laiko momentu $t + 1$, reikia visoms galimoms būsenoms S_j laiko momentu $t + 1$ įvertinti perėjimą iš S_i į S_j, stebėjimą O_{t+1} būsenoje j bei po to įvertinti likusią dalinę stebėjimų seką iš būsenos j.

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots, 1 \qquad (1.30)$$

$$1 \le i \le N$$

$\beta_t(i)$, kai $1 \le t \le T$ ir $1 \le i \le N$, paskaičiavimui taip pat reikia apie N^2T operacijų [5].

Bendra tikimybė $P = prob(O|\lambda)$ gali būti paskaičiuojama arba iš sklidimo pirmyn arba iš sklidimo atgal tikimybės [2]:

$$P = \alpha_N(T) = \beta_1(1) \qquad (1.31)$$

1.4. Skaičiavimų mažinimas ir tikslumo didinimas

Jei kalbos atpažinimo sistema atpažino ištartas fonemas, atpažinti ištartus žodžius nebeturėtu būti sunku. Sistema turėtu tik palyginti visas gautas fonemas su tarimo žodynu. Bet gali iškilti keletas problemų:

1. Žmogus gali ištarti žodi taip, kad tokio tarimo nebus žodyne.
2. Pati sistema gali klaidingai suprasti, kokią nors žodyje esančią fonemą.
3. Sistema gali neatskirti kada vienas žodis baigiasi, o kada prasideda.
4. Kalbos atpažinimas gali pareikalauti per daug procesoriaus išteklių.

Kad sumažinti skaičiavimus ir padidinti tikslumą, galima apriboti žmogaus pateiktus duomenis. Tai gali būti naudinga nes:

1. Žmogus gali pasakyti beprasmį žodį.
2. Žmogus paprastai naudoja santykinai mažą žodyną. Yra milijonai žodžių, bet dauguma žmonių kasdien naudoja tik keletą tūkstančių, o bendravimui su kompiuterių galbūt reikia dar mažiau žodžių.
3. Kalbėdami žmonės naudoją gramatiką. Vieni žodžiai eina po tam tikrų kitų žodžių.
4. Kai kurie žodžiai vartojami daug dažniau nei kiti.

Vienas iš būdų, skaičiavimams sumažinti ir tikslumą padidinti, yra žodžių skirstymas į kategorijas. Šis būdas veikia ribodamas kalbos atpažinimo žodyną ir sintaksės struktūrą, paliekant tik tuos žodžius ir sakinius, kurie yra priimtini esamai situacijai. Sistema tekstiniame faile tiksliai

nusako žodyną ir sintaksės struktūra. Pavyzdžiui: *<Pradėti> = (Skambinti | Siųsti laišką) (Jonui | Petrui | Antanui) | (Išjungti)*. Šią gramatiką galima suskaidyti į septynis galimus sakinius: *Skambinti Jonui, Skambinti Petrui, Skambinti Antanui, Siųsti laišką Jonui, Siųsti laišką Petrui, Siųsti laišką Antanui, Išjungti*. Šis kategorijų aprašymas gali būti ir sudėtingesnis. Kategorijų sudarymo svarbi savybė yra ta, kad ji riboja tai ką atpažinimo sistema tikisi išgirsti iki mažo žodyno ir siauros sintaksės. Kai žmogus pasako žodį *„Siųsti“* atpažinimo sistema laukia tik sekančio žodžio *„laišką“*. Tai žymiai sumažina generuojamų spėjimų kiekį [9].

Kalbos atpažinimo sistema gauna kiekvieno žodžio fonemas, ieškodama žodžio žodyne. Jeigu žodyne nėra tokio žodžio, tada sistema spėja tarimą. Nes kai kurie žodžiai gali būti ištarti keletu skirtingu variantu. Atpažinimo sistema paprastai vieną žodį, ištartą keliais skirtingais tarimais, traktuoja kaip skirtingus žodžius.

Žodžių suskirstymas į kategorijas šiek tiek keičia kalbos atpažinimo sistemos spėjimus. Vietoj to kad spėliotų visas fonemas, sistema spėja tik sekančią, pagal kategorijas tinkančią fonemą. Tokiu būdu ženkliai sumažinami skaičiavimai. Spėjimo variantų skaičius nedidėja ir išlieka stabilus per visą žodį. Panaudojant atmetimus paprastai visą laiką išlieką apie 10 spėjamų variantų. Kai žmogus baigia kalbėti sistema grąžina spėjimą su didžiausiu balu.

Naudojant žodžių skirstymą į kategorijas realiu laiku taip pat nereikia ir didelių procesoriaus išteklių [4].

1.5. Akustiniai kalbos atpažinimo metodai

Akustiniuose metoduose kompiuteris bando nuosekliai dekoduoti kalbos signalą remdamasis stebimais signalo akustiniais požymiais ir žinomais ryšiais tarp akustinių požymių bei fonetinių simbolių. Šis metodas buvo tyrinėjamas virš 40 metų. Nepaisant to, jis nepasiekė tokio sėkmingo pritaikymo kaip alternatyvūs metodai.

Akustiniai metodai remiasi akustinės fonetikos teorija, kuri sako, kad šnekamojoje kalboje egzistuoja baigtiniai, būdingi fonetiniai vienetai ir kad šie fonetiniai vienetai yra plačiai apibūdinami eile savybių, kurios aiškiai atsispindi kalbos signalo kitimo laike. Nors fonetinių vienetų akustinės savybės smarkiai priklauso nuo diktoriaus ir nuo kaimyninių fonetinių vienetų yra priimama, kad taisyklės, valdančios kitimą, gali būti atskleistos ir panaudotos praktinėse situacijose.

Pirmas žingsnis akustiniuose metoduose yra kalbos signalo segmentavimas ir žymėjimas. Jo metu kalbos signalas yra suskirstomas į diskretines laiko sritis, kuriose signalo akustinės savybės yra vieno ar kelių fonetinių vienetų atstovai. Po to pagal akustines savybes kiekvienam segmentui yra priskiriamos fonetinės žymės.

Antrame žingsnyje pagal sukurtą pirmame žingsnyje fonetinių žymių seką bandoma nustatyti teisingą žodį arba žodžių seką, kuri tenkina konkretaus kalbos atpažinimo tikslo apdorojimus.

Akustiniai kalbos atpažinimo metodai reikalauja gilaus fonetinių vienetų akustinių savybių žinojimo. Juose naudojami požymiai dažniausiai yra parenkami remiantis vien intuicija, sprendimo priėmimo taisyklės taip pat yra neoptimalios. Plačiausiai šiuose metoduose naudojami požymiai yra: formantės, pagrindinis tonas, energija, nosinumas, frikatyvumas, vokalizuotas/nevokalizuotas.

Požymiai yra išskiriami naudojant juostinių filtrų blokus arba tiesinės prognozės kodavimo (LPC) metodus [10].

Pirmame žingsnyje yra atliekama kalbos analizė, kuri atlieka laike kintančio kalbos signalo spektrinį pavaizdavimą.

Antrame žingsnyje atliekamas požymių išskyrimas. Kai kurie požymiai yra binariniai – nosinumas, frikatyvumas, vokalizuotas/nevokalizuotas, kitų yra tolydiniai – formančių padėtys, energijų santykiai. Šiame etape naudojama eilė lygiagrečiai veikiančių detektorių, kurie atlieka apdorojimą ir pagal tam tikras logines taisykles priima sprendimą apie minėtus požymius.

Trečiame etape atliekamas segmentavimas ir žymėjimas, kur sistema bando rasti stabilias sritis, kur požymiai kinta labai nežymiai. Tada segmentuotos sritys pažymimos, pagal tai kaip požymiai jose atitinka atskirus fonetinius vienetus. Tai yra centrinė ir pati sudėtingiausia akustinio – fonetinio metodo dalis. Segmentavimo ir žymėjimo rezultatas yra fonemų tinklelis. Pagal tą tinklelį žodyno apdorojimo procedūros nustato labiausiai atitinkantį žodį arba žodžių seką. Į procedūras gali būti įtraukti žodyno ir sintaksės apribojimai. Požymių atitikimo kokybė fonetiniams vienetams segmente gali būti naudojama priskirti žymėms tikimybes, kurios gali būti panaudotos tikimybinėse žodyno apdorojimo procedūrose. Galutinis kalbos atpažinimo sistemos išėjimas yra žodis arba žodžių seka, kuri geriausiai atitinka fonemų tinklelio fonetinių vienetų seką.

1.6. Dirbtinio intelekto metodai

Neuronų tinklas yra galingas modeliavimo aparatas, įgalinantis modeliuoti ypač sudėtingas funkcijas. Bendru atveju neuroninis tinklas yra netiesinė struktūra. Ilgą laiką buvo naudojami tiesiniai modeliavimo metodai, turintys gerai žinomas optimizavimo strategijas. Tais atvejais, kada tiesinės aproksimacijos nepakanka, tiesiniai modeliai nėra priimtini. Neuroniniai tinklai padeda išvengti dimensiškumo problemos, iškylančios modeliuojant daugelio kintamųjų funkcijas tiesiniais metodais [11].

Neuronų tinklai apmokomi iš pavyzdžių. Turint duomenų pavyzdžius ir naudojant mokymo algoritmus, neuroninis tinklas pritaikomas prie duomenų struktūros.

Neuronas yra pagrindinis dirbtinių neuroninių tinklų skaičiavimo elementas. Jis gauna keletą įėjimo reikšmių. Tai neuroninio tinklo įėjimo reikšmės arba kitų tinklo neuronų išėjimo reikšmės. Kiekviena įėjimo jungtis turi savo perdavimo koeficientą – svorį. Šie svoriai atitinka biologinio neurono sinapsių efektyvumą. Kiekvienas neuronas turi savo slenksčio reikšmę. Neurono sužadinimo reikšmė formuojama skaičiuojant svorinę įėjimo signalų sumą ir atimant slenksčio reikšmę. Pagal sužadinimo signalą naudojant neurono perdavimo funkciją skaičiuojama neurono išėjimo reikšmė.

Atskiri neuronai įvairiausiais būdais gali būti jungiami į neuroninį tinklą. Pagal neuronų tarpusavio sujungimo būdą ir pačių neuronų savybes išskiriami įvairūs neuronų tinklų tipai. Vienas iš pagrindinių neuronų tinklų tipų yra daugiasluoksnis perceptronas. Tai neuroninis tinklas, kurio neuronai yra grupuojami į atskirus sluoksnius. Įėjimo sluoksnis nėra neuronų sluoksnis, jis tik įveda į tinklą įėjimo kintamųjų reikšmes. Paslėptų ir išėjimo sluoksnių neuronai sujungti su visais prieš tai esančio sluoksnio elementais. Vieno sluoksnio neuronai dažniausiai turi tą pačią perdavimo funkciją, kuri netiesinė ir tolydi. Daugiasluoksnis perceptronas turi tiesioginio sklidimo tinklo struktūrą. Signalai sklinda iš įėjimų pirmyn per visus paslėptus elementus ir pasiekia išėjimo neuronus. Kai įėjimo reikšmės patenka į įėjimo sluoksnį, paeiliui skaičiuojamos paslėptų sluoksnių neuronų išėjimo reikšmės, po to išėjimo sluoksnio neuronų išėjimo reikšmės. Kiekvienas neuronas skaičiuoja svorinę prieš tai esančio sluoksnio neuronų išėjimų sumą ir pagal gautąją sužadinimo reikšmę skaičiuojama neurono perdavimo funkcijos reikšmė. Išėjimo sluoksnio neuronų išėjimo reikšmės laikomos neuroninio tinklo išėjimo reikšmėmis.

Tam, kad neuroninis tinklas galėtų atlikti požymių vektorių klasifikavimą, prieš tai reikia jį tinkamai apmokyti. Neuroninis tinklas apmokomas naudojant mokymo imtį. Mokyme su mokytoju, mokymo imtis sudaroma iš požymių vektorių kartu su atitinkamais ieškomų išėjimų vektoriais. Ieškomo išėjimo vektorius nurodo požymių vektoriaus atitikimą vienai iš apibrėžtų klasių. Mokymo metu tinklas mokosi priklausomybės tarp požymių ir atitinkamų ieškomų išėjimo vektorių. Jei tinklas tinkamai apmokytas, jis gali modeliuoti funkciją siejančią požymio vektorius su atitinkamais trokštamų išėjimų vektoriais ir tokiu būdu gali atlikti naujų požymių vektorių klasifikavimą.

Mokymo proceso metu dažniausiai skaičiuojamas neuroninio tinklo darbą įvertinantis parametras. Tai gali būti neuroninio tinklo daroma klaida visiems duomenims. Toliau pagal mokymo taisyklę koreguojami tinklo svoriai taip, kad neuroninis tinklas darytų kiek įmanoma mažesnę klaidą ir tinklo išėjimo reikšmės būtų, kaip galima artimesnės ieškomų išėjimų reikšmėms.

Daugiasluoksniams perceptronams egzistuoja efektyvus tikslumo atžvilgiu, mokymo algoritmas vadinamas klaidos atbulinio sklidimo algoritmu. Šis algoritmas grindžiamas gradientinio nusileidimo metodu [12].

1.7. Kalbos atpažinimo tyrimų analizė

2006 m. Kauno technologijos universiteto Kalbos signalų tyrimo laboratorijoje paruoštos dvi telefoninio kalbinio dialogo demonstracijos su *Microsoft* balso serveriu ir keturių kanalų *Intel Dialogic D41JCT* telefonine plokšte, taip pat įsisavinta telefoninio kalbinio dialogo, skirto *IBM* balso serveriui, paruošimo metodika.

Balso programų aptarnavimo serveriai, tokie, kaip *Microsoft Speech Server* ar *IBM WebShere Voice Server* pateikia automatinio kalbos atpažinimo ir šnekamosios kalbos generavimo resursus. Šie balso serveriai šiuo metu yra populiariausi pasaulyje.

Microsoft balso serveris buvo sukurtas siekiant eliminuoti sudėtingą specializuotą aparatūrinę ir programinę įrangą, supaprastinti kalbos integraciją, naująją technologiją padarant labiau patrauklia vidutinio dydžio įmonėms ir plataus vartojimo programų kūrėjams, sukurti programų šablonus telefonams ir mobiliems įtaisams, sukurti standartą ir specifikacijas, aprašančius kalbos elementus internetinėse, tradicinėse ir naujose multimodalinėse programose. *Microsoft* balso serverio pakete pateikiama visa reikalinga programinė įranga kalbos programų kūrimui, pritaikymui ir naudojimui.

IBM savo programinės įrangos pakete pateikia išbaigtus sprendimus, kaip pateikti informaciją vartotojui greičiau ir paprasčiau. Pakete pateikiama kalbos atpažinimo programinė įranga, kuri gali klausytis žodžių, šnekamų per telefoną, atpažinti tuos žodžius ir perduoti juos kaip tekstą kitai programinei įrangai.

Microsoft ir *IBM* balso serveriai neturi lietuviškų balso komandų atpažinimo galimybės. Naudojant lietuviškus žodžius, kalbos atpažinimo rezultatai gauti visiškai prasti – dauguma žodžių neatpažįstama arba atpažįstama klaidingai. Lietuviškos kalbos atpažinimo pagerinimui panaudotos angliškos lietuviškų žodžių transkripcijos. Taip anglų kalbos atpažinimo modulis interpretuoja išgirstą žodį kaip anglišką ir kai kuriais atvejais gauti geri rezultatai [13].

Vytauto Didžiojo universitete atliekami lietuvių kalbos atpažinimo tyrimai, taikant paslėptųjų Markovo modelių metodiką. Tyrimams naudojamas VDU trumpų rišlių frazių lietuvių kalbos garsynas. Garsynas apima keturių kalbėtojų šnekos įrašus. Kalbėtojai yra pateikę po 273 įrašus, kurių kiekvieną sudaro 1-4 rišlios frazės sudarytos iš 3-4 žodžių. Garsyno, naudojamo eksperimentuose, apimtis – 107,1 minutės šnekos įrašų. Sprendžiant 1550 žodžių apimties nuo

kalbėtojo nepriklausomą trumpų rišlių lietuvių kalbos frazių atpažinimo uždavinį, pasiekta 6-21 % santykinė žodžių atpažinimo klaida [14].

Vytauto Didžiojo universitete atliekant lietuvių kalbos atpažinimo eksperimentą, pagrįstą fonemų perėjimo momentų akustiniais modeliais, ištirta atpažinimo santykinės klaidos priklausomybė nuo melo dažnių kepstro koeficientų skaičiaus, lango dydžio ir naudingo dažnio diapazono. Gauta atpažinimo santykinė paklaida vidutiniškai svyruoja nuo 28 % iki 36 % [15].

Matematikos ir informatikos institute atliekami atskirai pasakytų lietuvių kalbos žodžių atpažinimo tyrimai naudojant neuroninius tinklus. Modeliuojant dešimties atskirai pasakytų žodžių atpažinimo sistemą, panaudotas vieno paslėpto sluoksnio daugiasluoksnis perceptronas, apmokomas pagal klaidos atbulinio sklidimo algoritmą. Atliekant eksperimentus sumodeliuota sistema buvo gautas didžiausias 87 % žodžių atpažinimo tikslumas [12].

2. LIETUVIŲ KALBOS ATSKIRAI TARIAMŲ ŽODŽIŲ ATPAŽINIMAS

Lietuvių kalbos atskirai tariamų žodžių atpažinimui reikia:

1. Akustinio modelio, kuriame būtų statistiškai aprašyti skirtingi garsai, kuriuos sukelia kiekvienas tariamas žodis.
2. Žodžių kategorijų, kuriose būtų aprašyti žodžių kombinacijų rinkiniai.
3. Kalbos atpažinimo variklio, kuris žmogaus ištartą garsą susietu su akustiniame modelyje esančiais garsais, o po to rastų tokią fonemų seką žodžių kategorijose.

Akustinis modelis sukurtas naudojant *Hidden Markov Toolkit (HTK)* įrankius, kurie skirti kurti ir manipuliuoti statistinius modelius. Po to šie modeliai kalbos atpažinime atspindi garsus. Žodžių kategorijos aprašytos *Backus-Naur* formatu. Eksperimentai atlikti ir rezultatai gauti naudojant *Julius* programinės įrangos įrankius.

2.1. Akustinis modelis

Kuriant lietuvių kalbos akustinį modelį reikia sudaryti lietuvių kabos tarimo žodyną bei fonemų transkripcijas, įrašyti garsyną ir konvertuoti jį į formatą, tinkamą išskirti požymio vektorius. Taip pat pagal išskirtus požymio vektorius reikia apmokyti paslėptuosius Markovo modelius.

2.1.1. Tarimo žodynas

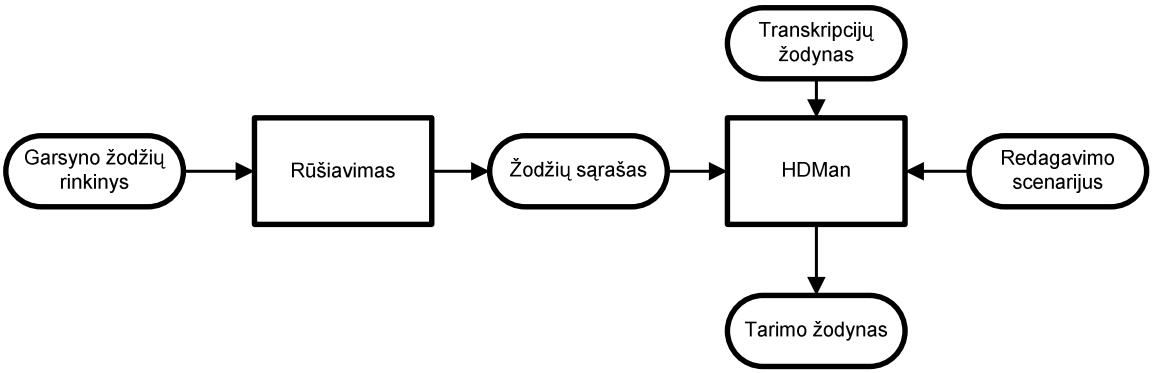
Akustiniam modeliui sukurti, sudarytas lietuvių kalbos garsynas iš 1410 atskiriamai tariamų žodžių. Žodžiai suskirstyti į nedideles grupeles ir įrašyti. Įrašai atlikti su *Audacity* programa, naudojant *Mono* kanalą, nustatant 48000 Hz dažnį ir 16 bitų formatą. Jie eksportuoti į nesuspaustą *WAV* 16 bitų *PCM* formatą.

Ištrauka iš sudaryto garsyno:

**/sample1 ABEJONĖ ABEJOTI ABIPUS ABIPUSIS ABITŪRA ABRIKOSAS ABRIKOSMEDIS ABU ABUDU ABĖCĖLINIS*
**/sample2 ABĖCĖLĖ ACETATAS ACETONAS ACTAS ADATA ADATINĖ ADIKLIS ADMINISTRACIJA ADMINISTRACINIS ADMINISTRATORIUS*
**/sample3 ADRESAS ADRESATAS ADVOKATAS ADYTI AFERA AFERISTAS AFIŠA AFORIZMAS AGENTŪRA AGITACIJA*
**/sample4 AGITACINIS AGITATORIUS AGRASTAS AGRESIJA AGRESORIUS AGRESYVUS AGRONOMAS AJERAS AKACIJA AKADEMIJA*

**/sample5 AKADEMIKAS AKADEMINIS AKCINIS AKIMIRKA AKIMIRKSNIS AKIMOJIS AKIS AKMENINIS AKORDAS AKTORIUS*
**/sample6 AKTYVUS AKUSTIKA AKYLAS AKĖTI AKĖČIOS ALAVAS ALBANAS ALBUMAS ALERGIJA ALFABETAS*
**/sample7 ALIO ALIUMINIS ALKŪNĖ ALPINISTAS ALUDĖ ALYVA ALĖJA AMATAS AMBICIJA AMFIBIJA*
**/sample8 AMPERAS AMPERMETRAS AMPLITUDĖ AMPUTACIJA AMULETAS AMUNICIJA ANALIZĖ ANANASAS ANGA ANGINA*
**/sample9 ANGIS ANGLIAVANDENIS ANGLIS ANKETA ANKSTI ANKSTOKAS ANKSTYVAS ANODAS ANOKS ANOMALIJA*
**/sample10 ANONIMAS ANONSAS ANOT ANOTACIJA ANTENA ANTRADIENIS ANTROKAS ANTROPOLOGAS ANYTA ANŪKAS*
...

Prieš kiekvieną žodžių grupę yra parašytas garso failo vardas, į kurį įrašyta atitinkama žodžių grupė. Kaip pavaizduota 2.1 paveiksle, kuriant tarimo žodyną, su *HTK* scenarijumi *prompts2wlist* atliekamas rūšiavimas, panaikinant garso failų vardus ir kiekvieną unikalų žodį atspausdinant atskiroje eilutėje, kitame faile. Į ši failą įterpiami du papildomi žodžiai *SENT-END* ir *SENT-START*. Jie žymi žodžio pradžioje ir pabaigoje esančią tylą.



2.1 pav. Tarimo žodyno sudarymo schema

Garsyne panaudotiems žodžiams reikia sudaryti transkripcijų žodyną. Jame turi būti visi žodžiai naudojami kalbos atpažinime. Sudarant transkripcijų žodyną atsižvelgta į kirčiavimą, priebalsių minkštumą ir kietumą, balsių ilgumą ir trumpumą [16]. Lietuviškų raidžių tarimas žymimas taip: *ė – ee*, *š – ss*, *č – cc*, *ž – zz*. Prie minkšto priebalsio pridedama raidė *m*, o prie kieto priebalsio nededamas papildomas žymėjimas. Prie ilgo balsio pridedama raidė *l*, o prie trumpo balsio nededamas papildomas žymėjimas. Kairinis kirtis žymimas raide *k*, dešininis kirtis žymimas raide *d*, o riestinis kirtis žymimas raide *r*. Iš viso panaudotos 87 fonemos.

Ištrauka iš transkripcijų žodyno:

ABEJONĖ [ABEJONĖ] a bm e j olr nm eel
ABEJOTI [ABEJOTI] a bm e j old tm i
ABIPUS [ABIPUS] a bm ik p u s
ABIPUSIS [ABIPUSIS] a bm i p uk sm i s
ABITŪRA [ABITŪRA] a bm i t ul r ak

ABRIKOSAS [ABRIKOSAS] a bm rm i k ok s a s
ABRIKOSMEDIS [ABRIKOSMEDIS] a bm rm i k ok sm mm e dm i s
ABU [ABU] a b uk
ABUDU [ABUDU] a b uk d u
ABĖCĖLINIS [ABĖCĖLINIS] a bm eel cm eelr lm i nm i s
ABĖCĖLĖ [ABĖCĖLĖ] a bm eel cm eelr lm eel
ACETATAS [ACETATAS] a cm e t alr t a s
ACETONAS [ACETONAS] a cm e t ok n a s
ACTAS [ACTAS] alr c t a s
ADATA [ADATA] alr d a t a
ADATINĖ [ADATINĖ] a d a tm ik nm eel
ADIKLIS [ADIKLIS] a dm ik k lm i s
ADMINISTRACIJA [ADMINISTRACIJA] a dm mm i nm i s t r alr cm i j e
ADMINISTRACINIS [ADMINISTRACINIS] a dm mm i nm i s t r alr cm i nm i s
ADMINISTRATORIUS [ADMINISTRATORIUS] a dm mm i nm i s t r alr t o rm u s

Tarimo žodynas sukurtas pasinaudojant *HTK* įrankiu *HDMan*. *HDMan* gali sukurti tarimo žodyną iš vieno ar daugiau šaltinių. Jis iš tam tikro scenarijaus nuskaito redagavimo komandas ir sukuria suredaguotą ir sujungtą kopiją iš vieno ar kelių žodynų. Kiekviena redagavimo komanda scenarijuje turi būti atskiroje eilutėje. Jei nėra redagavimo scenarijaus *HDMan* paprasčiausiai sujungia žodynus ir pateikia juos surūšiuotus pagal alfabetą. Kiekvienas žodynas gali turėti atskirą redagavimo scenarijų, kuriame būtų surašytos specialiai jam skirtos komandos. Naudojant vieną redagavimo scenarijų, komandas galima surašyti į *global.ded* failą. Žodynas gali būti išfiltruotas panaudojant žodžių sąrašą. Tuomet žodyno žodžiai, kurie nėra žodžių sąraše, yra ignoruojami. Iš žodyno šiems žodžiams paimamas tik tarimas [17].

Darbe panaudotas redagavimo scenarijus:

AS sp
MP sil sil sp

čia

AS – prie kiekvieno tarimo pridėti tylos modelį *sp*,

MP – sujungia bet kurias fonemų *sil* ir *sp* sekas ir pavadina jas *sil*.

Tarimo žodynas sukuriamas įvykdžius šią komandinę eilutę:

HDMan –A –D –T 1 –m –w wlist –n monophones1 –i –l dlog dict transkripcijos

čia

-A – atspausdina komandinės eilutės argumentus,

-D – parodo konfigūravimo kintamuosius,

-T – nustato žymėjimo vėliavėlę į 1, kuri parodo komandos vykdymo progresą,

-m – sujungia visų šaltinių tarimą,

-w – užkrauna žodžių sąrašą esantį *wlist* faile,

-n – išveda į failą *monophones1* skirtingas aptiktas fonemas,

-i – įtraukia žodžio išėjimo simbolius į išėjimo žodyną,

-l – sukuria žurnalą *dlog*, kuriame yra žodyno statistika ir kiekvienos fonemos aptikimo skaičius,

dict – kuriamo tarimo žodyno pavadinimas,

transkripcijos – transkripcijų žodynas.

Tarimo žodyne panaudotos fonemos ir jų pasikartojimų skaičius:

1. a : 1071
2. bm : 86
3. e : 287
4. j : 123
5. olr : 37
6. nm : 200
7. eel : 133
8. sp : 1315
9. old : 51
10. tm : 276
11. i : 641
12. ik : 188
13. p : 111
14. u : 116
15. s : 910
16. uk : 73
17. sm : 135
18. t : 303
19. ul : 36
20. r : 288
21. ak : 98
22. rm : 227
23. k : 372
24. ok : 70
25. mm : 92
26. dm : 172
27. b : 142
28. d : 146
29. cm : 42
30. eelr : 36
31. lm : 222
32. el : 7
33. alr : 212
34. n : 223
35. c : 6
36. o : 163
37. v : 75
38. ild : 67
39. fm : 26
40. ss : 58
41. f : 32
42. z : 23
43. m : 176
44. gm : 71
45. g : 138
46. ek : 95
47. il : 68

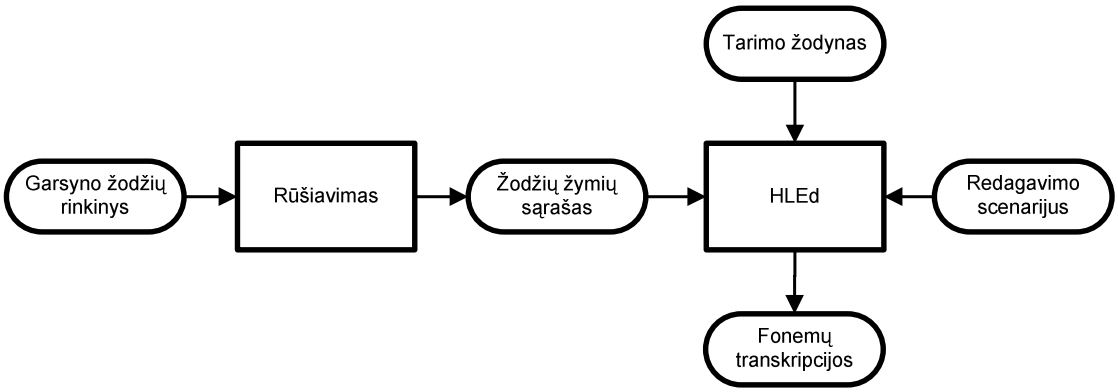
48. km : 81
49. ol : 38
50. ilr : 68
51. vm : 91
52. l : 173
53. eeld : 38
54. ccm : 23
55. uld : 18
56. pm : 102
57. uo : 17
58. zm : 31
59. nn : 52
60. nnm : 14
61. elr : 45
62. ie : 30
63. ssm : 65
64. udo : 32
65. ulr : 22
66. ier : 16
67. aldu : 20
68. ai : 35
69. x : 11
70. xm : 12
71. aulr : 13
72. eilr : 15
73. au : 38
74. zz : 30
75. ald : 26
76. zzm : 57
77. aelr : 6
78. ailr : 36
79. aeld : 7
80. dzzm : 8
81. ei : 27
82. hm : 9
83. aldi : 10
84. ide : 10
85. h : 7
86. uor : 5
87. sil : 2

Papildomai sukuriamas failas *monophones0*, išmetant iš *monophones1* pauzės fonemą *sp*.

2.1.2. Transkripcijų kūrimas

HTK paketas negali tiesiogiai apdoroti garsyno žodžių rinkinio. Todėl reikia sukurti *MLF* pagrindinį žymėjimo failą. Šiame faile garsyno žodžių rinkinys surūšiuojamas pašalinant garso įrašų failų pavadinimus ir pasikartojančius žodžius. Veiksmui atlikti reikia panaudoti *HTK* scenarijų *prompts2mlf*, kuris sugeneruoja *words.mlf* failą iš apmokymo sakinių.

Kaip pavaizduota 2.2 paveiksle, toliau su *HTK* įrankiu *HLEd* reikia iš žodžių lygio transkripcijų pereiti į fonemų lygio transkripcijas. Tai yra kiekvieną žodį pakeisti jo fonemomis. Tai daroma imant kiekvieną žodį iš *words.mlf* failo ir surandant to žodžio fonemas iš *dict* failo. Rezultatas išvedamas į *phones0.mlf* failą.



2.2 pav. Fonemų transkripcijų sudarymo schema

Į failą *mkphones0.led* surašomos *HLEd* įrankiui reikalingos redagavimo komandos:

EX
IS sil sil
DE sp

čia

EX – kiekvieną žodį esantį *words.mlf* pakeičia jį atitinkančiu tarimu;

IS – įterpia tylos fonemas *sil* sakinio pradžioje ir pabaigoje;

DE – ištrina pauzes *sp* tarp žodžių.

Fonemų transkripcijos su pašalintomis *sp* pauzėm gaunamos įvykdžius šią komandą:

HLEd –A –D –T 1 –l '' –d dict –i phones0.mlf mkphones0.led words.mlf*

čia

-*A* – atspausdina komandinės eilutės argumentus,

-*D* – parodo konfigūravimo kintamuosius,

-*T* – nustato žymėjimo vėliavėlę į 1, kuri parodo komandos vykdymo progresą,

-*l* – išveda kelią,

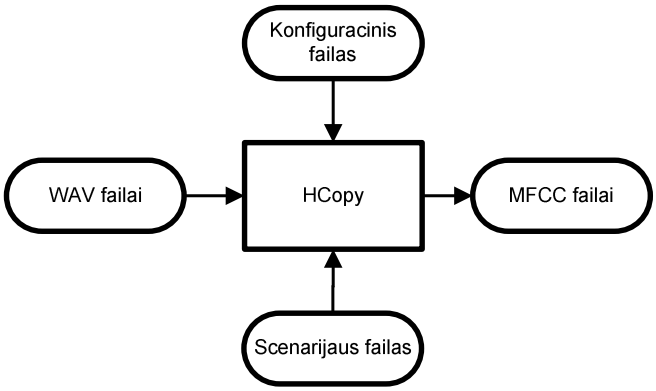
-*d* – užkrauną žodyną *dict* ir yra naudojamas, kai panaudojama redagavimo komanda *EX*

-*i* – išvedamos transkripcijos į *phones0.mlf* failą.

Taip pat sukuriamas fonemų transkripcijų failas *phones1.mlf*, kuriame tarp žodžių yra tylos pauzės *sp*. Tai atliekama iš redagavimo komandų scenarijaus pašalinus *DE sp* komanda ir vėl įvykdžius *HLEd* komanda, nurodant transkripcijas išvesti į *phones1.mlf* failą.

2.1.3. Kalbos įrašų kodavimas

Kad būtų galima lengviau išskirti požymio vektorius, *WAV* formato garso įrašų duomenis reikia konvertuoti į *MFCC* formatą. Kaip parodyta 2.3 paveiksle konvertavimas atliekamas su *HTK* įrankiu *HCopy*.



2.3 pav. Kalbos įrašų kodavimo schema

Pirmiausia faile *codetrain.scp* parašomas scenarijus, kuris nurodo kokius ir kur failus reikia konvertuoti. Taip pat sukuriamas konfiguracinis failas, kuriame nurodyti šie konvertavimo parametrai:

SOURCEFORMAT = WAV
TARGETKIND = MFCC_0_D
TARGETRATE = 100000.0
SAVECOMPRESSED = T
SAVEWITHCRC = T
WINDOWSIZE = 250000.0
USEHAMMING = T
PREEMCOEF = 0.97
NUMCHANS = 26
CEPLIFTER = 22
NUMCEPS = 12

čia

SOURCEFORMAT – šaltinio failo formatas *WAV*.

TARGETKIND – tikslo parametro rūšis, *MFCC* – dažnių skalės kepstro koeficientai, *_O* indikatorius rodo, kad pritaikomi kepstriniai koeficientai, *_D* indikatorius rodo, kad pritaikomi pirmos eilės regresijos koeficientai. Jis vadinamas delta koeficientu.

TARGETRATE – tikslo modelio periodas, vienetai 100 ns.

SAVECOMPRESSED – išsaugo suspaustą išėjimo failą.

SAVEWITHCRC – prideda kontrolinę sumą.

WINDOWSIZE – analizuojamo lango dydis, vienetai 250 ns.

USEHAMMING – naudoja *Hamming* langą.

PREEMCOEF – nustato pradinį koeficientą.

NUMCHANS – filtro kanalų skaičius.

CEPLIFTER – kepstrinio kėlimo koeficientas.

NUMCEPS – kepstrinių parametrų skaičius.

Konvertavimas atliekamas įvykdžius šią komanda:

HCopy –A –D –T 1 –C config –S codetrain.scp

čia

-A – atspausdina komandinės eilutės argumentus,

-D – parodo konfigūravimo kintamuosius,

-T – nustato žymėjimo vėliavėlę į 1, kuri parodo komandos vykdymo progresą,

-C – nurodo konfiguracinio failo vardą *config*,

-S – nurodo scenarijaus failo vardą *codetrain.scp*.

Po konvertavimo gauti dažnių skalės kepstro koeficientai, kurie aprašo garso failus. Turint šią informaciją nesunku išskirti požymio vektorius.

2.1.4. Paslėptųjų Markovo modelių apmokymas

2.4 paveiksle pavaizduota paslėptųjų Markovo modelių apmokymo schema. Atliekant PMM apmokymą visų pirmą aprašomas trijų būsenų pirminis modelis. Šio modelio parametrai nėra svarbūs. Jo tikslas yra apibrėžti modelio struktūrą.

HTK įrankis *HCompV* peržiūri *MFCC* formato failus, suskaičiuoja vidurkį ir kovariaciją, bei nustato duotam PMM gautą vidurkį ir kovariaciją. Kelias iki *MFCC* duomenų failų aprašomas *train.scp* scenarijuje. O iš konfiguracinio failo, naudoto garso įrašų kodavimui, pašalinama *SOURCEFORMAT* reikšmė. Taip pat pakeičiama *TARGETKIND* reikšmė į *MFCC_0_D_N_Z*. Papildomas indikatorius *_N* reikalingas energijos sustabdymui, o indikatorius *_Z* atlieka kepstrinio vidurkio normalizavimą.

Skaičiavimai atliekami įvykdžius komandą:

HCompV –A –D –T 1 –C config –f 0.01 –m –S train.scp –M hmm0 proto

čia

-*A* – atspausdina komandinės eilutės argumentus,

-*D* – parodo konfigūravimo kintamuosius,

-*T* – nustato žymėjimo vėliavėlę į 1, kuri parodo komandos vykdymo progresą,

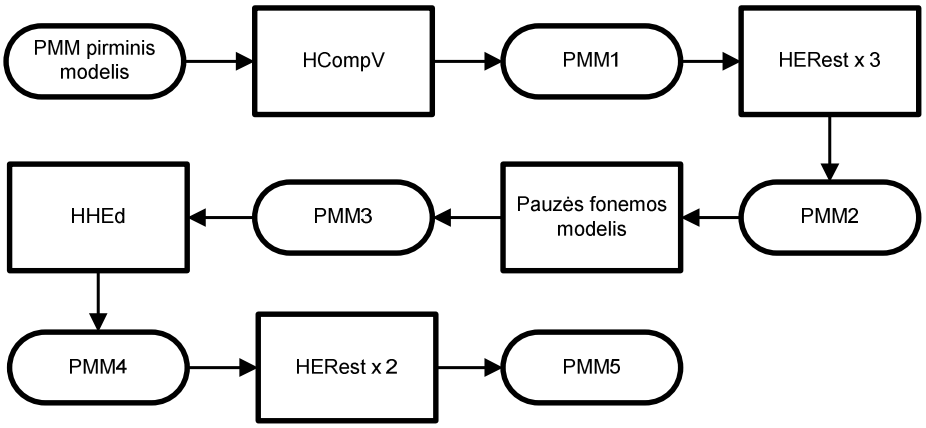
-*C* – nurodo konfiguracinio failo vardą,

-*f* – suskaičiuoja skirtumo minimumą, kurio reikšmės lygios 0,01 bendro skirtumo,

-*m* – atnaujina vidurkio reikšmes,

-*S* – nurodo scenarijaus failo vardą *train.scp*,

-*M* – nurodo katalogą *hmm0*, kuriame išsaugomas failas *proto* su pirminiais duomenimis.



2.4 pav. Paslėptųjų Markovo modelių apmokymo schema

Gauti duomenys atskirame faile pridedami prie kiekvienos fonemos ir iš naujo atliekami skaičiavimai. Perskaičiavimas atliekamas pasinaudojus *HTK* įrankiu *HERest*. *HERest* perskaičiuoja užkrautus fonemų modelius pasinaudojant *MFCC* failais.

HERest –A –D –T 1 –C config –I phones0.mlf –t 250.0 150.0 1000.0 –S train.scp –H hmm0/macros –H hmm0/hmmdefs –M hmm1 monophones0

čia

-*A* – atspausdina komandinės eilutės argumentus,

-*D* – parodo konfigūravimo kintamuosius,

-*T* – nustato žymėjimo vėliavėlę į 1, kuri parodo komandos vykdymo progresą,

-C – nurodo konfiguracinio failo vardą *config*,

-I – užkrauna *mlf* tipo failą *phones0.mlf*,

-t – nustatomas sumažinimo slenkstis, kuris riboja būsenų pasiskirstymą naudojant sumavimą pirmyn-atgal algoritme ir gali sumažinti skaičiavimo kiekį. Sumažinimo lygis nustatytas į 250.0. Jei perskaičiuojant atsiranda klaida, slenkstis padidinamas 150.0 ir perskaičiavimas kartojamas tol kol sėkmingai įvykdomas arba slenkstis pasiekia 1000.0.

-S – nurodo scenarijaus failo vardą *train.scp*,

-H – užkrauna PMM modelį,

macros – globalių nuostatų failas,

hmmdefs – PMM duomenų failas,

-M – katalogas kuriame išsaugomi nauji failai,

monophones0 – fonemų sąrašas be pauzės *sp* fonemos.

Tokiu būdu dar du kartus atliekamas sukurtų fonemų modelių perskaičiavimas, atsižvelgiant į požymio vektorius.

Prieš tai sukurtuose PMM modeliuose nebuvo įtrauktos tylos pauzės tarp žodžių *sp*. Tačiau buvo sukurtas tylos modelis *sil*, naudojamas sakinio pradžioje ir pabaigoje. *sil* modelis yra ilgesnis už *sp* modelį. Todėl *sp* modelio kūrimui reikia panaudoti centrinę *sil* modelio dalį, o po to šiuos abu modelius surišti. *sp* ir *sil* modelių surišimui naudojamas *HTK* įrankis *HHEd*. Šiam įrankiui panaudojamas scenarijus:

```
AT 2 4 0.2 {sil.transP}
AT 4 2 0.2 {sil.transP}
AT 1 3 0.3 {sil.transP}
TI silst {sil.state[3],sp.state[2]}
```

AT komandos prie duotų perėjimo matricų prideda perėjimus. *TI* komanda sukuria surištą modelį *silst*. Šio modelio parametrai saugomi kiekviename tylos modelyje.

Modelių surišimas atliekamas įvykdžius šią komandą:

```
HHEd –A –D –T 1 –H hmm4/macros –H hmm4/hmmdefs –M hmm5 sil.hed monophones1
```

čia

-A – atspausdina komandinės eilutės argumentus,

-D – parodo konfigūravimo kintamuosius,

-T – nustato žymėjimo vėliavėlę į 1, kuri parodo komandos vykdymo progresą,

-H – užkrauna PMM modelį,

macros – globalių nuostatų failas,

hmmdefs – PMM duomenų failas,

-M – katalogas kuriame išsaugomi nauji failai,

sil.hed – scenarijaus failas,

monophones1 – fonemų sąrašas su pauzės *sp* fonema.

Dar du perskaičiavimai su *HERest* įrankiu atliekami naudojant fonemų transkripcijas su *sp* pauzės modeliais tarp žodžių. Baigus skaičiavimus akustinis kalbos modelis yra apmokytas ir tinkamas naudoti kalbos atpažinime.

2.2. Žodžių skirstymas į kategorijas

Žodžių skirstymas į kategorijas – tai žodžių ar jų junginių sąrašų sudarymas, aprašant juos tam tikromis taisyklėmis. Į kategorijas galima įtraukti tik tuos žodžius, kurių fonemos buvo panaudotos akustiniame modelyje. Žodžių kategorijos yra naudojamos tam, kad apriboti atpažįstamų žodžių skaičių ir pagerinti kalbos atpažinimo tikslumą.

Žodžių kategorijų aprašymą galima suskirstyti į dvi dalis. Pirmojoje dalyje yra taisyklių rinkiniai, kurie aprašo kaip kalbos atpažinimo sistema turi elgtis su žodžių kategorijom. Antrojoje dalyje surašomi, taisyklių rinkinyje nurodytoms kategorijom priklausantys, žodžiai bei jų transkripcijos. Taisyklėmis yra galimybė aprašyti ne tik pavienių žodžių, bet ir žodžių sekų atpažinimą. Žodžių surašymas į vieną kategoriją atitinka įprastą žodžių atpažinimą, nenaudojant kategorijų.

Naudojant Julius programinę įrangą, taisyklių rinkinius reikia surašyti į *grammar* failą. Kategorijom priklausantys žodžiai bei jų transkripcijos surašomos voca faile.

Žodžių kategorijų taisyklėms aprašyti panaudotas *Backus-Naur* formatas [9]. Tai matematinis būdas aprašyti kalbą. Taisyklių rinkiniai aprašyti tokiu būdu:

Simbolis: [išraiška su simboliais]

čia

Simbolis – tai simbolis kurį galima išreikšti kitais simboliais,

išraiška su simboliais – tai išraiška susidedanti iš simbolių, kurie gali būti baigtiniai arba nebaigtiniai

Žodžių seka *grammar* faile aprašoma taip:

S : NS_B RAKTINIS_ZODIS ZODZIAI NS_E
ZODZIAI: ZODZIAI ZODIS
ZODZIAI: ZODIS

„S“ yra pradinis sakinio simbolis. *„NS_B“* ir *„NS_E“* atitinka tylą, kuri atsiranda prieš ir po norimos atpažinti išraiškos. *„S“*, *„NS_B“* ir *„NS_E“* yra būtini kategorijų aprašymo elementai. *„NS_B“*, *„NS_E“*, *„RAKTINIS_ZODIS“*, *„ZODIS“* yra baigtiniai simboliai – kategorijų pavadinimai, kurie atspindi žodžių kategorijose surašytus žodžius. *„ZODZIAI“* yra nebaigtinis simbolis, kuris neturi tiesiogiai priskirtų žodžių. Jis turi tolimesnį priskyrimą baigtiniam simboliui ir yra pakeičiamas išraiška *„ZODIS“*. Visi nebaigtiniai simboliai turi būti priskirti baigtiniam simboliui, kuris turi turėti jam priskirtų žodžių. Taip aprašius kategoriją ir ištarus *„RAKTINIS_ZODIS“* priklausantį žodį, kalbos atpažinimo sistema bando atpažinti visus ištartus žodžius iš kategorijos „*ZODIS“* iki ilgesnės tylos pauzės.

Pavienių žodžių atpažinimas aprašomas taip:

S : NS_B RAKTINIS_ZODIS ZODIS NS_E

Taip aprašyta kategorija leis atpažinti tik po vieną ištartą žodį.

Tokiu būdu galima sudaryti daug kategorijų. Ištarus kategorijos raktinį žodį, sekantys žodžiai bus ieškomi tik iš tai kategorijai priklausančių žodžių. Surašius žodžius į vieną kategoriją, raktinio žodžio naudoti nereikia.

voca failas turi žodžių sąrašus kiekvienai žodžio kategorijai nurodytai *grammar* faile. Kiekvienos žodžio kategorijos pradžioje turi būti *„%“* simbolis, kuris nurodo kategorijos pavadinimą. *voca* faile žodžiai suskirstyti į du stulpelius. Pirmajame stulpelyje yra žodžiai, kurie bus parodyti kaip atpažinti. Antrajame stulpelyje yra to žodžio transkripcijos.

Ištrauka iš tyrimui sukurto *voca* failo, kuriame žodžiai surašyti į vieną kategoriją:

% NS_B
<s> sil

% NS_E
</s> sil

% ZODIS
ABEJONĖ a bm e j olr nm eel
ABEJOTI a bm e j old tm i
ABIPUS a bm ik p u s
ABIPUSIS a bm i p uk sm i s
ABITŪRA a bm i t ul r ak
ABRIKOSAS a bm rm i k ok s a s

ABRIKOSMEDIS a bm rm i k ok sm mm e dm i s
ABU a b uk
ABUDU a b uk d u
ABĖCĖLINIS a bm eel cm eelr lm i nm i s
ABĖCĖLĖ a bm eel cm eelr lm el
ACETATAS a cm e t alr t a s
ACETONAS a cm e t ok n a s
ACTAS alr c t a s
ADATA alr d a t a
ADATINĖ a d a tm ik nm eel
ADIKLIS a dm ik k lm i s
ADMINISTRACIJA a dm mm i nm i s t r alr cm i j e
ADMINISTRACINIS a dm mm i nm i s t r alr cm i nm i s
ADMINISTRATORIUS a dm mm i nm i s t r alr t o rm u s
...

Šiam *voca* failui sukurtas *grammar* failas, kuris aprašo pavienių žodžių atpažinimą:

S : NS_B ZODIS NS_E

Tam pačiam *voca* failui sukurta taisyklė aprašanti žodžių sekų atpažinimą:

S : NS_B ZODZIAI NS_E
ZODZIAI: ZODZIAI ZODIS
ZODZIAI: ZODIS

Ištrauka iš *voca* failo, kuriame panaudota keletas kategorijų:

% NS_B
<s> sil

% NS_E
</s> sil

% MENUO
MĖNUO mm eeld n uo

% MENUO_ZODIS
SAUSIS s aulr sm i s
VASARIS v a s alr rm i s
KOVAS k old v a s
BALANDIS b a l ak nm dm i s
GEGUŽĖ gm e g u zzm eelr
BIRŽELIS bm i rm zzm elr lm i s
LIEPA lm ide p a
RUGPJŪTIS r u k pm j uld tm i s
RUGSĖJIS r u k sm eeld j i s

SPALIS s p alr lm i s
LAPKRITIS l alr pm k rm i tm i s
GRUODIS g r udo dm i s

% SAVAITE
SAVAITĖ s a v aldi tm eel

% SAVAITE_ ZODIS
PIRMADIENIS pm i r m alr dm ie nm i s
ANTRADIENIS a n t r alr dm ie nm i s
TREČIADIENIS tm rm e ccm elr dm ie nm i s
KETVIRTADIENIS km e tm vm i r t alr dm ie nm i s
PENKTADIENIS pm e nn k t alr dm ie nm i s
ŠEŠTADIENIS ssm e ss t alr dm ie nm i s
SEKMADIENIS sm e k m alr dm ie nm i s

% SKAICIUS
SKAIČIUS s k ailr ccm u s

% SKAICIUS_ ZODIS
NULIS n uk lm i s
VIENAS vm ide n a s
DU d uk
TRYS tm rm ilr s
KETURI km e t u rm ik
PENKI pm e nnm km ik
ŠEŠI ssm e ssm ik
SEPTYNI sm e pm tm il nm ik
AŠTUONI a ss t uo nm ik
DEVYNI dm e vm il nm ik
DEŠIMT dm elr ssm i m t
VIENUOLIKA vm ie n udo lm i k a
...

Ištrauka iš *grammar* failo, kuri aprašo *voca* faile nurodytų kategorijų žodžių sekų atpažinimą:

S : NS_B MENUO MENUO_SEKANTIS NS_E
MENUO_ SEKANTIS: MENUO_ SEKANTIS MENUO_ ZODIS
MENUO_ SEKANTIS: MENUO_ZODIS
S : NS_B SAVAITE SAVAITE_ SEKANTIS NS_E
SAVAITE_ SEKANTIS: SAVAITE_ SEKANTIS SAVAITE_ ZODIS
SAVAITE_ SEKANTIS: SAVAITE_ ZODIS
S : NS_B SKAICIUS SKAICIUS_ SEKANTIS NS_E
SKAICIUS_ SEKANTIS: SKAICIUS_ SEKANTIS SKAICIUS_ ZODIS
SKAICIUS_ SEKANTIS: SKAICIUS_ ZODIS
...

Ši *grammar* failo ištrauka aprašo pavienių žodžių atpažinimą:

S : NS_B MENUO MENUO_ ZODIS NS_E
S : NS_B SAVAITE SAVAITE_ ZODIS NS_E
S : NS_B SKAICIUS SKAICIUS_ ZODIS NS_E
...

Ištrauka iš *voca* failo, kuri aprašo *Microsoft Office Word 2003* meniu esančius žodžius:

```
% NS_B
<s>       sil

% NS_E
</s>      sil

% FAILAS
FAILAS       f ailr l a s

% FAILAS_MENIU
NAUJAS       n aulr j e s
ATIDARYTI    a tm i d a rm ild t i
UŽDARYTI     u zz d a rm ild t i
ĮRAŠYTI      ild r a ssm ild tm i
KAIP         k ailr p
TINKLAPĮ     tm ik nn k l a pm il
FAILO        f ailr l o
IEŠKA        j ie ss k ak
VERSIJOS     vm ek rm sm i j o s
TINKLAPIO    tm ik nn k l a pm o
PERŽIŪRA     pm elr rm zzm ul r a
PUSLAPIO     p uk s l a pm o
PARAMETRAI           p a r alr mm e t r ai
SPAUDINIO    s p aldu dm i nm o
SPAUSDINTI           s p aldu zm dm i nm tm i
YPATYBĖS     il p a tm ild bm eel s
IŠĖJIMAS     i ssm eel j ik m a s

% FAILAS_LEIDIMAS
LEIDIMAS     lm ei dm ik m a s

% FAILAS_LEIDIMAS_MENIU
NERIBOTA     nm e rm i b old t a
PRIEIGA      pm rm ik ei g a
NEPASKIRSTYTI        nm e p a sm km ik rm sm tm il tm i
APRIBOTI     a pm rm i b old tm i
TEISES       t eilr sm e s
KAIP         k ailr p

% FAILAS_SIUSTI
SIŲSTI       sm ulr sm tm i
```

% FAILAS_SIUSTI_MENIU
SIŲSTI sm ulr sm tm i
GAVĖJUI g a vm eeld j u i
PAŠTO p alr ss t o
GAVĖJAS g a vm eeld j a s
...

Šio *grammar* failo ištrauka aprašo *Microsoft Office Word 2003* meniu valdymo taisykles:

S : NS_B FAILAS FAILAS_MENIU_X NS_E
FAILAS_MENIU_X : FAILAS_MENIU_X FAILAS_MENIU
FAILAS_MENIU_X : FAILAS_MENIU
S : NS_B FAILAS FAILAS_LEIDIMAS FAILAS_LEIDIMAS_MENIU_X NS_E
FAILAS_LEIDIMAS_MENIU_X : FAILAS_LEIDIMAS_MENIU_X
FAILAS_LEIDIMAS_MENIU
FAILAS_LEIDIMAS_MENIU_X : FAILAS_LEIDIMAS_MENIU
S : NS_B FAILAS FAILAS_SIUSTI FAILAS_SIUSTI_MENIU_X NS_E
FAILAS_SIUSTI_MENIU_X : FAILAS_SIUSTI_MENIU_X FAILAS_SIUSTI_MENIU
FAILAS_SIUSTI_MENIU_X : FAILAS_SIUSTI_MENIU
...

Kalbos atpažinimo sistema, kad galėtu panaudoti *grammar* ir *voca* failus juos reikia sukompiliuoti į *dfa*, *term* ir *dict* failus. Tai padaroma naudojant *Julius* programinės įrangos įrankį *mkdfa.pl*. Tai žodžių kategorijų kompiliatorius. *mkdfa.pl* scenarijus ieško pradinio sakinio simbolio *„S“ grammar* faile ir pakeičia žodžio kategorijas visais įmanomais žodžio sąrašais iš *voca* failo, taip padarydamas galimų žodžių ir frazių kombinacijas, kurias turi atpažinti kalbos atpažinimo sistema. Sukompiliavus gaunami *Julius* formato failai: *dfa*, *term*, *dict*. *dfa* failas – tai determinuotai baigtinis automatizavimo failas. *dict* failas yra žodyno failas. *term* failas – tai baigtinių simbolių failas.

2.3. Julius programinė įranga

Julius – tai didelio našumo, atviro kodo kalbos atpažinimo sistema, skirta kalbos atpažinimo tyrėjams ir programų kūrėjams. Ji veikia paslėptųjų Markovo modelių pagrindu. Gali atlikti realaus laiko kalbos atpažinimą. Maksimalus žodyno dydis yra 65535 žodžiai. Ši sistema palaiko su *HTK*, *CMU-Cam SLM* įrankiais sukurtus akustinius modelius. Sukurta dirbti *Linux* ar *Unix* operacinėse sistemose, tačiau sukurta ir *Windows* operacinei sistemai skirta versija. Šią kalbos

atpažinimo sistemą galima panaudoti, bet kurios kalbos atpažinimui. Julius sukurtas Japonijoje ir šiuo metu vystomas ISTC konsorciumo [18].

Julius paketo įrankis *Julian* – tai žodžių kategorijų pagrindu veikiantis kalbos atpažinimo variklis. Jis yra didelio našumo ir gali atlikti realaus laiko kalbos atpažinimą. Ši sistema supranta *Backus-Naur* formatu aprašytas žodžių kategorijas. *Julian* gali atlikti kalbos atpažinimą iš garso failų, mikrofono ir požymio parametrų failų.

Į *Julius* paketą įeina ir daugiau įrankių skirtų kalbos atpažinimui. Tai žodžių kategorijų kompiliatorius *mkdfa.pl*. Taip pat yra žodžių sekų generatorius *generate*, kuris pagal paruoštas žodžių kategorijų taisykles sugeneruoja žodžių sekas.

Kijoto universitete sukurta *Julius* versija skirta *Microsoft Windows Speech API 5.1*. Tokiu būdu *Julius* galima panaudoti, kaip *SAPI* kalbos atpažinimo sistemą įvairioms programoms. Aprašytas žodžių kategorijas su įrankiu *gram2sapixml* galima konvertuoti į *SAPI XML* formatą.

3. TYRIMO REZULTATAI

Kalbos atpažinimo sistemos tikslumas buvo vertinamas skaičiuojant žodžių atpažinimo procentą, kuris apskaičiuojamas pagal formulę:

$$WRR = \frac{N - S - D - I}{N} \cdot 100\% \tag{3.1}$$

čia

N – visas bandomų atpažinti žodžių skaičius,

S – klaidingai atpažintų žodžių skaičius,

D – praleistų žodžių skaičius,

I – įterptų žodžių skaičius [19].

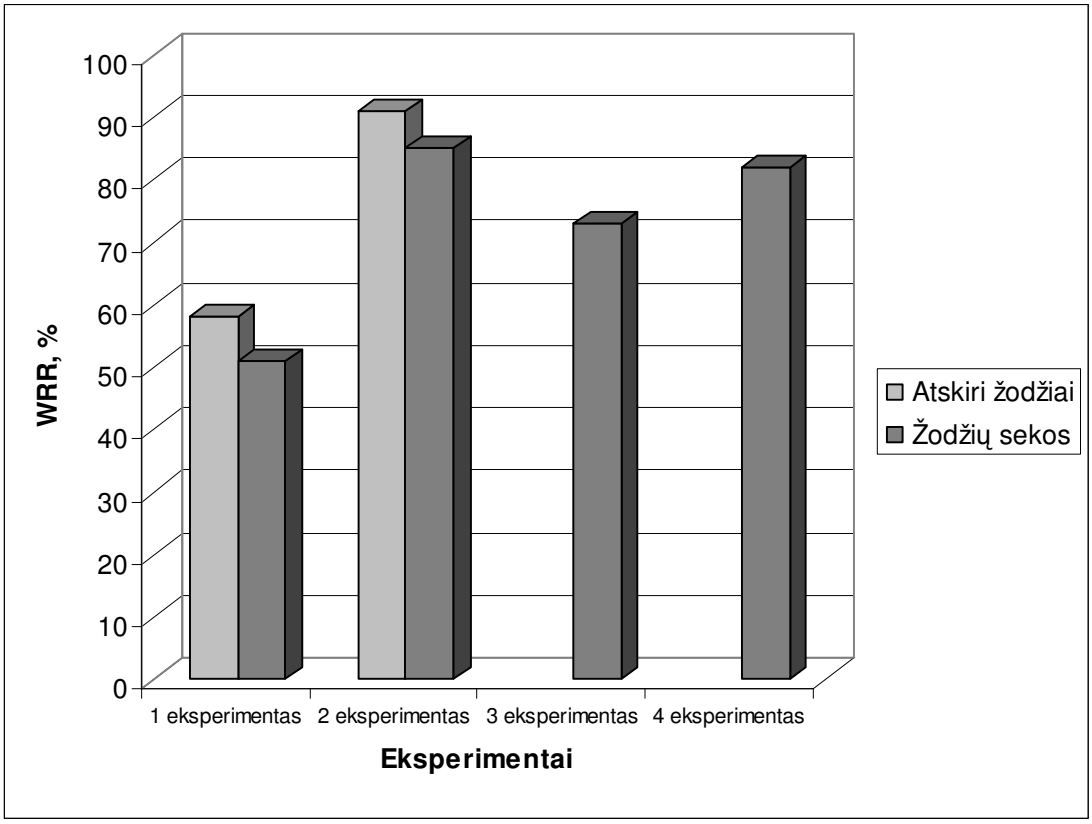
Eksperimentų metu buvo tiriama žodžių skirstymo į kategorijas įtaka kalbos atpažinimui. Tyrimui atlikti panaudota, žodžių kategorijų pagrindu veikianti, *Julius* paketo *Julian* kalbos atpažinimo sistema.

Pirmajame eksperimente tūkstantis žodžių iš paruošto transkripcijų žodyno buvo surašyti į vieną kategoriją, bei paruošti du taisyklių rinkiniai. Pirmas taisyklių rinkinys aprašo pavienių žodžių atpažinimą, antrasis rinkinys aprašo žodžių sekų atpažinimą. Pirmuoju atveju atliekant žodžių atpažinimą kiekvienas žodis buvo ištartas po vieną kartą. Antruoju atveju su *Julian* įrankiu *generate* buvo sugeneruotas šimtas žodžių sekų, kurios po to buvo diktuojamos kalbos atpažinimo sistemai. Žodžių surašymas į vieną kategoriją atitinka įprastą kalbos atpažinimą be kategorijų.

Antrajame eksperimente iš transkripcijų žodyno buvo sudaryta penkiolika žodžių grupių. Taip pat paruošti taisyklių rinkiniai atpažinti pavienius žodžius ir žodžių sekas. Pavyzdžiui kategorijoje *spalva* – surašytos spalvos, kategorijoje *mėnuo* – išvardinta dvylika mėnesiu. Kategorijoje *skaičius* – surašyti skaičiai nuo nulio iki dvidešimt, o taip pat trisdešimt, keturiasdešimt, penkiasdešimt, šešiasdešimt, septyniasdešimt, aštuoniasdešimt, devyniasdešimt, šimtai, šimtas, tūkstantis. Šioje kategorijoje galima atpažinti skaičius nuo nulio iki vieno tūkstančio devynių šimtų devyniasdešimt devynių. Atpažįstant pavienius žodžius buvo ištariamas kategoriją nurodantis žodis, bei vienas žodis iš tos kategorijos. Atpažįstant žodžių sekas buvo ištariamas kategoriją nurodantis žodis, bei vienas ar daugiau žodžių iš tos kategorijos. Kategoriją nurodantys žodžiai ir žodžiai iš kategorijų grupių buvo sugeneruoti *generate* įrankiu. Abiems taisyklėms buvo sugeneruota po šimtą žodžių rinkinių.

Trečiajame eksperimente buvo sumodeliuotas *Microsoft Office Word 2003* meniu valdymas atpažįstant meniu pavadinimus. Sudarytos devynios pagrindinės žodžių grupės: *Failas, Redagavimas, Rodymas, Įterpimas, Formatavimas, Įrankiai, Lentelė, Langas, Žinynas*. Kadangi kai kuriuose meniu yra papildomi vidiniai meniu, todėl šiose grupėse aprašomi papildomi pogrupiai. Pavyzdžiui grupėje *Failas* yra pogrupis *Leidimas*, kuriam priklauso žodžiai *Neribota prieiga, Nepaskirstyti , Apriboti teises kaip*. Modelio testavimui su *Julian* įrankiu *generate* sugeneruota šimtas žodžių rinkinių.

Ketvirtajame eksperimente akustinis modelis buvo papildomai apmokytas žodžiais esančiais *Microsoft Office Word 2003* meniu. Po to tyrimas pakartotas kaip ir trečiajame eksperimente.



3.1 pav. Žodžių atpažinimo tikslumas

Atlikus tyrimą gauti rezultatai pavaizduoti 3.1 paveiksle. Blogiausi rezultatai gauti atliekant pirmąjį eksperimentą. Atpažįstant žodžių sekas gaunamas tik 51 % žodžių atpažinimo tikslumo. Bandant atpažinti pavienius žodžius žodžių atpažinimo tikslumas padidėjo 7 % ir siekė 58 %. Atskirų žodžių atpažinimo tikslumas didesni už žodžių sekų atpažinimo tikslumą, nes prie klaidingai atpažintų žodžių prisideda ir klaidingai įterptų ir praleistų žodžių skaičius. Pavyzdžiui,

ištarta žodžių seka *gubernatorius, kedras, agentūra*, atpažinta, kaip *gubernatorius, du, kedras, architektūra*. Įterptas netartas žodis *du*, o žodis *agentūra* sumaišytas su žodžiu *architektūra*. Bandant atpažinti žodžių seką *kaklaraištis, tigras, ikona, būdvardis*, praleistas žodis *ikona*. Atpažįstant pavienius žodžius dažniausiai sumaišomi panašiai skambantys žodžiai: *akmeninis – aritmetinis, kampanija – kompanija, padas – badas, abejonė – abejoti, gruodas – gruodis.*

Antrojo eksperimento metu, surašius žodžius į kategorijas, rezultatai gauti žymiai geresni. Žodžių sekų atpažinimo tikslumas padidėjo iki 85 %. Pavienių žodžių atpažinimo tikslumas buvo didžiausias ir siekė 91 %. Šiuo atveju atpažinimo tikslumas padidėjo, nes suskirstant žodžius į kategorijas sumažėjo žodžių kiekis, iš kurio turi rinktis kalbos atpažinimo sistema. Tačiau vis tiek pasitaikė klaidingai atpažintų, įterptų ar praleistų žodžių. Pavyzdžiui, pavienių žodžių atpažinime padaryta klaida atpažįstant žodžius *mėnuo rugpjūtis*. Šie žodžiai atpažinti kaip *mėnuo rugsėjis*. Žodžių seka *skaičius šeši šimtai keturiasdešimt du*, atpažinta kaip *skaičius šešiolika šimtai keturiasdešimt*. Šioje sekoje žodis *šeši* sumaišytas su žodžiu *šešiolika*, o žodis *du* praleistas.

Atliekant trečiąjį eksperimentą gautas 73 % žodžių atpažinimo tikslumas. Šio eksperimento metu dažnai buvo maišomi ir visiškai skirtingai skambantys žodžiai, tokie kaip *pastraipa – stulpeliai, laukas – failas*. Sumažėjusio atpažinimo tikslumo priežastis yra ta, kad dauguma naudotų žodžių nebuvo panaudoti kuriant akustinį modelį.

Ketvirtajame eksperimente akustinį modelį apmokius žodžiais esančiais *Microsoft Office Word 2003* meniu atpažinimo tikslumas padidėjo 9 % ir siekė 82 %. Liko praleistų trumpų žodžių *ir*, *kaip*. Pasitaikė papildomai įterpti žodžiai *viską*, *laukas*. Buvo sumaišomi panašiai skambantys žodžiai *tamsva* ir *tamsiai*.

IŠVADOS

Šiame darbe iš 1410 žodžių buvo sukurtas lietuvių kalbos akustinis modelis. Kuriant šį modelį panaudotos 87 fonemos. Sudarant transkripcijų žodyną atsižvelgta į kirčiavimą, priebalsių minkštumą ir kietumą, balsių ilgumą ir trumpumą. Akustinio modelio kūrime panaudota paslėptųjų Markovo modelių metodika, kuri remiasi tikimybinių modelių sudarymu.

Naudojant *Backus-Naur* formatą sudaryti žodžių kategorijų rinkiniai. Pirmiausia žodžiai surašyti į vieną kategoriją, kas atitinka įprastą kalbos atpažinimą be kategorijų. Antru atveju sudaryta penkiolika žodžių kategorijų. Taip pat paruošti taisyklių rinkiniai, aprašantys pavienių žodžių ir žodžių sekų atpažinimą. Naudojant kategorijas sumodeliuotas *Microsoft Office Word 2003* meniu valdymas.

Surašius žodžius į kategorijas, pavienių žodžių atpažinimo tikslumas padidėjo 33 % ir siekia 91 %. Žodžių sekų atpažinimo tikslumas nuo 51 % padidėjo iki 85 %.

Dėl klaidingai praleistų ir įterptų žodžių, pavienių žodžių atpažinimo tikslumas, skirstant juo į kategorijas, 6 % didesnis už žodžių sekų atpažinimo tikslumą. Nenaudojant žodžių skirstymo į kategorijas atpažinimo tikslumas didesnis 7 %.

Eksperimente panaudoti žodžiai, kurie nebuvo įtraukti į akustinį modelį, 9 % sumažino žodžių atpažinimo tikslumą.

LITERATŪRA

1. Rabiner L., Juang B. H. Fundamentals of Speech Recognition [interaktyvus]. Prentice Hall, 1993 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://portal.acm.org/citation.cfm?id=153687&dl=>.

2. Young, S., Evermann G., Kershaw D., Moore G., Odell J., Ollason D., Povey D., Valtchev V., Woodland P. The HTK Book [interaktyvus]. Cambridge University Engineering Department, 2006 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://htk.eng.cam.ac.uk/docs/docs.shtml>.

3. Grune D., Jacobs C. J. H. Parsing Techniques A Practical Guide, 2nd ed [interaktyvus]. Springer, 2008 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://www.cs.vu.nl/~dick/PT2Ed.html>.

4. Kyoto University. Multipurpose Large Vocabulary Continuous Speech Recognition Engine Julius rev. 3.2 [interaktyvus]. 2001 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://julius.sourceforge.jp/book/Julius-3.2-book-e.pdf>.

5. Schroeder M. R. Computer speech: recognition, compression, synthesis [interaktyvus]. Springer, 2004 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://books.google.com/books? id=cg7x5_-vB-AC&printsec=frontcover>.

6. Jurafsky D., Martin J. H. Speech and Language Processing [interaktyvus]. 2008 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://www.cs.colorado.edu/~martin/slp2.html>.

7. Jelinek F. Statistical Methods for Speech Recognition [interaktyvus]. 1997 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://books.google.lt/books?id=1C9dzcJTWowC&printsec=frontcover&hl=en#PPP1,M1>.

8. Navakauskas D., Paulikas Š., Urbanavičius V., Martavičius R. Šiuolaikinės SSA priemonės [interaktyvus]. [žiūrėta 2008-01-19]. Prieiga per internetą: <http://www2.el.vgtu.lt/ssa/index.html>.

9. Garshol L. M. BNF and EBNF: What are they and how do they work? [interaktyvus] [žiūrėta 2008-01-19]. Prieiga per internetą: <http://www.garshol.priv.no/download/text/bnf.html>

10. Park S. W. Linear Predictive Speech Processing. [žiūrėta 2008-01-19]. Prieiga per internetą: <http://www.iua.upf.es/~xserra/cursos/TDP/referencies/Park-LPC-tutorial.pdf>.

11. Tebelskis J. Speech Recognition using Neutral Networks [interaktyvus]. 1995 [žiūrėta 2008-01-19]. Prieiga per internetą: < http://citeseer.ist.psu.edu/tebelskis95speech.html >.

12. Filipovič M. Atskirai pasakytų žodžių atpažinimo, naudojant neuroninius tinklus, tyrimas. Konferencijos „Informacinės technologijos 2003“ pranešimų medžiaga. KTU, Kaunas, 2003. p. 10-20.

13. Maskeliūnas R., Ratkevičius K., Rudžionis A., Rudžionis V. (2007). Balso serverių apžvalga. Konferencijos „Informacinės technologijos 2007“ pranešimų medžiaga. KTU, Kaunas, 2007. p. 47-51.

14. Raškinis G., Raškinienė D. Trumpų rišlių lietuvių šnekos frazių atpažinimas, naudojant paslėptų Markovo modelių metodiką. Konferencijos „Informacinės technologijos 2004“ pranešimų medžiaga. KTU, Kaunas, 2004. p. 33–38.

15. Štrimaitis M. Perėjimo momentų tarp fonemų akustinis modeliavimas naudojant paslėptus Markovo Modelius. Konferencijos “Informacinės Technologijos 2004” pranešimų medžiaga. KTU, Kaunas, 2004. p. 51–55.

16. Vitkauskas V. Lietuvių kalbos tarties žodynas. Vilnius, 2001. 318 p. ISBN 5-420-01482-3.

17. Kawahara T., Lee A., Kobayashi T., Takeda K., Minematsu N., Sagayama S., Itou K., Ito A., Yamamoto M., Yamada A., Utsuro T., Shikano K. Free software toolkit for Japanese large vocabulary continuous speech recognition [interaktyvus]. Int'l Conference on Spoken Language Processing, 2000 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://julius.sourceforge.jp/paper/icslp00-6.pdf>.

18. Lee A., Kawahara T., Shikano K. Julius – an open source real-time large vocabulary recognition engine [interaktyvus]. European Conference on Speech Comunication and Technology, 2001 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://julius.sourceforge.jp/paper/ri-eurospeech2001.pdf>.

19. McCowan I., Moore D., Dines J., Gatica-Perez D, Flynn M., Wellner P., Bourlard H. On the use of information retrieval measures for speech recognition evalution [interaktyvus]. IDIAP Research Institute, 2005 [žiūrėta 2008-01-19]. Prieiga per internetą: <http://www.idiap.ch/publications/mccowan-rr-04-73.bib.abs.html>.

PRIEDAS

KOMPAKTINIS DISKAS