
**Kauno technologijos universitetas**

Informatikos fakultetas

# Lietuviškos šnekos atpažinimo tikslumo tyrimas su telefoninių pokalbių garsynu

Baigiamasis magistro projektas

**Giedrius Stravinskas**

Projekto autorius

**Prof. Dr. Rytis Maskeliūnas**

Vadovas

**Kaunas, 2022**


**Kauno technologijos universitetas**

Informatikos fakultetas

# Lietuviškos šnekos atpažinimo tikslumo tyrimas su telefoninių pokalbių garsynu

Baigiamasis magistro projektas

Dirbtinio intelekto informatika

**Giedrius Stravinskas**

Projekto autorius

**Prof. Dr. Rytis Maskeliūnas**

Vadovas

**Dr. Liudas Motiejūnas**

Recenzentas

**Kaunas, 2022**


**Kauno technologijos universitetas**

Informatikos fakultetas

Giedrius Stravinskas

# Lietuviškos šnekos atpažinimo tikslumo tyrimas su telefoninių pokalbių garsynu

## Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Giedriaus Stravinsko, baigiamasis projektas tema „Lietuviškos šnekos atpažinimo tikslumo tyrimas su telefoninių pokalbių garsynu“ 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) |

**Kauno technologijos universitetas**

Informatikos fakultetas

**Baigiamojo magistro projekto užduotis**

| | |
|---|---|
| Projekto tema | |
| Reikalavimai ir sąlygos (tikslinti pavadinimą pagal poreikį) | |
| Vadovas / Vadovė | |
| | (vadovo pareigos, vardas, pavardė, parašas) (data) |

Stravinskas, Giedrius. Lietuviškos šnekos atpažinimo tikslumo tyrimas su telefoninių pokalbių garsynu. Magistro baigiamasis projektas vadovas Prof. Dr. Rytis Maskeliūnas; Kauno technologijos universitetas, informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai. Informatika.

Reikšminiai žodžiai: lietuvių kalbos transkripcija, kalbos vertimas į tekstą, telefoninių skambučių transkripcija

Kaunas, 2022. 58 p.

**Santrauka**

Šių dienų kontekste telefoninės kalbos vertimas į tekstą yra aktualus uždavinys. Panaudojimas gali būti pritaikomas natūralios kalbos tyrimams ar konteksto iš garso įrašų išgavimui, kadangi norint atlikti informacijos tyrimą kalbos įrašai turi būti perteikti tekstu. Komerciniai sprendimai dažniausiai nėra pritaikyti telefoninei kalbai. Taip pat dauguma didžiųjų kalbos vertimo į tekstą paslaugų tiekėjų nėra suinteresuotos kurti sprendimų sąlyginai nedidelį kiekį kalbančiųjų turinčioms kalboms, tokioms, kaip lietuvių. Minėtos priežastys ir lietuviškų telefoninių skambučių anotuotų duomenų trūkumas lemią mažą šių uždavinių tyrimų kiekį. Tačiau vis labiau siekiant automatizuoti paslaugų tiekimą, supaprastinti įrenginių įvestį ir dėl naujausių pasiekimų giliojo mokymosi algoritmų pritaikymo kalbos vertimo į tekstą tematika sulaukia vis daugiau dėmesio.

Šiame darbe yra apžvelgiamos kalbos į tekstą vertimo egzistuojančios modelių architektūros. Tiriami geriausius rezultatus šiuo metu pasiekiantys kalbos į tekstą vertimo modeliai juos apmokant su telefoninių skambučių duomenimis. Atsižvelgiama į tokius kriterijus, kaip tikslumas, greitaveika, mokymo greitis. Taip pat siekiant pagerinti rezultatus siūloma naudoti metodika pridedant žodžių gramatinių klaidų taisymo algoritmą su dažnių žodyno paieška.

Atlikus tyrimą nustatyti tikslumai komerciniam metodui ir dviem modeliam apmokytiem su lietuviškų skambučių duomenų rinkiniu. Palyginta spėjimo atlikimo greitaveika su dviem skirtingomis „NVIDIA“ vaizdo plokštėmis. Įvertintas tikslumų skirtumas tarp originalių metodų ir metodų pridedant pasiūlytą metodiką pasitelkiant žodžių dažnių žodyno paiešką.

Stravinskas, Giedrius. Research of speech recognition accuracy using Lithuanian phone call corpus. Master's Final Degree Project supervisor Prof. Dr. Rytis Maskeliūnas; Informatics Faculty, Kaunas University of Technology.

Study field and area (study field group): Computing. Informatics.

Keywords: Lithuanian language transcription, speech to text, phone call transcription

Kaunas, 2022. 58 p.

**Summary**

In today’s context, translating a telephone language into a text is a relevant task. The use can be adapted to the study of natural language or the extraction of context from sound recordings, as language recordings must be conveyed into text in order to carry out the study of information. Commercial solutions are usually not tailored to telephone language. Also, most major language translation service providers are not interested in developing solutions for languages with a relatively small number of speakers, such as Lithuanian. The above-mentioned reasons and the lack of annotated data on Lithuanian telephone calls lead to a small amount of research made on these tasks. However, the topic of language-to-text translation is gaining more and more attention due to desire to automate the delivery of services, simplify the device input, and the latest advances of deep learning algorithms.

In this work, the existing model architectures of language to text translation are reviewed. The best-performing text-to-text translation models for training with telephone call data are being investigated. Criteria such as accuracy, speed, training speed are taken into account. It is also proposed to use a methodology for adding grammatical error correction algorithm with a frequency dictionary search to improve the results.

After the research, the accuracy of the commercial method and two models that were trained with Lithuanian calls data set were determined. The performance speed of the inference was compared among two “NVIDIA” graphics cards. The difference in accuracy between original methods and methods with added proposed methodology to additionally use a word frequency vocabulary search was calculated.

**Turinys**

## Lentelių sąrašas

## Paveikslų sąrašas

## Santrumpų ir terminų sąrašas

Neuroninis tinklas (angl. *neural network*) - tarpusavyje sujungtų dirbtinių neuronų grupė.;

Prarasties funkcija (angl. *loss function*) – fukcija skirta išmatuoti, kaip gerai algoritmas modeliuoja duotus duomenis;

Epocha (angl. *epoch*) – modelio mokymo proceso dalis, kai pateikiami visi mokymo duomenų rinkinio duomenys modeliui. Tokių pateikimų gali būti ne vienas;

Mokymo grupė (angl. - *batch*) – mokymo duomenų kiekis naudojamas per vieną mokymo žingsnį.

Mokymo žingsnis (angl. *training step*) – vienas gradiento atnaujinimas. Per vieną mokymo žingsnį apdorojama viena mokymo grupė;

P – reikšmė (angl. *p – value*) – reikšmingumo lygmuo arba statistinis patikimumas;

WER (žodžių klaidos dažnis) – kalbos vertimo į tekstą klaidos metrika, skaičiuojanti žodžių klaidų dažnį per 100 žodžių;

CER (simbolių klaidos dažnis) – kalbos vertimo į tekstą klaidos metrika, skaičiuojant simbolių klaidos dažnį per 100 simbolių.

## Įvadas

Šiame tyrime nagrinėjamas šnekos atpažinimas ir perteikimas į tekstą iš lietuviško telefoninių įrašų garsyno. Toliau įvade perteikiama temos problematika ir specifinė sritis, išvardijami uždaviniai, pagrindžiamas aktualumas.

## Problemos kontekstas

Kuriant kompiuterinius įrenginius, visada yra stengiamasi padaryti valdymą kuo intuityvesnį vartotojui. Šis procesas apima tokių įvesties įrenginių, kaip perfokortos, kurios buvo naudojamos valdyti pirmiesiems kompiuteriams, pelė ir klaviatūra, kurie yra gerai žinomi visiems šios kartos atstovams, bei lietimui jautrių ekranų, sukūrimą [4]. Tačiau žmogui pats intuityviausias, kasdienybėje naudojamas valdymo ir komunikavimo įrenginys yra balsas. Būtent dėl šios priežasties vis daugiau mokslinių bendruomenių ir didelių kompanijų, tokių kaip „Amazon“, „Apple“, „Baidu“ ir panašios, atlieka automatinio kalbos atpažinimo tyrimus [5]. Geriausi rezultatai yra pasiekti anglų kalboje, kur žodžių klaidos dažnis (ang. *word error rate*) yra lygus 1.4 %/2.6 % atliekant testavimą su plačiausiai naudojamu „LibriSpeech test/test-other“ duomenų rinkiniu [6]. Nors ir rezultatai pasiekti su anglų kalbos duomenimis ir džiugina, tačiau automatinės kalbos atpažinimo projektai reikalauja didelių piniginių išteklių. Todėl kompanijų pagrindinis tikslas yra susigrąžinti investicijas ir didžiausias dėmesys yra kreipiamas į plačiai paplitusias kalbas. Dėl šios priežasties kompanijos nėra suinteresuotos sąlyginai mažai vartojamomis kalbomis, tokiomis, kaip lietuvių.

Tačiau vidaus rinkoje atsiranda vis didesnė automatinio kalbos atpažinimo paklausa. Paslaugų robotizavimas ir automatizavimas yra vienas pagrindinių šių dienų uždavinių. Ne išimtis ir vartotojų aptarnavimas nuotoliniu būdų. Nors jau yra ir kitų vartotojų aptarnavimo kanalų (pvz., elektroniniai laiškai, socialinių tinklų bendravimo platformos, pokalbio robotai ar kt.), telefoniniai skambučiai išlieka vienu iš svarbiausių aptarnavimo kanalų. Įmonės siekdamos gerinti savo telefoninių paslaugų kokybę, nori įsidiegti ir šneką atpažinti gebančias technologijas. Be paslaugų kokybės gerinimo, automatizuojant skambučių centrų veiklą yra pašalinama pasikartojanti, alinanti veikla, lemianti didelę darbuotojų kaitą užtikrinant darbinės aplinkos patogumą ir susitelkimą į svarbiausius darbus [7]. Paverčiant skambučių įrašus tekstu, taip perteikiant nestruktūrizuotą informaciją į struktūrizuota, lengviau atlikti jų analizę ir verslui svarbias metrikas ar įžvalgas. Tačiau tai nėra vienintelis tokių sistemų panaudojimas, kadangi pavertus balsą kompiuteriui atpažįstama išraiška, šios technologijos gali būti taikomos įvairiems aparatams valdyti balsu, tam tikriems žodžiams šalinti ar atpažinti iš garso įrašų, norint pašalinti keiksmažodžius ar identifikuoti vykdomą nusikaltimą.

Lietuvių kalbos, kaip ir daugelio kalbų, kurios nėra plačiai naudojamos, kalbos garsynų anotuotų duomenų nėra daug. Yra garsynas „LIEPA“, kuris yra 113 valandų ilgio. Taip pat yra „LIEPA-2“ garsynas, kuriame yra 1000 valandų garso įrašų. Tokio kiekio duomenų neužtenka gerai veikiančiam telefoninės kalbos atpažinimo modeliui sukurti. Duomenys šiuose garsynuose yra sudaryti iš garso studijose įrašytų kalbos įrašų: audioknygų (3 %), diktofonu surinktų įrašų (28 %), studijoje surinktų įrašų (62 %), televizijos laidų įrašų (3 %), radijo laidų įrašų (2 %) ir telefoninių įrašų (2 %) [9]. Pagrindą šių duomenų sudaro geros kokybės studijoje surinkti garso įrašai (62 %) ir tik mažoji dalis duomenų yra telefoninės kalbos įrašai (2 %). Duomenys įrašyti studijoje turi geresnę kokybę (diskretizavimo dažnis 16 kHz), tai yra izoliuoti garsai, kur vienu metu kalba tik vienas diktorius, nėra papildomo triukšmo. Garso įrašai iš telefoninių skambučių dažnai yra prastesnės kokybės (diskretizavimo dažnis 8 kHz), gali turėti nutrūkstantį garsą, kai nutrūksta ryšys, taip pat gali

būti papildomo triukšmo iš aplinkos (kalbantys žmonės, aplinkos garsai ir kt.) [10]. Todėl „LIEPA“ ir „LIEPA-2“ lietuviškos kalbos rinkiniai nėra optimalūs siekiant apmokyti tikslų telefoninės kalbos teksto į kalbą vertimo modelio. Šie duomenys neatspindi realių telefoninės kalbos įrašų, o duomenys yra vienas pagrindinių veiksnių, užtikrinančių tikslų dirbtinio intelekto modelio veikimą. Egzistuoja garsynas, sudarytas iš lietuviškų telefoninių skambučių. Tai viena iš projekto „IARPA Babel“ tarptautinio garsyno, skirto platų kalbų spektrą į tekstą paversti gebančiai technologijai sukurti [11]. Šį garsyną sudaro 210 valandų garso įrašų, iš kurių lietuvių kalbos yra ~40 valandų [12]. Tačiau tokia dalis duomenų nebūtinai užtikrins labai tikslius rezultatus. Kadangi vien tik pridedant į mokymo imtį daugiau duomenų, rezultatai gali pagerėti net keliolika procentų [13].

**Tyrimo sritis ir objektas**

Tyrimo rezultatai gali būti taikomi plačiai, kadangi gali būti pritaikomi daugeliui pasaulio kalbų, todėl tyrimo sritis šiuo atveju yra gana plati, tačiau yra apribota kalbos atpažinimu iš garso įrašų.

Tyrimo objektas – kalbos į tekstą vertimo neuroniniais tinklais grįsti algoritmai, galimos modifikacijos, siekiant užtikrinti tikslumą, mokymo spartumą ir greitaveiką. Neuroniniai tinklai ne tik padeda paprasčiau sukurti modelius, skirtus kalbai iš garso įrašų atpažinti, bet iškelia naujus ribojimus bei reikalavimus didesniems duomenų rinkiniams, todėl siekiant išmokyti efektyvų modelį reikia atsižvelgti į šiuos ribojimus.

**Tikslas ir uždaviniai**

Darbo tikslas – palyginti lietuviškų telefoninių garso įrašų atpažinimo tikslumą naudojant dirbtiniu intelektu grįstus algoritmus.

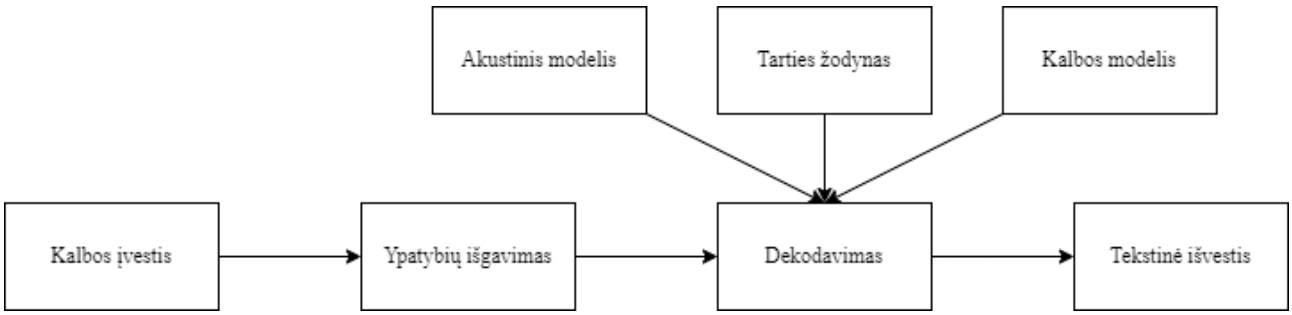
Tyrimo uždaviniai:
1. ištirti dirbtiniu intelektu grįstų algoritmų panaudojimą ir jų tikslumą;
2. paruošti lietuviškos šnekos atpažinimo modelio variacijas;
3. išmatuoti tikslumą paruoštoms šnekos atpažinimo modelių variacijoms;
4. pasiūlyti lietuvių kalbos gramatinių klaidų taisymo algoritmą, kuris užtikrintų tikslesnius šnekos atpažinimo rezultatus.

# 1. Kalbos vertimo į tekstą uždavinio analizė

## 1.1. Automatinis kalbos atpažinimas

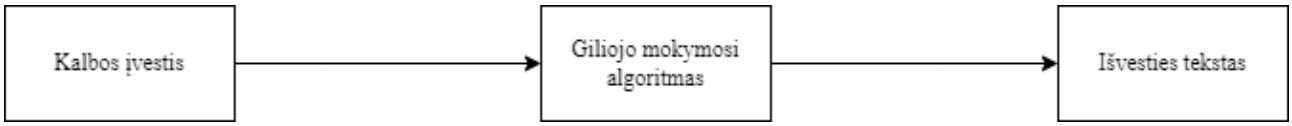
Kalbos atpažinimas yra ištariamų garsų vertimo tekstine išraiška uždavinys. Kalbos atpažinimas gali būti panaudotas valdymo balsu sąsaja valdomiems įrenginiams, pokalbio robotams, skambučių paslaugų centrų aptarnavimui ir t. t.. Kalbos atpažinimo sistemos gali būti skirstomos į izoliuoto žodžio (angl. *Isolated word*), apjungtų žodžių (angl. *Connected word*), vientisos kalbos (angl. *Continuous speech*), spontaniškos kalbos (angl. *Spontaneous speech*) ir kalbėtojui pritaikytos kalbos (angl. *Speaker adapted speech*). Izoliuotos kalbos garso įrašuose tariami žodžiai yra po vieną. Apjungtų žodžių garso įrašuose žodžiai yra tariami po kelis, bet su pertraukomis tarp žodžių tarimo. Vientisos kalbos atpažinime žodžiai yra tariami vienas po kito be pauzių. Spontaniškos kalbos įrašai yra kasdienės, natūralios kalbos garso įrašai [14].

Dažniausiai įprastinėms automatinio kalbos atpažinimo sistemoms reikia atskirai apmokytų akustinių, tarties ir kalbos modelio komponentų. Tarties leksikonų sudarymas, fonemų rinkinių apibrėžimas atskiroms kalboms reikalauja ekspertinių žinių ir reikalauja laiko. Siekiant atlikti kalbos atpažinimą, garso signalas yra paverčiamas į parametrinę formą, tuomet pagal ištrauktas ypatybes (angl. Extracted features) sukuriami akustiniai modeliai. Atpažinimo žingsnyje išgaunami parametriniai modelio skaičiavimai, kurie pagal tikimybes suranda žodžio atitikmenį ir ar jis buvo paminėtas (žr. **1 pav.**).




**1 pav.** Įprastinio automatinio kalbos atpažinimo modelio struktūra.

Ištisinės automatinės kalbos į tekstą vertimo sistemos tiesiogiai perverčia įvesties akustines ypatybes į grafemas arba žodžius (žr. **2 pav.**). Tokios sistemos yra treniruojamos optimizuoti kriterijų tiesiogiai susijusį su galutine uždavinio vertinimo metrika, kalbos atpažinime, dažniausiai tai žodžių klaidų dažnis (angl. *Word Error Rate*).




**2 pav.** Ištisinės automatinės kalbos atpažinimo modelio struktūra.

Galima matyti (žr. **1 pav.**, **2 pav.**), kad ištisinis kalbos atpažinimas ženkliai supaprastina tradicinį automatinės kalbos atpažinimo sistemos veikimą. Todėl ir supaprastėja pritaikymas įvairioms kalboms ir pritaikomumas.

## 1.2. Kalbos atpažinimo vertinimo metrikos

Architektūros tikslumui nustatyti svarbu pasirinkti tinkamas vertinimo metrikas, kadangi pagal jas yra vertinamas modelių tikslumas ir pagal gaunamus rezultatus daromi tolimesni pakeitimai kuriamoje architektūroje [24]. Dažniausiai straipsniuose susijusiuose su kalbos atpažinimu pasitaikanti vertinimo metrika yra žodžių klaidų dažnis (angl. *Word Error Rate*), taip pat raidžių klaidų dažnis (angl. *Character Error Rate*) [16].

Žodžių klaidų dažnis parodo kaip tiksliai architektūra arba modelis spėja žodžius. Dažnis skaičiuojamas atsižvelgiant į nekorektišką žodžių atpažinimą apibūdinamą toliau pateikiamomis išraiškomis – žodžių pakeitimus (angl. *Substitutions*), įterptis (angl. *Insertions*) ir žodžių pašalinimą (angl. *Deletions*). Apskaičiuojant procentinį koeficientą pakeitimai, įterptys ir žodžių pašalinimai yra padalijami iš bendro žodžių skaičiaus ir padauginama iš 100 norint gauti procentinę išraišką [25], formulė (1).

$$\text{Žodžių klaidų dažnis} = \frac{\text{Žodžių pakeitimai} + \text{Žodžių įterptys} + \text{Žodžių pašalinimai}}{\text{Visų žodžių skaičius}} * 100 \quad (1)$$

Raidžių klaidų dažnis parodo, kaip tiksliai architektūra arba modelis spėja sakinyje esančias raides. Ir skaičiuojamas panašiai, kaip žodžių klaidų dažnis, tik vietoj žodžių vertinant raidžių sakinyje pakeitimus, įterptis ir pašalinimus. Minėtų nekorektiškumo metrikų išraiškų sumą padalijant iš visu raidžių įraše sumos [16], formulė (2).

$$\text{Raidžių klaidų dažnis} = \frac{\text{Raidžių pakeitimai} + \text{Raidžių įterptys} + \text{Raidžių Žodžių pašalinimai}}{\text{Visų žodžių skaičius}} * 100 \quad (2)$$

Būtent šiomis metrikomis ir bus remiamasi matuojant atpažinimo tikslumą skambučių balso įrašų duomenų rinkiniui. Taip pat verta paminėti, kad į palyginimą reikia įtraukti ir sprendimo greičio metrikas – kiek laiko trunka sprendimo atiduodamas atsakymas, per kiek laiko balso įrašas yra paverčiamas tekstu, kadangi kuo mažesnis apdorojimo laikas tuo daugiau duomenų modelis gali apdoroti per tam tikrą laiko tarpą.

## 1.3. Automatinis telefoninės lietuvių kalbos atpažinimas

Lietuvių kalbai automatinės kalbos atpažinimo tyrimų nėra daug. Dažniausiai pasitaikantys tyrimai yra garso įraše žodžių radimui, taip pat tyrimai būna atliekami su geros kokybės garso įrašais, kuriuose kalba diktoriai. Telefoninės kalbos tyrimų yra dar mažiau ir yra atliekami su „Babel“ duomenų rinkiniu [17], [18]. Lileikytė ir kt. (2018) tyrime yra atliekamas grafemų (angl. *Grapheme*) vertimas fonemomis (angl. *Phoneme*). Todėl norint apmokyti modelius pasitelkiant naujus duomenis, jų negalima tiesiogiai mokyti su garso į tekstą transkripcijomis, kurios toliau išskaidomos į raides. Reikia paversti raides fonemomis, pagal papildomas taisykles, todėl šis metodas nėra labai patogus naudoti komercinėje veikloje, kur duomenys yra nuolatos papildomi naujais. Abiejuose straipsniuose atliekama žodžių paieška, neverčiamas visas garso įrašas į tekstą. Gales ir kt. (2015) straipsnyje pasiektas žodžių klaidos dažnis 68,6 % su 3 valandų treniravimo imtimi ir 48,3 % tikslumas su 40 valandų treniravimo imtimi, Lileikytė ir kt. (2018) straipsnyje paverčiant grafemas fonemomis žodžių klaidos metrika pagerėjo ~2 %.

## 1.4. Geriausi lietuvių kalbai pritaikyti kalbos atpažinimo metodai

Automatinio kalbos atpažinimo taikomosios programinės sąsajos (angl. *Applied programing interface*, API) paslaugas lietuvių kalba iš didžiųjų technologinių kompanijų teikia „Google“. Šios technologijos atpažinimo procentas matuojant žodžių klaidos dažnį su „LIEPA“ garsynu yra 40 % [10]. Detalesni rezultatai (žr. **3 pav.**).

| | Speech recognition results by speakers | | | | |
|---|---|---|---|---|---|
| | Average | Best | Average of 3 best speakers | Worst | Average of 3 worst speakers |
| WER, % | 40.74 | 10.00 | 14.74 | 100.00 | 96.39 |

**3 pav.** "Google" API rezultatai ant garsyno "LIEPA" [10]

Taip pat yra lietuviškų kompanijų, kurios transkribuoja garso failus į tekstą, lietuvių kalba. Viena tokių kompanijų yra „Tilde“, kuri gali transkribuoti failus arba tiesiogiai mikrofono įvestį. Šiuo metu šis sprendimas yra tiksliau veikiantis lietuvių kalbos garsų vertimo į tekstą užduotyje, nei „Google“ kompanijos sprendimas, ir pasiekiantis 21.8 % žodžių klaidos koeficientą, kuris yra beveik dvigubai tikslesnis, nei „Google“ kompanijos [5] (žr. **4 pav.**). Taip pat šis sprendimas yra viešai prieinamas tiek internetiniame „Tilde“ puslapyje, tiek telefoninėje aplikacijoje. Tačiau ankščiau minėti sprendimai nėra pritaikyti telefoninių pokalbių atpažinimui ir nėra tikslių matavimų, kaip šie sprendimai veikia ant duomenų surinktų iš skambučių. Kadangi telefoninė šneka turi trikdžių, tokių, kaip garso nutrūkimas, pašalinis triukšmas, balso iškraipymas, visi šie trikdžiai apsunkina balso atpažinimą. Taigi tyrimo rezultatų su telefoninės kalbos įrašų garsynu nelabai galima lyginti su esamu „Tilde“ ar „Google“ kompanijų rezultatais.

| Test set | Domain | Google | Alumäe&Tilk [4] | Ours |
|---|---|---|---|---|
| test_general | General domain | 40 (ignoring deletions - 26) | 25.2 | **21.8** |
| test_lt_radio | Radio broadcast | 54 (ignoring deletions - 27) | 32.3 | **29.2** |
| test_seimas | Seimas | 41 (ignoring deletions - 26) | 28.4 | **21.3** |

**4 pav.** "Tilde" kompanijos kalbos vertimo į tekstą rezultatai. [5]

Kaip jau buvo minėta, lietuvių kalba nesusilaukia didžiulio dėmesio iš mokslininkų, bei didelių kompanijų, kadangi turi tik apie tris milijonus kalbančiųjų. Taip pat dėl kalbos sudėtingumo ir skirtingo raidyno kitų kalbų modeliai negali būti tiesiogiai pritaikyti lietuvių kalbai, buvo bandymų rasti sąryšį tarp lietuviškų ir angliškų žodynų, kuris pasirodę esąs veiksmingas tik su mažu duomenų rinkiniu [19].

## 1.5. Geriausi anglų kalbai pritaikyti kalbos atpažinimo metodai

Pagrindinis duomenų rinkinys, naudojamas nustatyti anglų kalbos automatinio kalbos atpažinimo straipsniuose, yra „Librispeech“. Šis duomenų rinkinys yra paremtas viešai prieinamomis įgarsintomis knygomis. Sudarytas iš 1000 valandų 16 kHz dažnio kalbos įrašų. Šis duomenų rinkinys yra laisvai prieinamas visiems ir yra labai naudingas kuriant ir testuojant kalbos įrašų transkribavimo modelius [50].

Straipsnių apie automatinį anglų kalbos teksto atpažinimą yra labai daug. Tačiau vieni geriausių rezultatų buvo pasiekti dėl giliojo mokymosi tobulėjimo ir vis platesnio jo taikymo [21]. Tradicinės kalbos atpažinimo sistemos naudoja labai specifiškai sukuriamus apdorojimo žingsnius, skirtus garso įvedimo ypatybėms generuoti, akustiniam modeliui kurti bei paslėptiesiems Markovo modeliams (angl. *Hidden Markov Models,* HMM). Taip pat tokios sistemos yra pritaikomos kalbančiojo individo kalbėjimo stiliui, balso tembrui. Tokios architektūros remiasi leksikos žodynais, norint tiksliai perteikti balsą į tekstinę išraišką. Naudojant šią techniką žodžiai yra verčiami į vienos ar kelių fonemų (garso vienetai, kurie padeda atskirti vieną žodį nuo kito) sekas. Toliau fonemos yra dar toliau smulkinamos į dar mažesnius dalinių žodžių vienetus, vadinamus senonais. Senonai yra parenkami taikant procedūrą, apimančią fonetiniu kontekstu pagrįstą sprendimų medį, sukurtą panaudojant Gauso skirstinių mišinio modelį (angl. *Gaussian Mixture Model*, GMM) arba anksčiau minėtąjį paslėptąjį Markovo modelį. Kiekvieną kartą ruošiant modelį tokie žingsniai pasunkina duomenų paruošimą ir pačių akustinių modelių panaudojimą [45], [46].

Problemai spręsti buvo pasiūlyti architektūros atliekančios kalbos atpažinimą nuo pradžios iki pabaigos. Tokios sistemos siekia atitrūkti nuo iš anksto kietai užkoduotų (angl. *hardcoded* ) kintamųjų ir veiksmų. Pagrindinė keliama prielaida, kad turint pakankamai duomenų modelis turėtų netiesiogiai daryti išvadą apie tarimą ir taip pat turėtų būti pasiektas universalus modelis tinkantis ir pritaikomas skirtingą balsą ir kalbėjimo stilių turintiems žmonėms. Toks modelis taip pat supaprastintų duomenų paruošimą išvengdamas garso duomenų anotacijų sulygiavimo su įrašu [45], [46].

Vienas pirmųjų šiai problemai spręsti buvo pasiūlytas sprendimas pasitelkiant rekurentinius neuroninius tinklus (angl. *Recurrent Neural Network*), kuris ne tik supaprastino duomenų ir modelių paruošimą, nes tereikia anotuoto garsyno ir jį pasitelkiant apmokyti giliojo neuroninio tinklo modelį, bet ir buvo pasiekęs geriausią žodžių klaidos dažnį vertinimą atliekant su „Switchboard Hub5'00“ duomenų rinkiniu – 16 % (WER). Savo laiku parodė žymiai geresnius rezultatus ant triukšmingesnių duomenų rinkinių [1][22] (žr. **5 pav.**) pateikiami „DeepSpeech“, lentelėje žymimas „DS1”, ir „DeepSpeech 2“ lentelėje žymimas „DS2”, architektūrų žodžių klaidų dažnio rezultatai ir palyginami su žmogaus spėjimo tikslumu lyginant ant įvairių testavimo duomenų rinkinių.

| Read Speech | | | |
|---|---|---|---|
| Test set | DS1 | DS2 | Human |
| WSJ eval'92 | 4.94 | 3.60 | 5.03 |
| WSJ eval'93 | 6.94 | 4.98 | 8.08 |
| LibriSpeech test-clean | 7.89 | 5.33 | 5.83 |
| LibriSpeech test-other | 21.74 | 13.25 | 12.69 |

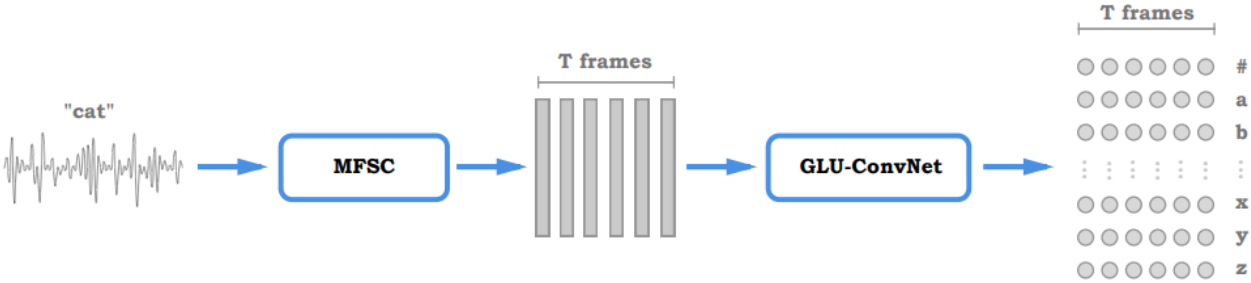
**5 pav.** „DeepSpeech“ ir „DeepSpeech 2“ architektūrų pavyzdžiai [12]

Taip pat yra variantų, naudojančių vienos dimensijos konvoliucinius tinklus („ConvNets“). Tokios architektūros galėtų būti „Wav2Letter“, kurios akustinis modelis yra grįstas standartiniu konvoliuciniu neuroniniu tinklu, kuriam paduodamos įvairios garso ypatybės išgautos naudojant „Connectionist Temporal Classification“ alternatyvą pavadinta „AutoSegCriterion“. Šios charakteristikos yra tokios kaip, kalbai specifiškos ypatybės – „Mel-Frequency Cepstrum Coefficients“ (MFCC), kurios dažnai randamos klasikinėse „GMM“ ar „HMM“ kalbos sistemose [2], galios spektras ir gryna garso banga. Galios spektro ypatybės panaudojimas yra įprasta technika pastarųjų metų giliojo mokymosi akustinių modelių kūrime [22]. Grynos bangos panaudojimas taip

pat minimas kituose to laiko straipsniuose [47]. Taip pat naudojamas spindulio paieškos modelis, kuris atlieka kalbos dekodavimo modelio funkciją, pateikdamas tikėtiniausią sakinio išvestį. Tokiu būdu kuriami modeliai pasiekia 7,2 % žodžių klaidos dažnio metriką skaičiuojant ant LibriSpeech test-clean [46]. Šio modelio architektūra skirta grynai garso bangai (žr. **6 pav.**). Pirmi du sluoksniai yra konvoliucijos su žingsniu. Paskutiniai du sluoksniai yra konvoliucijos su pločiu = 2, kuris yra lygus pilnai sujungtiems sluoksniams. Galios spektro ir „MFCC“ tinklai neturi pirmo sluoksnio.


**6 pav.** „Wav2Letter“ modelio architektūra skirta grynai garso bangai. [36]

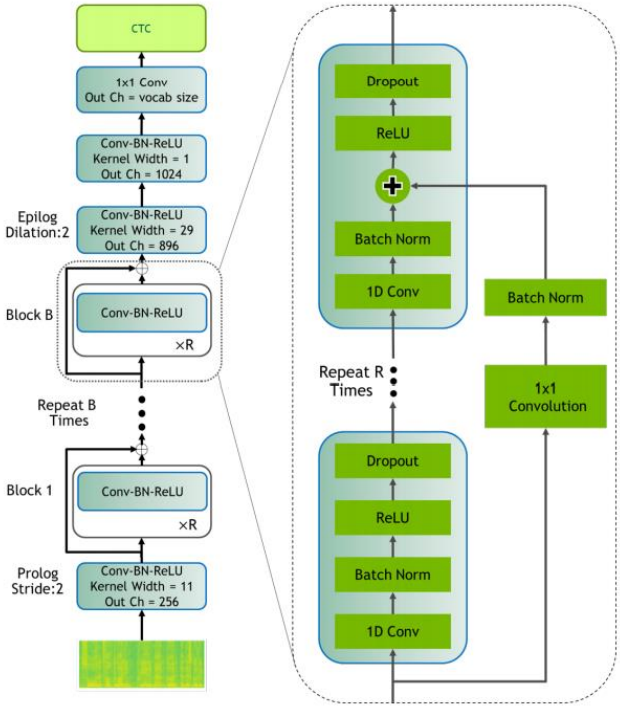
Panašiu principu yra sukurtas ir patobulintas algoritmas minimas „Wav2LetterV2“, kuris skiriasi nuo ankščiau minėto pirmos versijos konvoliucinio tinklo architektūroje padarytais pakeitimais. Vienas iš pakeitimų yra naudojimas „Gated Linear Unit“ funkcijos, kuri sumažina išnykstančio gradiento problemą giliose tinklų architektūrose suteikiant linijinį kelią gradientams išlaikant ne linijines savybes [48]. Taip pat išmetimo funkciją (angl. *Dropout*) kiekvieno sluoksnio, išskyrus paskutiniame išvesties sluoksnyje, aktyvacijose. Ši funkcija atsitiktinai išmeta neuronus iš tinklo sluoksnių taip gerindama modelio generalizavimą – apsimokymą su įvairesniais duomenimis, suardydama fiksuotą išmoktų ypatybių kombinaciją [49]. Galiausiai spindulio paieškos algoritmas yra naudojamas su kiekiu paremtu kalbos modeliu. Šie pagerinimai leidžia modeliui pasiekti 4,8 % žodžių klaidos dažnio metriką skaičiuojant su „LibriSpeech test-clean“ duomenų rinkiniu [45]. Šio modelio architektūra (žr. **7 pav**). Modelis apskaičiuoja „log-mel filterbanks“ (MFSC) ypatybes, kurios toliau yra perduodamos į uždarą konvoliucinį tinklą. Tuomet tinklas grąžina tikimybes kiekvienai raidei, kuri yra žodyne, ir kiekvienam įvesties požymiui. Kuomet yra vykdomas įrašo atpažinimas šie įverčiai yra perduodami dekodavimo modeliui, kuris suformuoja tikėtiniausią žodžių seką. Mokymo metu šie įverčiai yra perduodami CTC („Connectionist Temporal Classification“) arba ASG („AutoSegCriterion“) kriterijams, kurie skatina raidžių sekas, kurios nuveda iki reikiamo raidžių perteikimo iš garso takelio (kuris šiuo atveju yra „c a t“).




**7 pav.** Wav2LetterV2 modelio architektūra

Dar vienas automatinio kalbos atpažinimo architektūros pavyzdys pasiekiantis aukštus rezultatus yra „Jasper“. Šio modelio architektūra taip pat yra iš atpažinimą nuo pradžios iki pabaigos

atliekančių modelių, kurie pakeičia akustinį ir tarimo modelį į konvoliucinį neuroninį tinklą. „Jasper“ naudoja „Mel-Filterbank“ garso ypatybes išskaičiuotas iš 20ms lango su 10ms persidengimu ir grąžina tikimybių išsidėstymą kiekvienai raidei. Šiam metodui yra naudojama grupinė architektūra, Jasper BxR modelis turi B kiekį grupių ir R kiekį pogrupių (žr. **8 pav.**). Kiekvienas pogrupis atlieka tokias operacijas: vienos dimencijos konvoliuciją, partijos normalizaciją, ištaisytą tiesinę aktyvaciją (angl. *Rectified linear activation*, ReLu) ir išmetimo funkciją (angl. *Dropout*). Kiekvienos grupės įvestys yra tiesiogiai sujungiamos su paskutiniu pogrupiu per liekamojo ryšio jungtį. Liekamojo ryšio jungtis pirmą kartą yra projektuojama per 1x1 konvoliuciją, kad būtų atsižvelgta į skirtingą įvesties iš išvesties kanalų skaičių, tuomet perleidžiami per grupės normalizaciją (angl. *Batch normalization)*. Šios partijos normalizacijos išvestis pridedama prie paskutinio pogrupio partijos normalizacijos. Tada šio rezultato suma yra perleidžiama per „ReLu“ aktyvacijos funkciją ir išmetimo funkciją (angl. *Dropout)*, kad būtų gaunama esamos grupės išvestis. Šiuo sprendimu paremti kuriami modeliai pasiekia 3,86 % žodžių klaidos dažnio metriką skaičiuojant su „LibriSpeech test-clean“ duomenų rinkiniu, nenaudojant kalbai dekoduoti skirto modelio ir 2,95 %, naudojant spindulio paieškos dekodavimą panaudojant išorinį kalbos modelį [51].



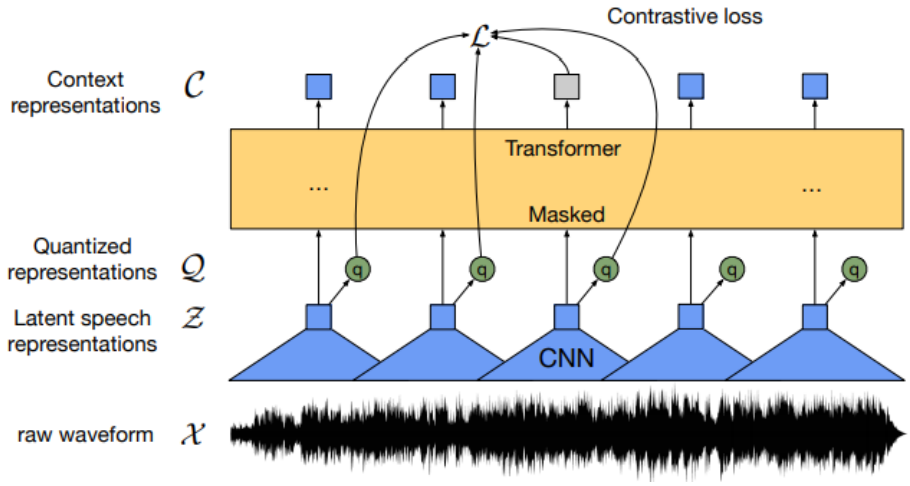

**8 pav.** „Jasper BxR“ modelio architektūra: B – grupių skaičius, R pogrupių skaičius.[41]

Šios naujos architektūros paremtos giliaisiais neuroniniais tinklais pagrindinis trūkumas, tai jog šiam metodui reikia labai daug anotuotų duomenų, kadangi neuroninis tinklas mokosi iš duomenyse esančių šablonų. Tačiau toks būdas nėra efektyvus ir sunkiai pritaikomas, norint pasiekti gerus rezultatus su įvairiomis kalbomis, kurių pasaulyje yra apie 7000 [41]. Taip pat mokymasis, kuomet naudojami anotuoti duomenys neatspindi žmonių mokymosi, kadangi naujagimiai mokosi klausydamiesi aplink juos esančių suaugusiųjų. Tokį mokymąsi, kuomet vaikai mokosi klausydamiesi galima pavadinti savarankišku mokymųsi. Savarankiškas ir pusiau savarankiškas mokymasis yra taikomas ir dirbtinio intelekto srityje sprendžiant natūralios kalbos apdorojimo problemas, kuomet yra iš anksto apmokomas giliojo mokymosi modelis su neanotuotais duomenimis,

toks problemos sprendimo būdas leidžia pasiekti geresnius rezultatus, nei mokant vien tik su mažesniu kiekiu anotuotų duomenų [42]. Taip pat pastaraisiais metais vis plačiau siekiama pritaikyti savarankišką ir pusiau savarankišką mokymąsi kompiuterinės regos užduotyse [43][44].

Savarankiško mokymo sprendimas labai palengvintų ir automatinio kalbos atpažinimo užduotį. Lyginant su anotuotais duomenimis neanotuoti kalbos duomenys yra lengvai prieinami. Tokius duomenis galima laisvai gauti internete iš įvairių vaizdo įrašų, radijo transliacijų įrašų, audio knygų. Panašiu principu buvo kuriamas ir kompanijos „Tilde“ naudojamas įrankis, kadangi jie savo modeliui sukurti lietuvių kalbos automatiniam kalbos atpažinimui pasitelkė Lietuvos Respublikos Seimo transliacijų vaizdo įrašus, tik kadangi jie naudojo realizaciją, kuriai reikėjo anotuotų duomenų, jie duomenis pirmiausia susianotavo su iš anksto apmokytu modeliu, paskui anotacijas pertikrino ir papildė duomenų rinkinį [5]. Tačiau dabar randasi architektūros, kurioms nėra reikalingi dideli kiekiai anotuotų duomenų. Tokiomis architektūromis kuriami modeliai yra iš anksto treniruojami dideliu kiekiu duomenų ir pasiekia vienus geriausių rezultatų (angl. *state of the art*). Tai yra pusiau prižiūrimos architektūros, kurioms apmokyti galima naudoti ne tik anotuotus duomenis, bet galima apmokyti modelį su dideliu kiekiu neanotuotų duomenų ir toliau patikslinti mokymą su sąlyginai nedideliu kiekiu anotuotų duomenų. Tokios architektūros naudojant angliškus duomenų rinkinius su 10 min anotuotų duomenų ir 53 tūkst. valandų neanotuotų duomenų „Wav2Vec 2.0“ architektūra apmokytas modelis pasiekia 4.8 %/8.2 % žodžių klaidos dažnio metriką atitinkamai su „Librispeech test clean/other“ duomenų rinkiniais, o geriausias pasiektas rezultatas, naudojant visus anotuotus „Librispeech“ duomenis buvo 1,8 %/4,1 % atitinkamai su „Librispeech test clean/other“ duomenų rinkiniais [14].

„Wav2vec 2.0“ architektūra susideda iš grynos garso bangos pavertimo į klasifikavimo išvestį (jeigu apmokyta ne tik su pradiniais duomenimis, bet ir apmokyta su anotuotais duomenimis) arba konteksto reprezentacija (jeigu apmokyta tik su pradiniais, neanotuotais duomenimis). Garso banga pirmiausiai yra paduodama vienos dimensijos konvoliuciniam tinklui. Šis tinklas turi 7 grupes branduolių, kurie grąžina 512 dydžio ypatybių žemėlapį, kiekvienam sluoksniui. Šio konvoliucinio tinklo išvestis yra vadinama nematomu kalbos atvaizdavimu (angl. *Latent speech representation*), abstraktesnis garso bangos atvaizdavimas. Taip pat naudojamos „GeLu“ (angl. *Gaussian error linear unit*) aktyvacija ir „LayerNorm“ sluoksnio normalizavimas, kad garso bangos įvesties vidurkis būtų nulis ir prioretizuojama vieneto dispersija. Tuomet konvoliucinio tinklo išvestys yra praleidžiamos per kvantizavimo fragmentą, kuris parenka artimiausią diskretų vektoriaus atvaizdą, nenutrūkstančiam vektoriui, kad būtų sumažintas galimų vektorių rinkinys. Taip pat konvoliucinio tinklo išvestys yra naudojamos kaip įvestys transformaciniam modeliui (ang. *Transformer model*), kuris yra vadinamas konteksto tinklu (angl. *Context network*), generuojantis kontekstinius atvaizdus (angl. *Context representation*) [13]. Kuomet vykdomas pradinio mokymo etapas, paduodant modeliui neanotuotus duomenis, dalis įvesčių į transformacinį modelį yra uždangstomos, kad atlikti uždangstytos kalbos modeliavimo užduotį (angl. *Masked Language Model task*, MLM), kuri padeda modeliui geriau prisitaikyti mokymo sakiniams [52]. Šio modelio architektūra (žr. **9 pav.**).




**9 pav.** Wav2Vec 2.0 modelio architektūra [13]

**1 lentelė** Rezultatų su Librispeech test/clean duomenų rinkinių tarp peržvelgtų architektūrų

| Architektūra | Rezultatai su Librispeech test/clean | Komentaras |
|---|---|---|
| Deep speech 1 | 7,89 % | Reikalingas didelis kiekis anotuotų duomenų (kuo daugiau tuo geriau) |
| Deep speech 2 | 5,33 % | Reikalingas didelis kiekis anotuotų duomenų (kuo daugiau tuo geriau) |
| Wav2Letter | 7,2 % | Reikalingas didelis kiekis anotuotų duomenų (kuo daugiau tuo geriau) |
| Wav2LetterV2 | 4,8 % | Reikalingas didelis kiekis anotuotų duomenų (kuo daugiau tuo geriau) |
| Jasper | 2,95 % | Reikalingas didelis kiekis anotuotų duomenų (kuo daugiau tuo geriau) |
| Wav2Vec 2.0 | 1,8 % | Reikalingas didelis kiekis neanotuotų duomenų, taip pat reikalingas nedidelis kiekis anotuotų duomenų. |

Palyginus architektūras (žr. **1 lentelė**) galime matyti, jog geriausias sprendimas, nereikalaujantis didelio kiekio anotuotų duomenų, nes galima atlikti išankstinį modelio apmokymą su neanotuotais duomenimis, yra „Wav2Vec 2.0“, kuris ant „Librispeech test/clean“ duomenų rinkinio pasiekia 1,8 % žodžių klaidų dažnį. Kadangi tyrime naudojamų duomenų, anotuotų duomenų kiekis yra palyginus nedidelis, tačiau yra užtektinai įprastų skambučių įrašų, ši architektūra atrodo optimali sukurti balso skambučių atpažinimo modelį. Taip pat visose architektūrose siekiant pagerinti modelio tikslumą yra naudojami dekodavimo ir kalbos modeliai, kurie patikslina atpažintus žodžius ir jų išsidėstymą sakiniuose.

## 1.6. Esami karkasai ir įrankiai garso atpažinimui

Didelės kompanijos, siekdamos supaprastinti ir pagreitinti giliųjų tinklų kūrimą ir modelių eksperimentavimą, kuria giliojo mokymosi karkasus. Įrankiai kuriuos galima paminėti yra „Google“ kompanijos plėtojamas „TensorFlow“, „Facebook“ kompanijos „PyTorch“, „Apache“ kompanijos „MxNet“ ir daugelis kitų. Šie karkasai paspartina darbą, kadangi turi bibliotekas, kurios supaprastina darbą su dideliais duomenų kiekiais, giliųjų tinklų kūrimą, modelių vertinimą, vaizdo plokščių panaudojimą suteikiant paprastesnę aukštesnio lygio programavimo sąsają su „CUDA“ paralelinių

skaičiavimų platforma [26]. Populiariausi karkasai šiuo metu yra „Google“ kompanijos „TensorFlow“, bei „Facebook“ kompanijos „PyTorch“. „TensorFlow“ populiarumas kyla iš pritaikomumo produkcijoje, kadangi turi modelių konvertavimą, kad šie būtų pritaikomi mobiliuosiuose įrenginiuose, taip pat buvo orientuojamas skaičiavimų greitumą nuo pat karkaso kūrimo pradžios [27][28]. O „Pytorch“ karkasas pamėgtas mokslininkų bendruomenės, kadangi pastaraisiais metais mokslinėse konferencijose pateikiamų straipsnių naudojant „PyTorch“ karkasą pranoko straipsnių skaičių, kuriuose buvo naudojamas „TensorFlow“ karkasas [28].

**2 lentelė** Neuroninių tinklų karkasų palyginimas [16]

| **Karkasas** | **Sukurtas naudojant programavimo kalbą** | **Galimos kalbos programavimui** | **Populiarumas [28][29]** |
|---|---|---|---|
| TensorFlow | C++, Python | Python, C++, Java, Go | Aukštas ir dar vis populiarėjantis |
| PyTorch | C, Python | Python, Java | Aukštas ir dar vis populerėjantis |

Karkasų palyginimas (žr. **2 lentelė**). Abu karkasai yra patogūs eksperimentavimui, kadangi galima naudoti su „Python“ programavimo kalbos sąsaja, kuri yra viena populiariausių ir intuityviausių kalbų dėl savo paprastumo ir patikimumo [30]. „Python“ programavimo kalba rašomas kodas yra glaustas, efektyvus, lengvai suprantamas, bei tvarkomas [31].

Minėtų „TensorFlow“ ir „PyTorch“ karkasų panaudojimas yra platus. Nuo neuroninių tinklų kūrimo nuo nulio iki jau sukurtų specifinėms užduotims panaudojamų bibliotekų. Šių bibliotekų, kurios yra paremtos neuroniniais tinklais, paskirtis yra įvairi. Viena iš krypčių yra vaizdo apdorojimas: vaizdų klasifikavimas, objektų atpažinimas, segmentavimas [32]. Kita krypti – užduotys skirtos įveikti natūralios kalbos apdorojimo užduotis tokias, kaip teksto ir kalbos apdorojimas, morfologinė analizė ir panašios [33][34]. Tarp natūralios kalbos apdorojimo bibliotekų yra ir bibliotekos skirtos apdoroti garso failus ir realizuoti šnekos apdorojimą, kuris leidžia apmokyti modelius skirtus šneką paversti tekstu [35][36]. Šios funkcijos dar labiau palengvinamos specifiškoms užduotims skirtuose įrankių rinkiniuose, specialiai automatiniam kalbos apdorojimui. Vienas iš „TensorFlow“ karkasu grįstų įrankių yra „DeepSpeech“ modelio apmokymui pritaikytas „Mozilla DeepSpeech“ įrankių rinkinys, kuris supaprastina duomenų augmentacijos ir rekurentiniu neuroniniu tinklu grįsto akustinio modelio apmokymo užduotis [37]. Taip pat dar vienas „TensorFlow“ karkasu grįstas įrankis yra „OpenSeq2Seq”, kuris pritaikytas apmokyti platesnį spektrą modelių nei „Mozilla DeepSpeech“, kuris yra pritaikytas tik vieno tipo modeliui. „OpenSeq2Seq” palaiko tris kalbos atpažinimo modelių tipus [38]. Ir taip pat kaip ir prieš tai minėtas turi duomenų augmentacijos biblioteką. Iš automatinio kalbos atpažinimo įrankių paremtų „PyTorch“ karkasu vieni populiariausių yra šie – „Kaldi PyTorch“ [39], bei „PyTorch Fairsec“ [40][23]. Abu šie įrankiai nusileidžia minėtiems „TensorFlow“ karkasams, kadangi neturi funkcijų tiesiogiai pritaikytų duomenų augmentacijai. Tačiau su „PyTorch Fairsec“ galima apmokyti „wav2vec” modelį, kuris pasiekia konkurencingą žodžių klaidos dažnį su angliškų testavimo „LibriSpeech clean/other“ duomenų rinkiniu 1.8 %/3.3 %, šiuos rezultatus modelis pasiekia pirmiausia jį apmokant ant neanotuotų duomenų ir paskui patikslinant apmokymą su anotuotais duomenimis [23].

**3 lentelė** Automatinės kalbos atpažinimo įrankių vertinimas

| Įrankis | Karkasas | „GitHub“ žvaigždės | Palaikomų modelių skaičius | Duomenų augmentacija | Pasileidimo paprastumas |
|---|---|---|---|---|---|
| „Mozilla DeepSpeech“ | TensorFlow | 16.3 tūkst. | 1 | Yra | Paprastas, yra išsami dokumentacija |
| OpenSeq2Seq | TensorFlow | 1.4 tūkst. | 3 | Yra | Paprastas, yra išsami dokumentacija |
| Kaldi PyTorch | PyTorch | 1.9 tūkst. | 7 | Reikia naudoti PyTorch įrankius | Paprastas, yra išsami dokumentacija |
| Pytorch Fairsec | PyTorch | 11 tūkst. | 3 (tipai) | Reikia naudoti PyTorch įrankius | Vidutiniškas, dokumentacijos nėra daug, visi pavyzdžiai naudojant anglų kalbos alfabetą. |
| Hugging face | PyTorch/TensorFlow | 46.9 tūkst. | 60 (įskaitant ne tik ASR, taip pat turi Wav2Vec 2.0 modelį) | Reikia naudoti PyTorch įrankius | Paprastas, yra išsami dokumentacija |

Iš automatinių kalbos atpažinimo įrankių vertinimo lentelės (žr. **3 lentelė**) galime matyti, jog populiariausi įrankiai pagal „GitHub“ žvaigždžių skaičių yra „Hugging face“ su 46.9 tūkst. žvaigždžių ir „PyTorch Fairsec“ su 11 tūkst. žvaigždžių. Didžiausią modelių tipų skaičių turi „Hugging face“ įrankis palaikantis tiek „TensorFlow“ tiek „PyTorch“ karkasus. „Kaldi PyTorch“ įrankis turi 7 skirtingus modelių tipus ir realizuotas pasitelkiant „PyTorch“ karkasą, o iš „TensorFlow“ karkaso „OpenSeq2Seq” įrankis turi 3 modelių tipus. Realizuotą įrankyje įvesties duomenų augmentaciją turi tik „Tensorflow“ karkaso sprendimai. Galiausiai visi įrankiai yra gerai dokumentuoti, todėl pasileidžiant juos neturėtų kilti problemų.

Iš apžvelgtų karkasų ir priemonių daugiausia vilčių teikia „Hugging face“ biblioteka naudojanti „PyTorch“ karkasą, kadangi šis įrankis yra plačiai naudojamas ir palaiko „Wav2Vec 2.0“ architektūrą. Taigi naudojant minėtą realizaciją galimą būtų apmokyti architektūrą panaudojant išankstinį apmokymą su dideliu kiekiu neanotuotų duomenų ir patikslinti apmokymą su anotuotais duomenimis.

### 1.7. „Facebook“ kompanijos apmokyti modeliai, kurie gali būti pritaikyti telefoninės šnekos atpažinimui

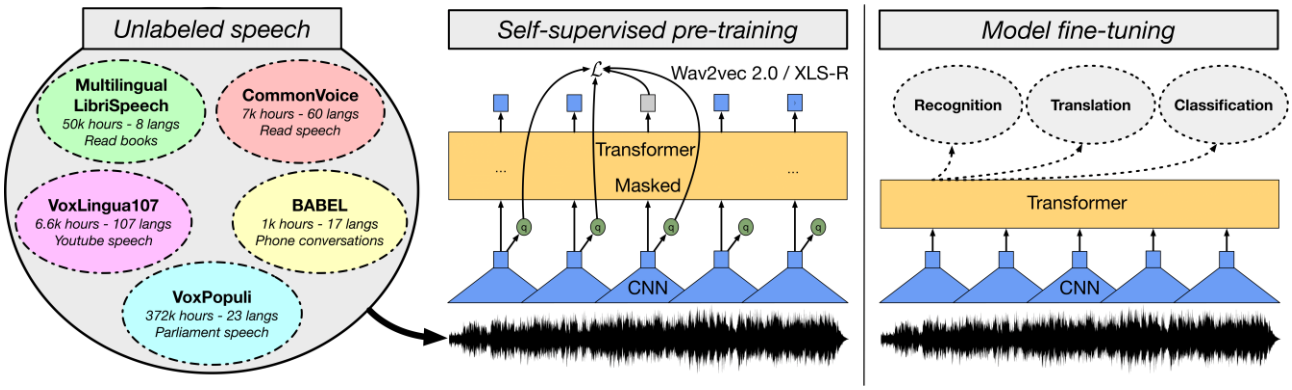
„Hugging face“ modelių bibliotekoje, galima pasirinkti iš anksto apmokytą modelį ir jį pritaikyti savo duomenų rinkiniui. Kas palengvina darbą ir pagreitina rezultato gavimą, kadangi nereikia apmokyti savaime apsimokančio modelio su dideliu kiekiu neanotuotų duomenų, užtenka paimti mažesnę dalį anotuotų duomenų, pagal kuriuos modelis toliau apsimoko, nuo iš anksto apmokyto ir užšaldyto modelio. Pasitelkiant tokio tipo modelius, pasiruošti apmokymui reikia tik duomenis, bei susikurti žodyną ir teksto skirstytoją (angl. *tokenizer*), kuris tekstą skaido į mažesnes leksemas (angl. *token*).

### 1.7.1. „Wav2Vec2 Base“ modelis

Modelis yra apmokytas ant 960 valandų neanotuoto garsyno ir yra sukurtas pagal originalų „Wav2Vec2“ mokslinį straipsnį [23].

### 1.7.2. „Wav2Vec2 XLS-R“ modelis

Šis modelis yra apmokytas su 436 tūkstančiais valandų neanotuotų duomenų iš 128 kalbų. Šie duomenys buvo paimti iš tokių duomenų rinkinių, kaip „Multilingual LibriSpeech“, „Common Voice“, „VoxLingua107“, „Babel“, „VoxPopuli“. Šiuo modeliu siekiama palengvinti modelio pritaikymą kalboms, kurios neturi didelio kiekio šnekos duomenų rinkinių, pirmiausia apmokant modelį ant didelio kiekio neanotuotų šnekos įrašų duomenų rinkinių, o vėliau apmokyti modelį nuo užšaldyto atskaitos taško (angl. checkpoint) panaudojant pasirinktos kalbos mažesnį pasirinktos kalbos duomenų rinkinį, taip pritaikant modelį minėtai kalbai [54]. Modelio architektūra (žr. **10 pav.**).

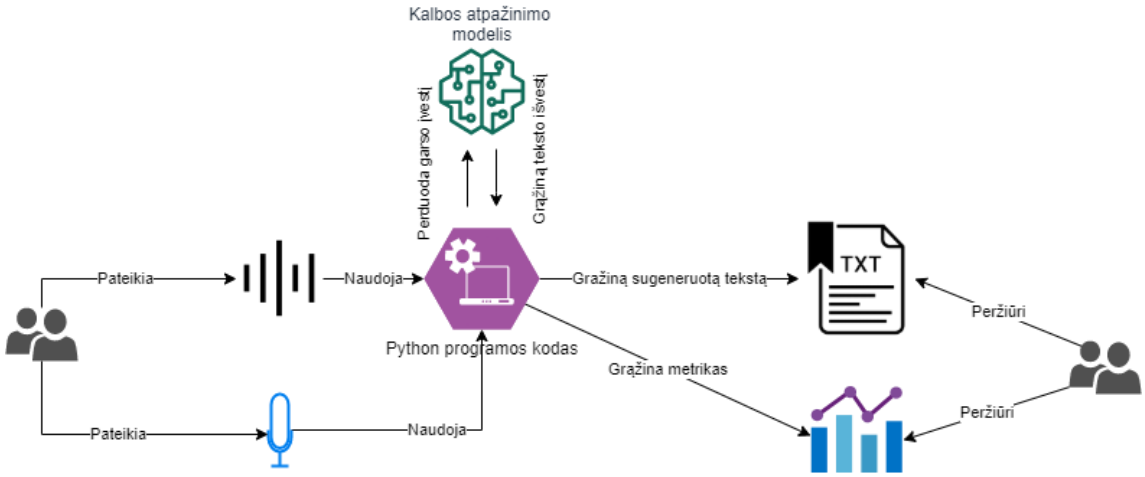



**10 pav.** Wav2Vec2 XLS-R architektūra [53][54]

# 2. Sistemos projektavimas

## 2.1. Sistemos koncepcija

Sistemos prototipo, kuri bus skirta atlikti šnekos atpažinimo tikslumo tyrimus su lietuviško telefoninių įrašų garsynu (žr. **11 pav.**). Kaip pavaizduota koncepcijoje sistema naudos „Python“ programavimo kalbos skriptą, kuris priims kaip įvestį arba garso įrašą (gali priimti ir garso įrašų partiją, taip pat jeigu norimas pamatuoti modelio tikslumas sistema taip pat galės priimti garso įrašų tekstines anotacijas) arba tiesioginę mikrofono įvestį. Tuomet ši garso įvestis bus praleidžiama per kalbos atpažinimo modelį, kuris grąžins į sistemą garso įvestį paverstą tekstine išraiška. Tuomet pradinis skriptas grąžins tekstinę išraišką ir/arba atpažinimo tikslumo metrikas. Tekstas vartotojui bus grąžinamas sukuriant sistemoje, garso failams sugeneruotą „.csv“ failą (su kompiuterio nuoroda į garso failą, bei transkripcija), taip pat „.csv“ failą su sugeneruotomis tikslumo vertinimo metrikomis. Kuriuos vartotojas galės peržiūrėti.




**11 pav.** Kuriamos sistemos koncepcija

## 2.2. Funkciniai reikalavimas

Šiame poskyryje yra pateikiami reikalavimai sistemai perteikiami funkciniai reikalavimai.

- Vartotojas turi turėti galimybę pateikti kalbos garso įrašą.
- Vartotojas turi turėti galimybę kaip garso įvestį naudoti mikrofoną prijungtą prie kompiuterio.
- Vartotojas turi galėti paleisti sistemą su pasirinktais parametrais.
- Vartotojas turi galėti pateikti garso įrašų duomenis su anotacijomis modelio apmokymui.
- Vartotojas turi galėti pateikti garso įrašų duomenis su anotacijomis modelio metrikų vertinimui.
- Sistema pateikus garso įrašą turi grąžinti atpažintą kalbą rišlaus teksto išraiška.
- Sistema gavus įvestį iš mikrofono turi grąžinti atpažintą kalbą rišlaus teksto išraiška.
- Sistema turi leisti vartotojui apmokyti kalbos atpažinimo modelį su savo pateiktais duomenimis.
- Sistema turi grąžinti pamatuotas metrikas kalbos rinkiniui su anotacijomis.

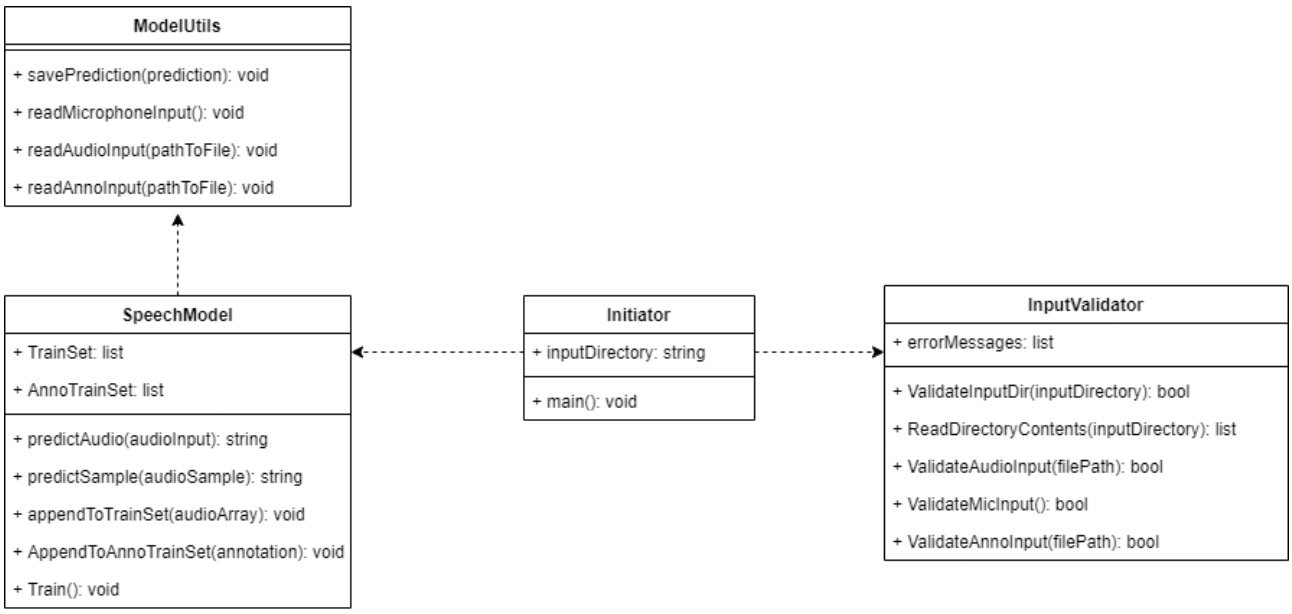
## 2.3. Nefunkciniai reikalavimai

Šiame poskyryje yra pateikiami reikalavimai sistemai perteikiami nefunkciniai reikalavimai.

- Sistema turi veikti su lietuvių kalbos telefoninių skambučių balso įrašais.
- Tikslumas – matuojamas žodžių klaidos dažniu. Siekiama šios metrikos išraišką ant telefoninių skambučių balso įrašų <60 % žodžių klaidos dažnis.
- Ankščiau minėti tikslumo rezultatai turi būti pasiekti naudojant nedidelį kiekį anotuotų duomenų <100 valandų. Neanotuotų duomenų panaudojimas nėra ribojamas.
- Sistema turi būti naši – naudojant realizaciją su vaizdo plokšte geresne nei NVIDIA RTX2070m, kalbos garso takelis turi būti perteikiamas tekstu greičiau nei per 1.5*garso įrašo ilgis. Jeigu garso įrašo ilgis yra 10 sekundžių sistema perteikti tekstu šį įrašą turėtų per trumpiau nei15 sekundžių.
- Sistema turi veikti „Ubuntu“ > 20.04 operacinėje sistemoje.
- Sistema turi būti paleidžiama naudojant „Ubuntu“ sistemos terminalą, neturi būti sukurta grafinė vartotojo sąsaja.
- Sistemos nustatymai turi būti įvedami per „Ubuntu“ sistemos terminalą paleidžiant programą.
- Sistema turi priimti „.wav“ garso failus.
- Sistema turi grąžinti įvertintas žodžių klaidos dažnio ir raidžių klaidos dažnio metrikas, jeigu pateikiami anotuoti duomenys garso įrašams ir nustatomas kalbos vertinimo argumentas.
- Kalbos įrašas paverstas į tekstą turi būti išsaugomas „.txt“ formatu.
- Vertinant kalbos metrikas įvertinimai turi būti grąžinti „.csv“ failo formatu.
- Vertinant kalbos metrikas į failą įrašomi šie duomenys kiekvienam tiriamam garso įrašui - garso įrašo failo talpinimo kompiuteryje vieta, žodžių klaidos dažnio metrika, raidžių klaidos dažnio metrika.
- Nepavykus apdoroti garso failo sistema turi išvesti klaidą, bet nenutraukti tolimesnio garso failų apdorojimo.
- Sistema turi turėti argumentą, kurį įvedus vartotojui būtų pateikiama informacija apie galimus naudoti argumentus.
- Sistemai anotacijos pateikiamos „.txt“ formatu.

## 2.4. Klasių diagrama

Klasių diagrama (žr. **12 pav.**), joje yra 4 pagrindinės klasės skirtos atspindėti sistemoje vykstančius veiksmus. Klasė „Initiator“ yra atsakinga už skirtingų metodų paleidimą. Klasė „InputValidator“ skirta patikrinti įvesties duomenų tinkamumą tolimesniam veiksmų atlikimui. Klasė „SpeechModel“ yra skirta atlikti veiksmus su modeliu, tokius kaip garso įrašo atpažinimas, mikrofono garso pavyzdžio atpažinimas, garso įrašų bei anotacijų sudėjimas į mokymo imtis ir apmokymas. Klasė „ModelUtils“ yra pagalbinė „SpeechModel“ klasė, kuri atlieka tokius veiksmus, kaip spėjimų saugojimas ir įvesties nuskaitymas.




**12 pav.** Klasių diagrama

## 2.5. Aparatūrinė posistemė

Aparatūrinės posistemės pagrindinis komponentas yra vartotojo naudojamas kompiuteris, kuris bus naudojamas sistemos skripto paleidimui. Kompiuteris turi turėti įdiegtą „Ubuntu 20.04“ operacinę sistemą, taip pat įdiegtą „Python 3.8.0“ arba aukštesnės versijos „Python“ programavimo kalbos paketą, kad sistema veiktų greitai ir išnaudotų grafikos apdorojimo bloką (angl. *graphics processing unit,* GPU) turi būti įdiegtas „CUDA“ įrankių rinkinys su versija naujesne, nei 10.1, taip pat yra reikalinga „cuDNN“ biblioteka, bibliotekos versija turi būti pasirinkta pagal įdiegiama „CUDA“ versiją.

**4 lentelė** Aparatūrinės posistemės serverio reikalavimai

| **Minimalūs kompiuterio komponentų reikalavimai** | **Rekomenduojami kompiuterio komponentų reikalavimai** |
|---|---|
| 16 GB RAM atminties<br>6x2.6 GHz procesorius<br>1TB vidinės atminties<br>RTX 2070m | 64 GB RAM atminties<br>8x4.2 GHz procesorius<br>2TB vidinės atminties<br>2xRTX2080 |

Keliami reikalavimai aparatūrinei posistemei aprašyti lentelėje (žr. **4 lentelė**). Dideli vidinės atminties reikalavimai yra dėl modelių apmokymui naudojamo didelio kiekio duomenų, todėl reikia šiuos duomenis turėti, kur išsaugoti ir užtikrinti jų pasiekiamumą mokymo ir atpažinimo metu. Vidinės atminties reikalavimai gali kisti nuo to kiek duomenų yra naudojama modelio apmokymui arba atpažinimui. Taip pat reikalavimai atsižvelgia ir laisvosios kreipties atmintį (angl. *random access memory*, RAM), kadangi apmokant modelį siekiant pagreitinti apmokymą dalis duomenų yra įrašoma arba į vaizdo procesoriaus arba į pačio kompiuterio laisvosios kreipties atmintį, todėl didesnis kiekis leidžia greičiau apmokyti modelį, nes galima dirbti su didesnėmis partijomis (angl. *batch*). Siekiant sumažinti apmokymo ir atpažinimo laiką turi būti naudojamas pakankamai galingas vaizdo procesorius, todėl turi būti naudojama vaizdo plokštė bent „RTX2070m“.

## 2.6. Sprendimo kūrimo metodai ir priemonės

Sprendimui įgyvendinti bus pasitelkta „Python“ programavimo kalba. Kuri leidžia patogiai integruoti „Huggin Face“ modelių biblioteką reikalingą modelių mokymams ir spėjimams daryti. Duomenų ir modelių užkrovimui bei modifikacijoms, skaičiavimų perdavimui į „CUDA“ naudojamas „PyTorch“ karkasas, „Librosa“ biblioteka.

## 3. Eksperimentai su modeliais pasitelkiant „Hugging face“ karkasą su „Babel“ anotuotu duomenų rinkiniu

Siekiant apskaičiuoti tikslumą su lietuviškos telefoninės šnekos duomenų rinkiniu bus pritaikyti du šnekos atpažinimo modeliai. Vienas, pradinis kompanijos „Facebook“ „Wav2Vec2 Base“ modelis [23]. Antras modelis taip pat „Facebook“ kompanijos sukurtas modelis „Wav2Vec2 XLS-R 300m“ 300 milijonų parametrų modelis [54], kuris yra skirtas apmokyti kalbas nesiejant su viena kalba, modelio pradinis duomenų rinkinys sudarytas iš didelio kiekio skirtingų kalbų. Šie modeliai yra prieinami atsisiųsti ir toliau nuo užšaldyto (angl. *frozen*) modelio galimas tolimesnis modelio parametrų pritaikymas pasirinktam duomenų rinkiniui.

Tolimesniam mokymui (angl. *fine-tuning*) bus naudojamas „Babel“ anuotuotų lietuviškos šnekos telefoninių skambučių duomenų rinkinys.

Siekiant, kad šnekos atpažinimo modelis galėtų būti apmokytas, turi būti sukurtas duomenų rinkinio žodyno. Kadangi jungiamąja laiko klasifikacija (angl. *connectionist temporal classification*) įprasta klasifikuoti atskiras raides, todėl leksemos bus sudarytos iš atskirų raidžių, kurios bus sukurtos iš mokymo duomenų rinkinio anotacijų.

Kalba yra tolydus signalas, todėl siekiant, kad šį signalą galėtų apdoroti kompiuteris, jis pirma turi būti diskretus, toks signalo transformavimas vadinamas diskretizavimu (angl. *sampling*). Todėl svarbu atsižvelgti į įrašų diskretizavimo dažnį, kadangi šis parodo, kiek duomenų taškų bus išmatuojama kiekvieną sekundę kalbos signale. Todėl kuo didesnis diskretizavimo dažnis, tuo labiau garso įrašas atspindi tikrą kalbą, tačiau reikalauja ir daugiau duomenų reikšmių kiekvieną sekundę [55]. Užšaldytas modelis reikalauja, kad tolimesnių duomenų rinkinio diskretizavimo dažnis būtų identiškas arba labai panašus, tam ant kurio buvo apmokytas iki modelį užšaldant atskaitos taške. Kadangi kompiuteriui apdorojant garso signalą, pavyzdžiui signalas, kuris turės dvigubai didesnį diskretizavimo dažnį atrodys dvigubai ilgesnis laiku. Tiek „Wav2Vec2 base“, tiek „Wav2Vec2 XLS-R 300m“ modelis buvo apmokytas naudojant 16kHz diskretizavimo dažnį, todėl pritaikant šių modelius „Babel“ duomenų rinkiniui, kurio diskretizavimo dažnis yra 8kHz, reikės du kartus padidinti diskretizavimo dažnį. Siekiant iš signalo išskirti ypatybes bus pasitelktas ypatybių traukėjas (angl. *feature extractor*).

Pritaikant modelių parametrus „Babel“ duomenų rinkiniui modelių tikslumas mokymo metu bus matuojamas mokymo (angl. *training*) duomenų rinkinio nuostolio (angl. *loss*) funkcija, įvertinimo (angl. *evaluation*) duomenų rinkinio nuostolio funkcija, bei skaičiuojant įvertinimo duomenų rinkinio žodžio klaidos dažnio metrika.

Dėl telefoninės šnekos sudėtingumo, gali būti įnešamas triukšmas kalbos atpažinime, kurios gali privesti prie klaidingo šnekos į tekstą vertimo. Todėl papildomai bus bandomas būdas, kaip galima nerišlų žodį susieti su žodžiu iš žodyno, taip pagerinant spėjimo kokybę ir sumažinant žodžio bei ženklų klaidos dažnio metriką.

## 3.1. Tyrimo planas

- Paruošti „Wav2Vec2 base“ modelį ir išmatuoti klaidos metrikas.
- Paruošti „Wav2Vec2 XLS-R 300m“ modelį ir išmatuoti klaidos metrikas.
- Pasitelkti „Google Speech to Text API“ siekiant išmatuoti paruoštų modelių tikslumą egzistuojančiais komerciniais garso į tekstą vertimo sprendimais.
- Paruošti rašybos klaidų taisymo algoritmą, kuris padėtų pagerinti klaidos metrikas apmokytiems šnekos atpažinimo modeliams.
- Išmatuoti klaidos metrikas apmokytiems šnekos atpažinimo modeliams pridedant klaidų taisymo algoritmą.

## 3.2. Aparatūrinė įranga

Pirminiams mokymams ir garso įrašų testavimo rinkinio vertinimui buvo panaudotas nešiojamas kompiuteris pasižymintis vidutiniais techniniais parametrais būdingiems šių dienų nešiojamiems kompiuteriams (žiūrėti **5 lentelė**).

**5 lentelė** Mokymo aplinkos kompiuterio techniniai parametrai

| **Parametro pavadinimas** | **Parametro vertė** |
|---|---|
| Kompiuterio pavadinimas | MSI GE75 Raider 9SF |
| Procesorius | Intel(R) Core(TM) i7-9750H |
| Procesoriaus dažnis | 2,60GHz – 4,50GHz |
| Operatyvioji atmintis | 16GB DDR4-2666 |
| Operacinė sistema | Ubuntu 20.04 |
| Vaizdo plokštė | Nvidia RTX2070 (mobile) |
| NVIDIA Compute Capability | 7,5 |
| Vaizdo plokštės operatyvioji atmintis | 8GB |
| Vaizdo plokštės tranzistorių kiekis | 10,8 milijardo |

Siekiant paspartinti mokymus tolimesni mokymo buvo atliekami pasitelkiant galingesnį kompiuterį (žiūrėti **6 lentelė**).

**6 lentelė** Galingesnio mokymo kompiuterio techniniai parametrai

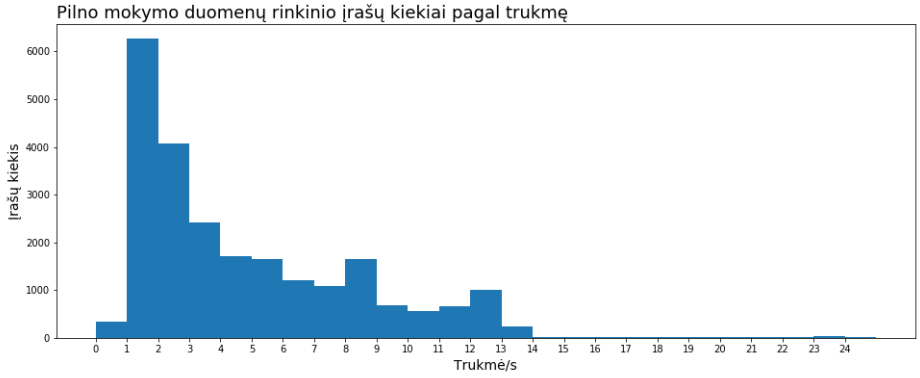
| **Parametro pavadinimas** | **Parametro vertė** |
|---|---|
| Procesorius | AMD EPYC 7313 |
| Procesoriaus dažnis | 3GHz – 3,7GHz |
| Operatyvioji atmintis | 64GB DDR4-2666 |
| Operacinė sistema | Ubuntu 20.04 |
| Vaizdo plokštė | Nvidia A100 |
| NVIDIA Compute Capability | 8 |
| Vaizdo plokštės operatyvioji atmintis | 40GB |
| Vaizdo plokštės tranzistorių kiekis | 54 milijardai |

## 3.3. Tyrimo eiga ir metrikos

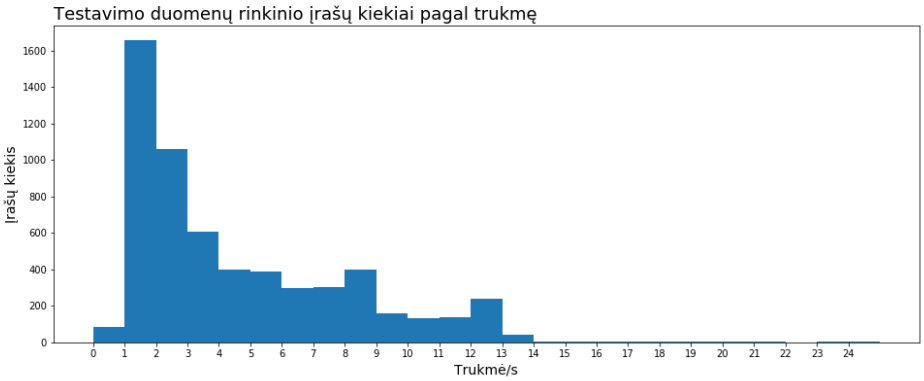
### 3.3.1. Duomenų paruošimas

Modelių apmokymui buvo pasirinktas „IARPA Babel“ telefoninės kalbos įrašų duomenų rinkinys. Rinkinyje yra kalbama lietuvių, aukštaičių ir žemaičių dialektu. Tarp įrašų lyčių pasiskirstymas yra maždaug vienodas. Kalbančiųjų amžius yra tarp 16 ir 71 metų. Skambučių įrašai yra padaryti su įvairių tipų telefonais (pvz., mobiliaisiais, laidiniais) iš įvairių aplinkų, gatvės, namų ar ofisų, viešų vietų, automobilio vidaus. Garso duomenų įrašai yra 8 kHz diskretizavimo dažnio ir turi „WAV“ (8 bitų „a-law“) koduotę. Transkripcijos yra pateikiamos „.txt“ failais ir yra „UTF-8“ koduotės [56].

Pradiniam „Babel“ duomenų rinkiny lietuviškų telefoninių skambučių garso įrašų yra ~36 valandos. Šis pradinis duomenų rinkinys buvo išskaidytas į treniravimo bei testavimo imtis. Išskaidymas buvo atliekamas pseudoatsitiktiniu būdu, išmaišant garso įrašus su atsijojimo parametru (angl. random seed) ir išskiriant 80 % duomenų imties į treniravimo imtį ir likusius 20 % į testavimo duomenų imtį. Taip treniravimo duomenų imty gavosi ~29 valandos, o testavimo duomenų imty gavosi ~7 valandos garso įrašų. Garso įrašų kiekio pasiskirstymą pagal laiką treniravimo duomenų imty (žr. **13 pav.**), testavimo duomenų imty (žr. **14 pav.**). Treniravimo imtį sudaro 174703 , o testavimo 42176 žodžiai.

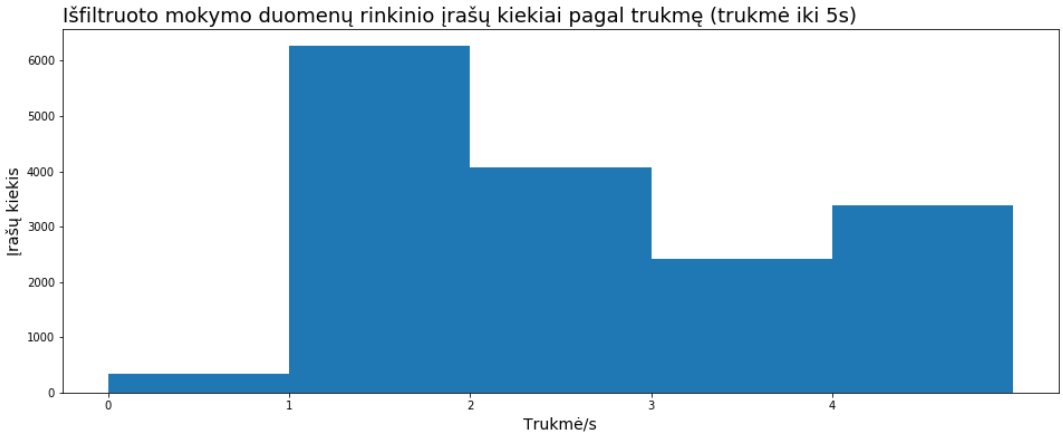



**13 pav.** Nefiltruoto mokymo duomenų rinkinio garso įrašų kiekio pasiskirstymo grafikas pagal trukmę




**14 pav.** Testavimo duomenų rinkinio garso įrašų kiekio pasiskirstymo grafikas pagal trukmę

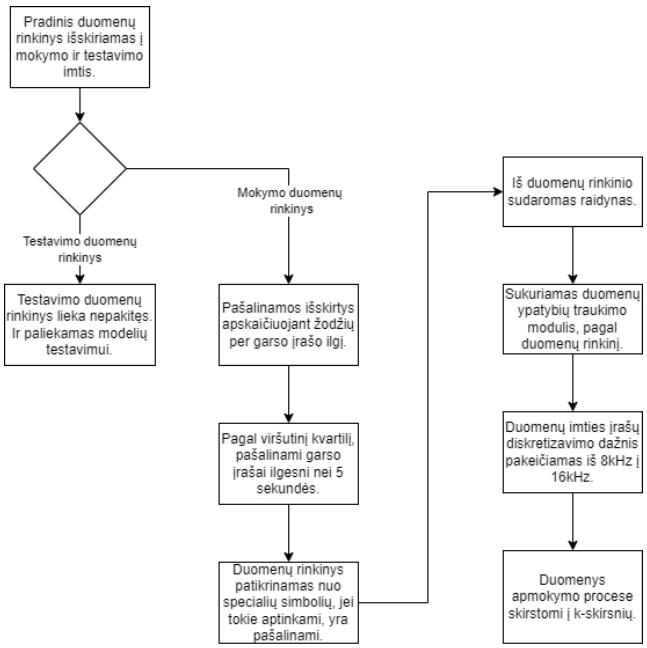
Siekiant optimizuoti mokymo procesą, iš mokymo imties buvo pašalintos išskirtys. Išskirtys buvo nustatomos apskaičiuojant žodžių kiekį ir susiejant su garso įrašo ilgiu. Taip buvo išskaičiuotas pasakomų žodžių dažnis per laiko tarpą. Radus pasakomų žodžių dažnį per laiko tarpą, galima buvo rasti įrašus, kurie būtų arba per ilgi pasakomų žodžių kiekiui arba per trumpi. Tai buvo atlikta apskaičiuojant apatinį (atkerta apatinius 25 % duomenų) ir viršutinį (atkerta viršutinius 25 % duomenų) kvartilius, taip paliekant tik tarpkvartilinį plotą. Taip išfiltravus išskirtis treniravimo duomenų rinkinys sutrumpėjo iki ~17 valandų.




**15 pav.** Filtruoto mokymo duomenų rinkinio garso įrašų kiekio pasiskirstymo grafikas pagal trukmę. Kur garso įrašo ilgis yra iki 5 sekundžių.

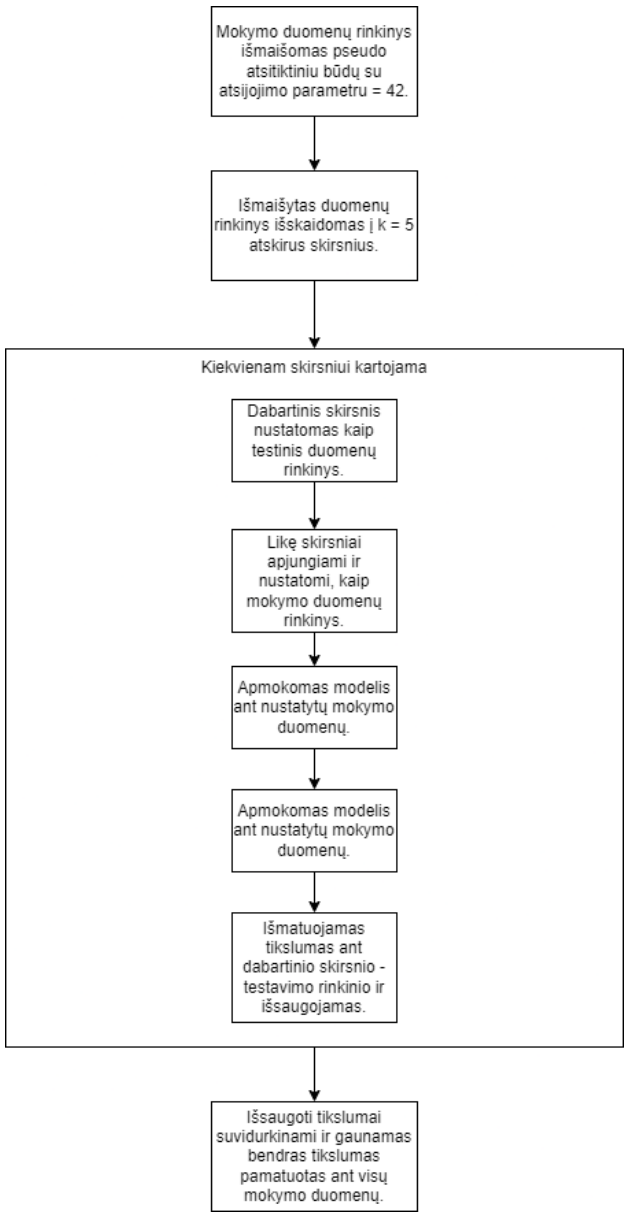
Papildomai, kad būtų mažiau apkraunama sistema ir greičiau būtų vykdomas mokymo procesas buvo nuspręsta apriboti garso įrašo perduodamo į mokymo procesą ilgį. Tam buvo apskaičiuotas garso įrašų ilgio treniravimo duomenyse viršutinis kvartilis, kuris buvo lygus ~5 sekundėm. Todėl garso įrašus, kurie buvo ilgesni, nei 5 sekundės buvo nuspręsta pašalinti iš mokymo duomenų rinkinio. Pašalinus šiuos duomenis, duomenų rinkinio trukmė tapo ~7 valandos (žr. **15 pav.**).

Iš garso įrašų anotacijų (transkripcijos) pašalinami specialūs simboliai, kurie nėra raidės arba neįeina į lietuvių kalbos raidyną. Galiausiai pritaikant duomenis jau apmokytiems „Wav2Vec2 base“ ir „Wav2Vec2 XLS-R 300m“ modeliams duomenų diskretizavimo dažnis turi būti padidintas iš 8kHz į 16kHz. Bendras procesas (žr. **16 pav.**).




**16 pav.** Duomenų pasiruošimo modelio mokymui procesas

Tikslumo sekimui ir rezultatų validumui užtikrinti mokymas buvo vykdytas pasitelkiant k-skirsnių kryžminę validaciją (angl. *k-Fold Cross-Validation*) metodą. Šis metodas pseudoatsitiktiniu būdų išskirsto treniravimo duomenų imtį į k skirsnių ir su kiekviena iteracija priskiria vis kitą skirsnį kaip testavimo imtį, o su likusiomis imtimis atlieka mokymą (žiūrėti **17 pav.**).




**17 pav.** k – skirsnių kryžminė validacija

**7 lentelė** Iš "Babel" duomenų rinkinio sukurtas raidynas

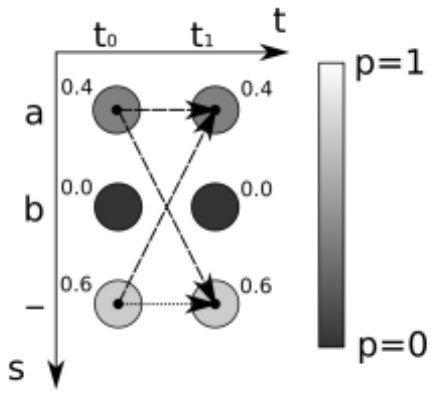
| **Raidė** | **Raidės unikalus numeris** | **Raidė** | **Raidės unikalus numeris** |
|---|---|---|---|
| o | 0 | f | 18 |
| š | 1 | h | 19 |
| ž | 2 | m | 20 |
| c | 3 | j | 21 |
| a | 4 | d | 22 |
| e | 5 | l | 23 |
| n | 6 | s | 24 |
| i | 7 | g | 25 |
| u | 8 | b | 26 |

| | | | |
|---|---|---|---|
| ą | 9 | t | 27 |
| ė | 10 | x | 28 |
| y | 11 | į | 29 |
| p | 12 | v | 30 |
| č | 13 | ę | 31 |
| ū | 14 | z | 32 |
| k | 15 | \| | 33 |
| ų | 16 | [UNK] | 34 |
| r | 17 | [PAD] | 35 |

Sukūrus raidyną **7 lentelė** iš „Babel“ duomenų rinkinio galima matyti, jog raidyną sudaro 32 raidės (kaip ir lietuvišką abėcėle), taip pat papildomi simboliai: „|“, kuris perteikia kalboje esančią tylą arba esantį tarpą išvesties tekste, taip pat „[UNK]“, kuris atstoja neatpažintą simbolį - raidę, galiausiai „[PAD]“, kuris yra reikalingas sulygiuoti kalbos įrašą su išvesties tekstu pagal jungiamosios laiko klasifikacijos modelį, kadangi garso įraše yra daugiau duomenų eilutės įrašų atspindinčių vienos raidės tarimo garsą, nei anotacijų faile [57].

### 3.3.2. „Wav2Vec2 Base“ ir „Wav2Vec2 XLS-R 300m“ modelių paruošimas

Sekti modelio tikslumą naudojama jungiamosios laiko klasifikacijos prarasties funkcija. Jungiamosios laiko klasifikacijos išvestis yra matrica sudarytą iš raidyno raidžių (**7 lentelė**) tikimybės per laiką. Pavyzdinė matrica sudaryta iš dviejų raidžių (a, b) ir įterpties simbolio „-“ (žr. **18 pav.**), galima matyti, jog didžiausia tikimybė yra sekos per laiką „-,-“.su pasitikėjimo rodikliu 0,36 (0,6*0,6) [58]. Tuomet išvedama tikslumo metrika lyginant spėjamą išvestį su anotacija ir taip apskaičiuojama prarasties funkcijos reikšmė. Apmokymo metu, duomenų rinkiniui D, skaičiuojama negatyvus logaritminis panašumas (tarp įvesties X ir išvesties Y) (žr. **19 pav.**) ir siekiama jį sumažinti [59]. Minėta prarasties funkcija yra skaičiuojama tiek testavimo, tiek validavimo duomenų rinkiniams apmokymo žingsnio metu. Validavimo duomenų rinkiniui galima nustatyti kas kiek žingsnių bus apskaičiuojama netekties funkcija. Eksperimento metu validavimo žingsnis atliekamas, kas 2000 žigsnių, modelio išsaugojimo, kas 2000 tūkstančius. Validavimo duomenų rinkinys mokymo procese nedalyvauja, tačiau pagal skirtumus tarp validavimo rinkinio vertinimo metrikų ir treniravimo duomenų rinkinio metrikų galima daryti įžvalgas dėl modelio tinkamumo ir prisitaikymo mokymo duomenims (angl. overfitting).



**18 pav.** Jungiamosios laiko klasifikacijos modelio grąžinama reikšmių tikėtinumo matrica [58]

$$\sum_{(X,Y)\in\mathcal{D}} -\log\ p(Y \mid X)$$

**19 pav.** Negatyvaus logaritminio panašumo lygtis [59]

Taip pat mokymo proceso tikslumui sekti naudojama žodžių klaidos dažni, kuri skaičiuojama validavimo duomenų rinkiniui per validavimo žingsnį. Vertinant modelio tikslumą ši metrika skaičiuojama testavimo duomenų imčiai apmokius modelį. Testavimo rinkiniui be žodžio klaidos dažnio skaičiuojama ir raidžių klaidos dažnio metrika. Pagal minėtas klaidos dažnio metrikas bus nustatomas modelio tikslumas ir optimalumas.

Mokymo parametrų vertės pažymėtos modelio parametrų reikšmių lentelėje (žr. **8 lentelė**). „Train_epochs“ parametras nurodo, kiek kartų mokymo algoritmas praeina visą mokymo duomenų rinkinį, keisdamas svorių parametrus. „Save_steps“ – nurodo, kas kiek algoritmo žingsnių bus išsaugomi svoriai. „Eval_steps“ – nurodo, kas kiek algoritmo žingsnių bus daromas testavimas su testavimo duomenų skirsniu. „Learning_rate“ – mokymosi greitis, parametras nurodo žingsnio dydi kuriuo bus atnaujintas gradientas. Nustatant didesnį mokymosi greitį algoritmas labiau pakeičia svorius kiekvieną iteraciją, rezultate optimali reikšmė turėtų būti pasiekta greičiau, tačiau esant per didelei reikšmei tiksli optimali reikšmė gali būti nepasiekta. Taip pat nustatant per mažą mokymosi greičio reikšmę mokymosi procesas užtrunka ilgiau, tačiau didesnė tikimybė, kad prarasties funkcija ras tikslią optimalią reikšmę. „Attention_dropout“ – parametras dėmesiu grįstose (angl. attention-based) architektūrose nurodo tikimybę, su kuria bus pašalinami neuronai iš neuroninio tinklo architektūros. Pašalinant neuronus, galima padidinti tikimybę, kad neuroninis modelis pernelyg prisitaikys (angl. overfit) prie mokymo duomenų, rezultate sumažės tikslumas ant testavimo ar realių duomenų. „Mask_time_prob“ – viso vektoriaus procentas (reikšmė nuo 0 iki 1), kuri bus užmaskuojama, taip siekiant pagerinti apmokymą uždengiant dalį garso įrašo. „Ctc_loss_reduction“ – su parametru „mean“ nurodo prarasties funkcijos išvesties reikšmes vidurkinti per mokymo grupės ilgį. Parametrų optimizavimui šiame tyrime nebuvo skiriamas didelis dėmesys, todėl tolimesniuose tyrimuose šiai daliai turėtų būti skirtas didesnis dėmesys.

**8 lentelė** Modelio mokymo parametrų reikšmės

| **Mokymo parametro (angl. hyperparameter) pavadinimas** | **Mokymo parametro vertė** |
|---|---|
| Train_epochs | 20 |
| Save_steps | 2000 |
| Eval_steps | 2000 |
| Learning_rate | $5x10^{-5}$ |
| Attention_dropout | 0 |
| Mask_time_prob | 0,05 |
| Ctc_loss_reduction | “mean“ |

### 3.3.3. Rašybos korekcijos algoritmo paruošimas

Galima rasti straipsnių, kuriuose yra tiriamas gramatinių klaidų pataisymas pasinaudojant kontekstiniais raktiniais žodžiais [59][61]. Tokiuose darbuose algoritmai ieško žodžio atitikmens

pagal kriterijus (pavyzdžiui fonemų arba raidžių atstumas) žodyne ir jeigu randamas yra pakeičiamas rastu atitikmeniu. Tokių būdų galima pataisyti kai kurias gramatines klaidas, jeigu vartojami žodžiai yra specifiniai ir retai vartojami kasdienėje kalboje. Taip siekiant pritaikyti tekstą prie specifinio konteksto [59].

Siekiant pagerinti kalbos vertimo į tekstą modelių tikslumą, buvo pasitelktas gramatinių klaidų korekcijos algoritmas. Šio algoritmo tikslas yra sutvarkyti kalbos vertimo į tekstą modelio paliktas klaidas. Kadangi modelis nesieja atpažįstamo garso su žodžiu, o su raidėmis. Todėl gautas žodžio spėjimas gali būti su paliktomis klaidomis, kurios neatitinka lietuvių kalbos rašybos taisyklių. Dalis žodžio gali būti nukirsta, sumaišytos panašaus skambėjimo raidės, pridėtos papildomos raidė. Šiame darbe buvo pasitelktas „Symmetric Delete Spelling Correction Algorithm (SymSpell)“ algoritmas. Tyrimams buvo naudojama „Python“ programavimo kalbos realizacija [60]. Algoritmas yra paremtas panašiausio žodžio iš žodyno paieška. Panašumo atstumas tarp žodžių yra nustatomas pagal „Damerau – Levenshtein“ atstumą ir pagal žodžių dažnio žodyne kriterijų. „SymSpell“ algoritmo nauda yra ta, kad jis pagrįstas simetrinio pašalinimo taisymo algoritmu, tai sumažina pakeitimo kandidato generavimo sudėtingumą, ieškant atstumui tarp žodžių porų. Šio algoritmo skaičiavimų sudėtingumas turėtų būti pastovus (O(1) trukmė), kadangi paieškos indeksavimas yra grįstas maišos lentele, kurios paieškos laiko sudėtingumas yra O(1)[62]. Reiškia, kad paieška visada užtruks tą patį laiką. Dėl išvardintų priežasčių „SymSpell“ algoritmas buvo pasirinktas siekiant sumažinti garso įrašų transkripcijos klaidas ir šiame darbe.

Atliekant tyrimą buvo panaudoti du žodynai. Pirmas iš įvairių šaltinių surinktas šiuolaikinių lietuviškų žodžių žodynas „Wordlist of the Contemporary Corpus of Lithuanian language“[63]. Šis žodynas buvo pasitelktas siekiant išsiaiškinti, kokią įtaką žodžių klaidos dažniui gali daryti gramatinių klaidų taisymo žodynas, kuris nėra susijęs su mokymo ar testavimo duomenimis. Žodynas turi 1850478 žodžių ir dažnių porų. Antras kontekstinis žodžių dažnių žodynas, tyrimo metu sudarytas iš „Babel“ treniravimo imties transkripcijų (žr. **20 pav.**). Šis žodynas pasitelktas, siekiant išsiaiškinti, kokią įtaką žodžių klaidos dažniui gali daryti gramatinių klaidų žodynas paremtas mokymo duomenų transkripcijomis. Gautas „Babel“ žodynas susidėjo iš 22413 žodžių ir dažnių porų.

**Algoritmas 1** Žodyno sudarymo algoritmas

Kiekvienam garsyno failo transkripcijos $T$ žodžiui $W$ sugeneruojamas dažnis i faila *zodynas.txt*.

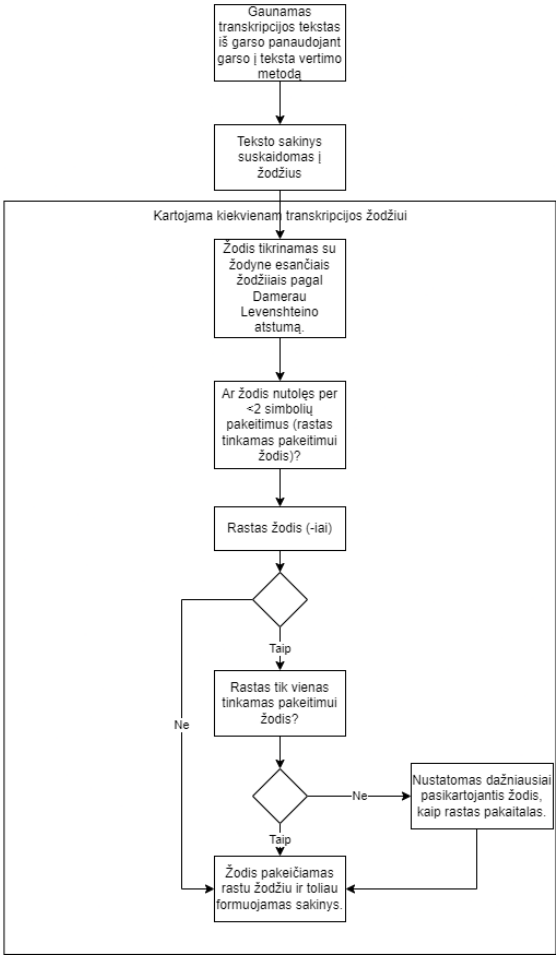
**Ivestis:** Transkripciju failas $TF$

**Inicijuojamas:** Žodžiu dažniu žodynas $Fq = \{\}$

```
1: Nuskaitomos transkripciju failo FT eilutės
2: for each FT.readLines() do
3:     Transkripcija T išskaidoma i W pagal T esančius tarpus. Funkcija split()
4:     for each W ∈ T.split() do
5:         Kiekvienam žodžiui W išsaugomas dažnis i dažniu žodyna Fq
6:         if not Fq.get(W) then
7:             Fq = {W : 1}
8:         else
9:             Jei žodis egzistuoja žodyne, prie dažnio fq pridedamas 1
10:            Fq = {W : fq + 1}
11:        end if
12:    end for
13: end for
14: Žodynas Fq irašomas i faila zodynas.txt
```

**20 pav.** Žodyno sudarymo algoritmo pseudokodas

Sukūrus žodžių dažnių žodyną, „SymSpell“ algoritmas buvo apjungtas su garso įrašų vertimo į tekstą logika. Vertimo į tekstą procesas parodytas (žr. **21 pav.**). Pirmiausia gaunamas transkripcijos tekstas iš pasirinkto metodo. Tuomet teksto sakinys suskaidomas į žodžius. Kiekvienam žodžiui tekste yra ieškomas panašiausias žodis (-iai) iš dažnių žodyno (šiame darbe yra naudojami du žodynai - šiuolaikinių žodžių žodynas, bei iš „Babel“ mokymo duomenų sudarytas žodynas), kurio „Damerau – Levenshtein“ atstumas yra ne didesnis nei 2 simboliai. Jeigu nerandamas toks žodis ar žodžiai, sakinyje paliekamas nepakeistas žodis. Jeigu rastas tik vienas žodis, tuomet sakinyje žodis pakeičiamas į rastą žodį. Jeigu randami keli žodžiai turintys vienodą atstumą, nuo sakinyje esančio žodžio, pirmumas teikiamas žodžiui, kuris turi didesnį dažnio skaičių žodyne. Šie veiksmai kartojami kiekvienam transkripcijos sakinio žodžiui. Atlikus šiuos veiksmus turėtų būti pataisomos gramatinės klaidos pakeičiant žodžius į atitikmenį iš žodyno.




**21 pav.** Garso transkripcijos procesas pridedant „SymSpell“ algoritmą

## 3.4. Tyrimo rezultatai

Tyrimo metu buvo apmokyti du kalbos į tekstą vertimo modeliai, su lietuviškos telefoninės šnekos anotuotu duomenų rinkiniu, panaudojant „Hugging face“ transformacinių modelių karkasą ir pasitelkiant iš anksto apmokytus modelius su užšaldytais svoriais (angl. *frozen weights*).

Taip pat siekiant pagerinti minėtų modelių tikslumą buvo panaudotas „Symmetric Delete Spelling Correction Algorithm (SymSpell)“ algoritmas, kuris pasitelkdamas „Damerau – Levenshtein“ atstumą siekia surasti klaidingai parašyto žodžio artimiausią atitikmenį iš žodyno, taip mažindamas žodžio klaidos dažnio metriką ir gerindamas transkripcijos tikslumą.

### 3.4.1. Metodų tikslumo vertinimai pagal klaidų metrikas su k – skirsniais

Metodų rezultatai buvo vertinami k – skirsnių metodu, bei ant testavimo duomenų rinkinio. Žodžių klaidos dažnio metrikos lentelėje (žr. **9 lentelė**), yra nurodomas kiekvieno iš garso įrašų vertimo į tekstą modelių rezultatas kiekvienam k – skirsniui. „Google STT“ stulpelyje yra „Google Speech To Text API“ aplikacijos vertimo į tekstą rezultatai. „Wav2Vec2 – Base“ stulpelyje yra „Wav2Vec – Base“ modelio apmokyto su „Babel“ duomenimis rezultatai. „Wav2Vec2 XLS-R“ stulpelyje yra „Wav2Vec2 XLS-R“ modelio apmokyto su „Babel“ duomenimis rezultatai. Papildomai, prie kiekvieno modelio yra dar du stulpeliai. „+ Šiuolaikinis žodynas“ prie modelio

pridedant „SymSpell“ algoritmo žodžių paiešką šiuolaikinių žodžių dažnių žodyne, siekiant ištaisyti gramatines klaidas. Taip pat stulpelis „+ Babel žodynas“ prie modelio pridedant kontekstinę žodžių paiešką, atliekant artimiausio žodžių paiešką „Babel“ treniravimo duomenų žodžių dažnių žodyne. Lentelėje, taip pat yra eilutė su apskaičiuotais kiekvieno metodo vidurkiais. Tuomet kiekvienas metodas buvo reitinguojamas pagal klaidos dažnio reikšmę kiekviename skirsnyje. Vidurkio reikšmė buvo paskaičiuota suvidurkinant kiekviename skirsnyje užimamą vietą pagal tikslumą, kuo žodžių klaidos dažnis mažesnis, tuo aukštesnė vieta algoritmui yra priskiriama.

**9 lentelė** Metodų žodžių klaidos dažnio metrikos kiekvienam iš k - skirsnių

| **Modelis** / **Skirsnio nr** | **Google STT, %** | **Google STT + Šiuolaikinis žodynas, %** | **Google + Babel žodynas, %** | **Wav2Vec2 base, %** | **Wav2Vec2 base + Šiuolaikinis žodynas, %** | **Wav2Vec2 base + Babel žodynas, %** | **Wav2Vec2 XLS-R, %** | **Wav2Vec2 XLS-R + Šiuolaikinis žodynas, %** | **Wav2Vec2 XLS-R + Babel žodynas, %** |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 83.64 | 85.87 | 83.77 | 75.90 | 75.50 | 72.40 | 55.00 | 55.70 | 52.30 |
| 2 | 82.45 | 84.81 | 82.44 | 74.30 | 74.00 | 71.20 | 53.50 | 54.00 | 51.60 |
| 3 | 83.62 | 85.95 | 83.67 | 74.90 | 75.10 | 72.40 | 52.00 | 52.30 | 50.50 |
| 4 | 84.12 | 86.33 | 84.19 | 75.30 | 74.70 | 71.50 | 54.30 | 54.90 | 52.30 |
| 5 | 83.76 | 85.84 | 83.72 | 74.40 | 74.00 | 71.40 | 50.00 | 49.80 | 47.70 |
| 6 | 83.82 | 86.16 | 83.84 | 73.90 | 74.00 | 72.00 | 54.50 | 54.70 | 51.10 |
| | | | | | | | | | |
| Vidurkis | 83.57 | 85.83 | 83.60 | 74.78 | 74.55 | 71.82 | 53.22 | 53.57 | 50.92 |
| Vietos pagal kiekvieną skirsnį vidurkis. | 7.17 | 9.00 | 7.83 | 5.50 | 5.50 | 4.00 | 2.17 | 2.83 | 1.00 |

„Wilcoxon“ porų testas buvo atliktas nustatyti ar skirtumas tarp modelių yra statistiškai reikšmingas.

**10 lentelė** Garso įrašo vertimo į tekstą statistinio reikšmingumo palyginimo tarp metodų. „Wilcoxon“ porų testo p – reikšmės

| | **Google STT** | **Wav2Vec2 base** | **Wav2Vec2 XLS-R** |
|---|---|---|---|
| **Google STT** | | 0,031 | 0,031 |
| **Wav2Vec2 base** | 0,031 | | 0,031 |
| **Wav2Vec2 XLS-R** | 0,031 | 0,031 | |

**11 lentelė** Garso įrašo vertimo į tekstą statistinio reikšmingumo palyginimo tarp metodų su pridėta šiuolaikinio žodyno paieška. „Wilcoxon“ porų testo p – reikšmės

| | **Google STT+ Šiuolaikinis žodynas** | **Wav2Vec2 base+ Šiuolaikinis žodynas** | **Wav2Vec2 XLS-R+ Šiuolaikinis žodynas** |
|---|---|---|---|
| **Google STT+ Šiuolaikinis žodynas** | | 0,031 | 0,031 |
| **Wav2Vec2 base+ Šiuolaikinis žodynas** | 0,031 | | 0,031 |

| | | | |
|---|---|---|---|
| **Wav2Vec2 XLS-R+ Šiuolaikinis žodynas** | 0,031 | 0,031 | |

**12 lentelė** Garso įrašo vertimo į tekstą statistinio reikšmingumo palyginimo tarp metodų su pridėta „Babel“ žodyno paieška. „Wilcoxon“ porų testo p – reikšmės

| | **Google STT+ Babel žodynas** | **Wav2Vec2 base+ Babel žodynas** | **Wav2Vec2 XLS-R+ Babel žodynas** |
|---|---|---|---|
| **Google STT+ Babel žodynas** | | 0,031 | 0,031 |
| **Wav2Vec2 base+ Babel žodynas** | 0,031 | | 0,031 |
| **Wav2Vec2 XLS-R+ Babel žodynas** | 0,031 | 0,031 | |

Tarp skirtingų metodų buvo atmesta (p – reikšmė < α = 0,05) nulinė hipotezė, kad modelių rezultatų panašumas yra statistiškai reikšmingas. Lyginimas atliktas tarp originalių metodų (žr. **10 lentelė**), tarp originalių metodų su pridėta šiuolaikinio žodyno paieška (žr. **11 lentelė**), bei originalių metodų su pridėta „Babel“ žodyno paieška (žr. **12 lentelė**).

**13 lentelė** Garso įrašo vertimo į tekstą statistinio reikšmingumo palyginimo tarp vienodų metodų (su ir be pridėtų žodynų). „Wilcoxon“ porų testo p – reikšmės. „Google STT“ metodas.

| | **Google STT** | **Google STT + Šiuolaikinis žodynas** | **Google + Babel žodynas** |
|---|---|---|---|
| **Google STT** | | 0,031 | 0,219 |
| **Google STT + Šiuolaikinis žodynas** | 0,031 | | 0,031 |
| **Google + Babel žodynas** | 0,219 | 0,031 | |

„Google STT“ (žr. **13 lentelė**) metodo nulinė hipotezė, kad modeliai yra statistiškai reikšmingai panašūs, buvo atmesta (p – reikšmė = 0,031 < α = 0,05) tarp „Google STT“ ir „Google STT + Šiuolaikinis žodynas“, bei „Google STT + Šiuolaikinis žodynas“ ir „Google + Babel žodynas“ porų. Tačiau, statistiškai reikšmingo panašumo hipotezė nebuvo atmesta (p – reikšmė = 0,219 > α = 0,05) tarp „Google STT“ ir „Google + Babel žodynas“ poros. Todėl galima teigti jog šie modeliai statistiškai nėra skirtingi.

**14 lentelė** Garso įrašo vertimo į tekstą statistinio reikšmingumo palyginimo tarp vienodų metodų (su ir be pridėtų žodynų). „Wilcoxon“ porų testo p – reikšmės. „Wav2Vec2 base“ metodas.

| | **Wav2Vec2 base** | **Wav2Vec2 base + Šiuolaikinis žodynas** | **Wav2Vec2 base + Babel žodynas** |
|---|---|---|---|
| **Wav2Vec2 base** | | 0,156 | 0,031 |
| **Wav2Vec2 base + Šiuolaikinis žodynas** | 0,156 | | 0,031 |
| **Wav2Vec2 base + Babel žodynas** | 0,031 | 0,031 | |

Priešingai nei „Google STT“ metodo poroms, „Wav2Vec2 base“ (žr. **14 lentelė**) metodo nulinė hipotezė, kad modeliai yra statistiškai reikšmingai panašūs, buvo atmesta (p – reikšmė = 0,031 < α = 0,05) tarp „Wav2Vec2 base“ ir „Wav2Vec2 base + Babel žodynas“, bei „Wav2Vec2 base +

Šiuolaikinis žodynas“ ir „Wav2Vec2 base + Babel žodynas“ porų. Statistiškai reikšmingo panašumo hipotezė nebuvo atmesta (p – reikšmė = 0,156 > α = 0,05) tarp „Wav2Vec2 base“ ir „Wav2Vec2 base + Šiuolaikinis žodynas“ poros. Todėl galima teigti jog šie modeliai statistiškai nėra skirtingi.

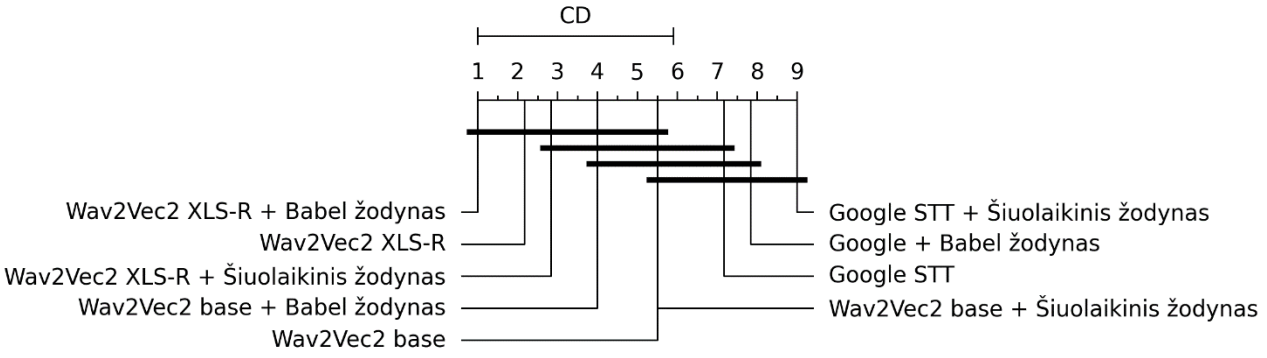
**15 lentelė** Garso įrašo vertimo į tekstą statistinio reikšmingumo palyginimo tarp vienodų metodų (su ir be pridėtų žodynų). „Wilcoxon“ porų testo p – reikšmės. „Wav2Vec2 XLS-R“ metodas.

| | **Wav2Vec2 XLS-R** | **Wav2Vec2 XLS-R + Šiuolaikinis žodynas** | **Wav2Vec2 XLS-R + Babel žodynas** |
|---|---|---|---|
| **Wav2Vec2 XLS-R** | | 0,094 | 0,031 |
| **Wav2Vec2 XLS-R + Šiuolaikinis žodynas** | 0,094 | | 0,031 |
| **Wav2Vec2 XLS-R + Babel žodynas** | 0,031 | 0,031 | |

Panašiai, kaip „Wav2Vec2 base“ metodo poroms, „Wav2Vec2 XLS-R“ (žr. **15 lentelė**) metodo nulinė hipotezė, kad modeliai yra statistiškai reikšmingai panašūs, buvo atmesta (p – reikšmė = 0,031 < α = 0,05) tarp „Wav2Vec2 XLS-R“ ir „Wav2Vec2 base + Babel žodynas“, bei „Wav2Vec2 base + Šiuolaikinis žodynas“ ir „Wav2Vec2 base + Babel žodynas“ porų. Statistiškai reikšmingo panašumo hipotezė nebuvo atmesta (p – reikšmė = 0,094 > α = 0,05) tarp „Wav2Vec2 XLS-R“ ir „Wav2Vec2 XLS-R + Šiuolaikinis žodynas“ poros. Todėl galima teigti jog šie modeliai statistiškai nėra skirtingi.

Iš „Wilcoxon“ porų testų rezultatų galima kelti hipotezę, jog tarpusavy kiekvienas teksto vertimo į kalbą metodas nėra panašus. Panašumas atsiranda tarp to paties metodo variacijų, originalaus ir su pridėtu vienu iš žodynų. „Google STT“ metodo atveju, originalus metodas statistiškai reikšmingo skirtumo nerodo tarp originalaus metodo ir originalaus metodo su pridėtu „Babel“ dažnių žodynu. „Wav2Vec2 base“ bei „Wav2Vec2 XLS-R“ metodų atveju, statistiškai reikšmingo skirtumo nerodo originalaus metodo pora su metodu papildytu šiuolaikinio žodyno paieška.

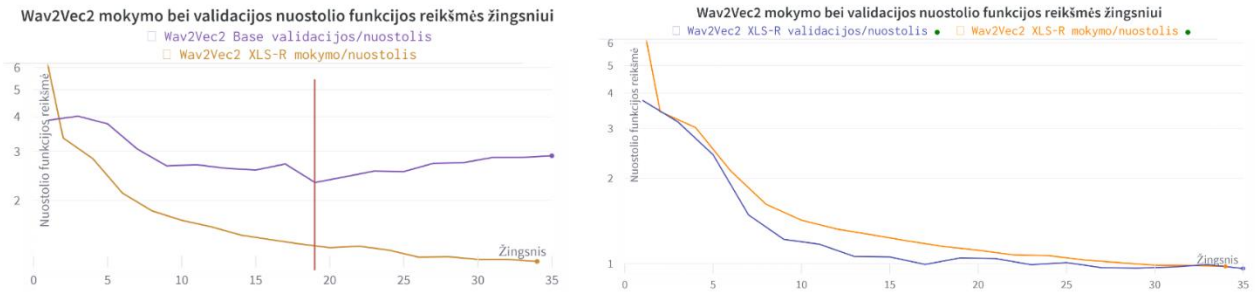
Taip pat buvo atlikta statistinė analizė siekiant palyginti visus metodus per visų šešių skirsnių rezultatus. Metodai atvaizduojami „Demšar“ lentelėje [64] (žr. **22 pav.**). Statistiškai neturintys reikšmingų skirtumų algoritmai apjungiami. Taip pat užbrėžiamas virš grafiko kritinis atstumas apskaičiuotas pasitelkiant Nemanyi testą. Kritinis atstumas apima „Wav2Vec2“ metodus, dėl to galima sakyti, jog skirtumas tarp šių metodų nėra statistiškai reikšmingas. Tačiau „Google STT“ kartu su jam pritaikytais klaidų taisymo metodais neįeina į kritinį atstumą, todėl galima sakyti, jog šio metodo skirtumas nuo kitų yra statistiškai reikšmingas.




**22 pav.** Metodų palyginimas naudojant Nemanyi testą. CD – kritinis atstumas.

### 3.4.2. Metodų tikslumo vertinimas su išskirtu testavimo rinkiniu

Testuojami „Wav2Vec2 Base“ ir „Wav2Vec2 XLS-R“ modeliai buvo parinkti pagal treniravimo ir patikros duomenų imtims apskaičiuotą prarasties funkcijos reikšmes (žr. **Error! Reference source not found. pav.**). „Wav2Vec2 Base“ modelio testavimui buvo imamas modelis išsaugotas treniravimo žingsnyje, kur žingsnio minėtos metrikos yra žemiausios ir tarpusavyje panašiausios, tai buvo ties maždaug 19 žingsniu grafike (žr. **23 pav.** „A)“**Error! Reference source not found.**). „Wav2Vec2 XLS-R“ modelis buvo imamas iš paskutinio žingsnio, kadangi tarp mokymo ir patikrinimo netekties funkcijos reikšmių nebuvo reikšmingos atskirties (žr. **23 pav.** „B)“).




A) B)

**23 pav** „Wav2Vec2“ modelių mokymo proceso prarasties funkcijos grafikai. „A)“ grafikas yra „Wav2Vec2 Base“, „B)“ grafikas yra „Wav2Vec2 XLS-R“ modelio.

Papildomai buvo atlikti tyrimai su testavimo duomenų imtimi. Kuri buvo atskirta iš „Babel“ duomenų rinkinio. Testavimo imtis buvo išskirta, siekiant atlikti tyrimus su visais duomenimis esančiais mokymo imtyje. Kadangi k – skirsnių metodas dalį duomenų paskiria testavimui. Taip pat testavimo imtis yra sudaryta iš nefiltruotų duomenų. Garso įrašų trukmė gali būti ilgesnė nei 5 sekundės (priešingai nei treniravimo imty). Be to testavimo imtis atspindi realesnes sąlygas, kadangi išskirtiniai, pagal įrašo trukmę ir pasakomų žodžių santykį, garso įrašai nėra pašalinami.

**16 lentelė** Metodų spėjimo rezultatai testavimo duomenų imčiai

| **Modelio pavadinimas** | **WER, %** | **CER, %** | **WER+ Šiuolaikinis žodynas, %** | **CER+ Šiuolaikinis žodynas, %** | **WER+ „Babel“ žodynas, %** | **CER+ „Babel“ žodynas, %** |
|---|---|---|---|---|---|---|
| **Google Speech to Text API** | 80,8 | 37,1 | 83,1 | 38,5 | 81,6 | 39,2 |
| **Wav2Vec2 base** | 79,1 | 36,4 | 77,7 | 38,4 | 72,7 | 37,9 |
| **Wav2Vec2 XLS-R 300m** | 50,9 | 22,6 | 52,0 | 24,2 | 47,7 | 23,7 |

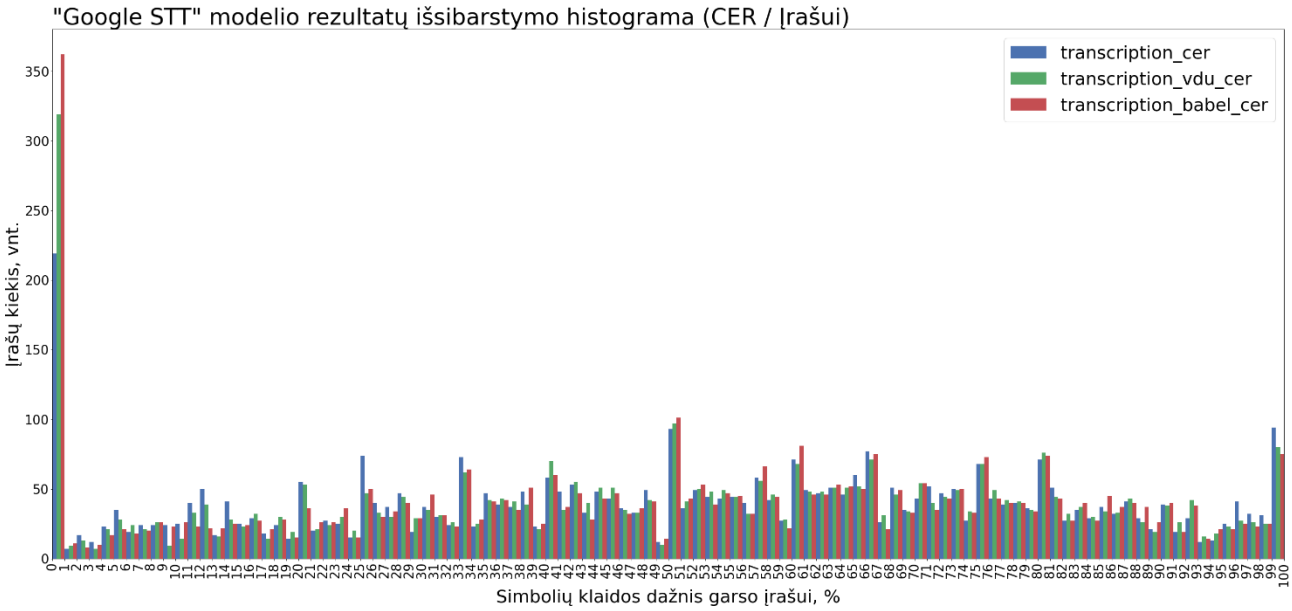
Kadangi „Wav2Vec2“ modeliams mokymo duomenys buvo filtruojami pagal laiką, papildomai klaidų metrikos buvo apskaičiuotos ir atskirai garso įrašams iki 5 sekundžių (viso 3989 garso įrašai) ir garso įrašams viršijantiems 5 sekundes (viso 1969 garso įrašai) (žr. **17 lentelė**). Galima matyti jog modelis pagal žodžių klaidų dažnio metriką veikia geriau su garso įrašais, kurie yra iki 5 sekundžių ilgio. Pagal simbolių klaidos dažnį „Wav2Vec2 base“ modelis taip pat prognozuoja tiksliau garso įrašus iki 5 sekundžių, tačiau „Wav2Vec2 XLS-R“ modelis pagal tą pačią metriką tiksliau prognozuoja garso įrašus virš 5 sekundžių trukmės.

**17 lentelė** „Wav2Vec2“ modelių spėjimo rezultatai testavimo duomenų imčiai išskiriant garso įrašus iki 5 ir virš 5 sekundžių

| | **WER, %** | | **CER, %** | | **WER+ Šiuolaikinis žodynas, %** | | **CER+ Šiuolaikinis žodynas, %** | | **WER+ „Babel“ žodynas, %** | | **CER+ „Babel“ žodynas, %** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **<=5s** | **>5s** | **<=5s** | **>5s** | **<=5s** | **>5s** | **<=5s** | **>5s** | **<=5s** | **>5s** | **<=5s** | **>5s** |
| **Wav2Vec2 base** | 72,70 | 82,81 | 36,17 | 36,61 | 71,64 | 81,21 | 37,99 | 38,66 | 67,22 | 75,86 | 37,70 | 38,05 |
| **Wav2Vec2 XLS-R 300m** | 49,30 | 51,95 | 23,97 | 21,95 | 50,29 | 53,10 | 25,54 | 23,51 | 46,23 | 48,66 | 24,83 | 23,06 |

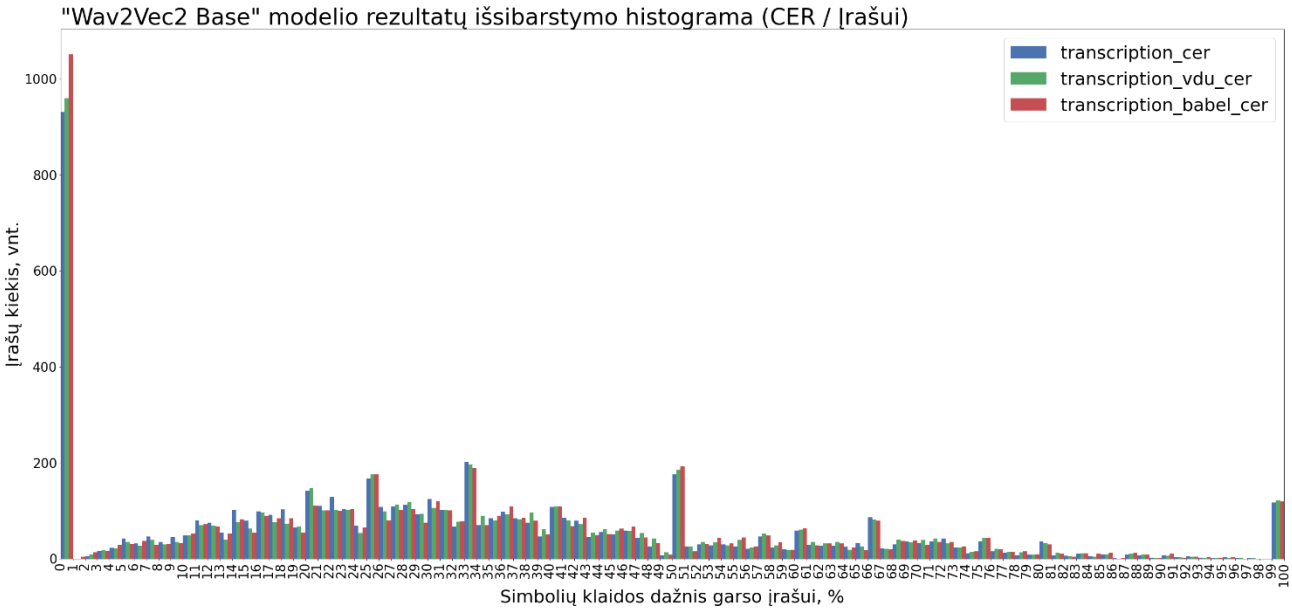
Modelių metodų tikslumo pasiskirstymą galima matyti metodų rezultatų išsibarstymo stulpelinėse diagramose (žr. **24 pav.**, **25 pav.**, **26 pav.**). Kadangi tikslumo išsibarstymas buvo matuojamas kiekvienam garso įrašui atskirai, (vidutiniškai ~7 žodžiai per garso įrašą), matuojant žodžių klaidos dažnį rezultatas būtų iškreipiamas. Todėl kaip klaidos metrika buvo pasirinktas simbolių klaidos dažnis.

„Google Speech to Text API“ metodo išsibarstymo histogramoje (žr. **24 pav.**), galima matyti, kad didžiausias įrašų kiekis yra turinčių 0 % simbolių klaidos dažnį. Toliau didesnis reikšmių kiekis yra turinčių simbolių klaidos dažnį virš 50 %.



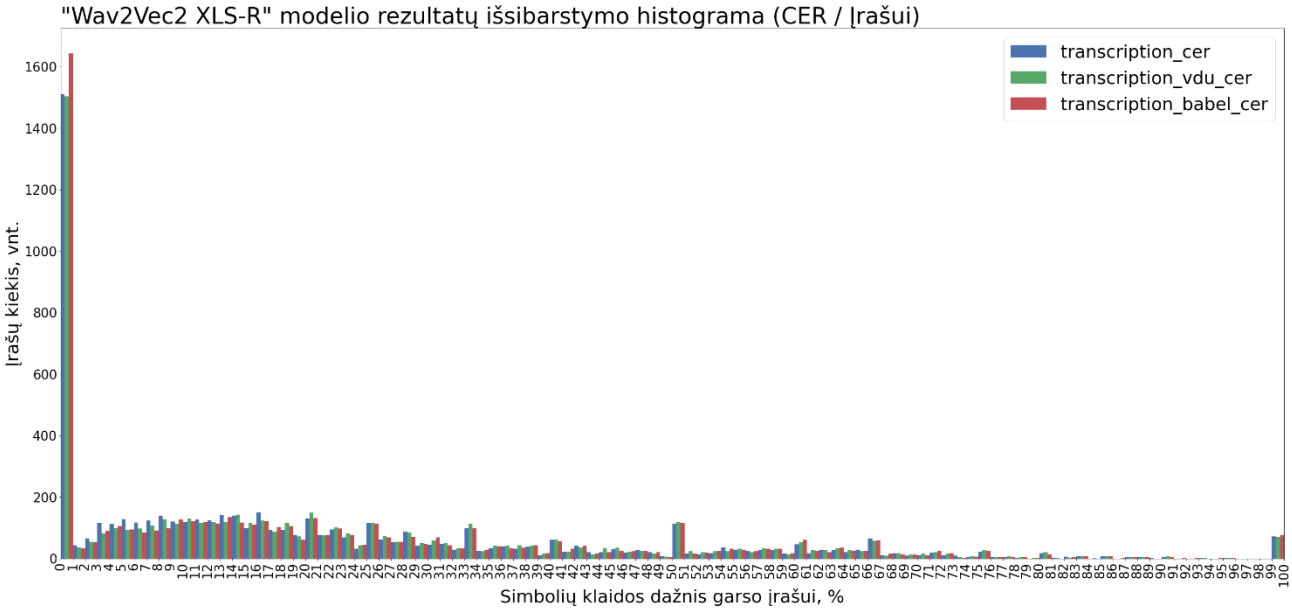

**24 pav.** „Google Speech to Text API“ metodų rezultatų išsibarstymo stulpelinė diagrama. Nurodanti įrašų kiekį turinčių atitinkamą modelių simbolių klaidos dažnio metriką procentais.

„Wav2Vec2 Base“ metodo išsibarstymo histogramoje (žr. **25 pav.**), galima matyti panašią tendenciją kaip ir „Google Speech to Text API“ sprendimuose. Kadangi didžiausias įrašų kiekis yra turinčių 0 % simbolių klaidos dažnį. Tačiau priešingai, nei minėtas sprendimas „Wav2Vec2 Base“ didesnis reikšmių kiekis yra turinčių mažesnį nei 50 % žodžių klaidos dažnį.




**25 pav.** „Wav2Vec2 Base“ metodų rezultatų išsibarstymo stulpelinė diagrama. Nurodanti įrašų kiekį turinčių atitinkamą modelių simbolių klaidos dažnio metriką procentais.

„Wav2Vec2 XLS-R“ metodo išsibarstymo histogramoje (žr. **26 pav.**), galima matyti panašią tendenciją kaip ir prieš tai aptartuose sprendimuose. Kadangi didžiausias įrašų kiekis yra turinčių 0 % simbolių klaidos dažnį, taip pat šis sprendimas turi didžiausią kiekį įrašų su apskaičiuotu 0 % simbolių klaidų dažniu iš visų trijų metodų. Panašiai, kaip ir minėtas sprendimas „Wav2Vec2 Base“, didesnis reikšmių kiekis yra turinčių mažesnį nei 50 % simbolių klaidos dažnį.




**26 pav.** „Wav2Vec2 XLS-R“ metodų rezultatų išsibarstymo stulpelinė diagrama. Nurodanti įrašų kiekį turinčių atitinkamą modelių simbolių klaidos dažnio metriką procentais.

Iš metodų rezultatų išsibarstymo stulpelines diagramas (žr. **24 pav.**, **25 pav.**, **26 pav.**). Galima matyti jog visi metodai turi išsiskiriantį kiekį ties 100 % simbolių klaidos dažniu. Taip galėjo nutikti dėl „nešvarių“ duomenų. Galima manyti jog testavimo (tikriausiai ir mokymo) duomenyse yra įsivėlusių anotavimo klaidų, kurios mažina modelio tikslumą. Anotavimo klaidos yra pastebimos ir

kituose darbuose ir duomenų rinkiniuose. Kurios gali atsirasti dėl žmogaus nuovargio, kadangi anotavimas yra monotoniška užduotis arba dėl nepakankamų kompetencijų atlikti anotavimo užduočiai . Taip pat visi modeliai turi gana aukštą kiekį garso įrašų atpažintų su 0 % klaidos metrika. Galima manyti, jog visi metodai geriau atpažįsta trumpus garso įrašus. Galima matyti, jog daugiausia garso įrašų, kurie patenka į 0 % simbolių klaidos dažnį, yra vieno žodžio ilgio (žr. **18 lentelė**). Todėl galima teigti, jog visi metodai labai gerai veikia su trumpais žodžiais.

**18 lentelė** Žodžių kiekio statistiniai įverčiai garso įrašams, kuriems metodų spėjimo simbolių klaidos dažnis 0 %.

| | **Google STT** | **Wav2Vec2 base** | **Wav2Vec2 XLS-R** |
|---|---|---|---|
| **Vidurkis** | 2,25 | 1,51 | 2,28 |
| **Mediana** | 1 | 1 | 1 |

Pagal „Google STT“ pavyzdines prognozes (žr.**19 lentelė**) galima matyti jog metodas kai kuriais atvėjais spėja visiškai atsitiktinius žodžius, nesusijusius su kalbos anotacija. Tai gali būti, kad pats „Google STT“ algoritmas turi žodžių pataisymo metodą savyje ir bando priskirti artimą žodį. Iš teksto, kuris gautas pridėjus šiuolaikinio žodyno paiešką, galima matyti, kad žodyne nėra žodžio „o“ ir jis yra pakeičiamas raide „į“, taip padarant klaidą. Taip pat galima matyti, jog „Babel“ žodyne nėra žodžio „pelyja“, nes šis žodis yra pakeičiamas į žodį „nelyja“. Peržvelgiant prognozes, matosi labai dideli transkripcijos neatitikimai kalbos anotacijai.

**19 lentelė** „Google STT“ pavyzdinės prognozės

| **Kalbos anotacija** | **Prognozuojamas tekstas** | **Su šiuolaikinio žodyno paieška** | **Su „Babel“ žodyno paieška** |
|---|---|---|---|
| nu tai normaliai visai | kavos | kavos | kavos |
| tai nedlyva toks kai lyja tai ko gero ir nedirba žinai | mediena pelyja tai nugarą nedarbas | mediena pelyja tai nugarą nedarbas | mediena nelyja tai nugarą nedarbas |
| kasparas skambino praeitą savaitę gal skambino | kasparas skambino savaitę gal skambino | kasparas skambino savaitę gal skambino | kasparas skambino savaitę gal skambino |
| o tai kas o tai kas tau valgys | o tai kas tau skambino | į tai kas tau skambino | o tai kas tau skambino |
| tai čia reik grindis dar susidėt | susidėti | susidėti | susidėti |

Pagal pavyzdines prognozes (žr. **20 lentelė**), galima matyti, jog modelis bando spėti panašius žodžius, tiesiog kai kuriais atvejais yra sumaišomos arba pridedamos papildomos raidės.

**20 lentelė** „Wav2Vec2 Base“ pavyzdinės prognozės

| **Kalbos anotacija** | **Prognozuojamas tekstas** | **Su šiuolaikinio žodyno paieška** | **Su „Babel“ žodyno paieška** |
|---|---|---|---|
| nu tai normaliai visai | u tai normaliaisai | u tai normaliais | u tai normaliais |
| tai nedlyva toks kai lyja tai ko gero ir nedirba žinai | tai nedivai tas kai lie tgt u gera nedirba žinai | tai nežinai tas kai lie tai u gera nedirba žinai | tai nedyvai tas kai lie tt u gera nedirba jinai |
| kasparas skambino praeitą savaitę gal skambino | kasvaras skambino praėitę savaitė gal skambino | kasparas skambino praeitą savaitė gal skambino | kasparas skambino praeitę savaitė gal skambino |
| o tai kas o tai kas tau valgys | o tai kas o tai kas tau valgysams | o tai kas o tai kas tau valgymas | o tai kas o tai kas tau valgymams |

| | | | |
|---|---|---|---|
| nu tai normaliai visai | u tai normaliaisai | u tai normaliais | u tai normaliais |

Iš „Wav2Vec2 XLS-R“ pavyzdinių prognozių (žr. **21 lentelė**), galima matyti jog spėjimai tikslesni už „Wav2Vec2 base“ pavyzdinės prognozės reikšmes (žr. **20 lentelė**), tačiau vis tiek kai kurie žodžiai ne visiškai atitinka garso įrašų anotacijas. Ši informacija taip pat atsispindi metodų spėjimo lentelėje (žr. **16 lentelė**).

**21 lentelė** „Wav2Vec2 XLS-R“ pavyzdinės prognozės

| **Kalbos anotacija** | **Prognozuojamas tekstas** | **Su šiuolaikinio žodyno paieška** | **Su „Babel“ žodyno paieška** |
|---|---|---|---|
| nu tai normaliai visai | tai normaliai visai | tai normaliai visai | tai normaliai visai |
| tai nedlyva toks kai lyja tai ko gero ir nedirba žinai | tai nedyla t kaily tk tuka gero nedirba žinai | tai nelyja t kaip tu tukas gero nedirba žinai | tai nedyla t kaily tk tuka gero nedirba jinai |
| kasparas skambino praeitą savaitę gal skambino | kasparas skambino praeitą savaitę gal skambino | kasparas skambino praeitą savaitę gal skambino | kasparas skambino praeitą savaitę gal skambino |
| o tai kas o tai kas tau valgys | o tai kas o tai kas tau valgysams | o tai kas o tai kas tau valgymas | o tai kas o tai kas tau valgymams |
| tai čia reik grindis dar susidėt | žei grindią susėdėt | nei grindis susidėt | bei grindis susidėt |

### 3.4.3. Metodų greitaveikos tyrimas.

Atliekant testavimą taip pat buvo išmatuota metodų greitaveika. Greitaveikos rezultatai pateikiami modelių greitaveikos lentelėje (žr. **22 lentelė**). Greitaveikai patikrinti buvo atlikti penki pakartojimai su „Wav2Vec2“ architektūros modeliais. Su „Google Speech to Text API“ sprendimu buvo atliktas tik vienas bandymas, dėl to, kad bandymas trunka virš valandos ir kainuoja pinigus. „Wav2Vec2 Base“ algoritmo vidutinis spėjimo greitis visai testavimo duomenų imčiai nešiojamame kompiuteryje truko vidutiniškai ~6 minutėmis ilgiau, nei „Wav2Vec2 XLS-R“ modelis. Tačiau serveryje testavimo atlikimo trukmė skyrėsi tik per ~40 sekundžių. Taip galėjo nutikti kadangi atliekant spėjimus testavimo imčiai, buvo vykdomas grupuotas spėjimas (angl. *batch inference*). Tą galimai lėmė didesnis grafinės kortos operatyviosios atminties kiekis.

**22 lentelė** Modelių greitaveikos matavimas testavimo duomenų rinkiniui, be žodyno paieškos logikos. Matuojant kiek laiko modelis užtruko atpažinti visus testavimo duomenų imties garso įrašus skaičiuojant minutėmis.

| **Bandymo nr.** | **Modelio pavadinimas** | **Nešiojamas kompiuteris** | **Serveris** | **“Google Speech to Text API”** |
|---|---|---|---|---|
| **1** | „Wav2Vec2 Base“ | 00:08:58 | 00:07:34 | |
| | „Wav2Vec2 XLS-R“ | 00:14:40 | 00:08:31 | |
| **2** | „Wav2Vec2 Base“ | 00:09:05 | 00:07:22 | |
| | „Wav2Vec2 XLS-R“ | 00:14:54 | 00:08:20 | |
| **3** | „Wav2Vec2 Base“ | 00:08:40 | 00:07:44 | |
| | „Wav2Vec2 XLS-R“ | 00:15:40 | 00:08:22 | |
| **4** | „Wav2Vec2 Base“ | 00:08:35 | 00:07:52 | |
| | „Wav2Vec2 XLS-R“ | 00:15:12 | 00:08:40 | |
| | „Wav2Vec2 Base“ | 00:08:56 | 00:07:51 | |

| | | | | |
|---|---|---|---|---|
| **5** | „Wav2Vec2 XLS-R“ | 00:16:24 | 00:08:48 | |
| | | | | |
| **Vidurkis** | „Wav2Vec2 Base“ | 00:08:50 | 00:07:40 | |
| | „Wav2Vec2 XLS-R“ | 00:15:22 | 00:08:32 | |
| | „Google Speech to Text API“ | | | 01:31:00 |

Metodo vidutinio greičio atliekant spėjimus su testavimo duomenų imtimi (žr. **23 lentelė**) galima matyti, kiek vidutiniškai užtrunkama spėjant vieną minutę garso įrašo. Sparčiausiai tokios trukmės garso įrašą apdorotų „Wav2Vec2 Base“ užduotį atlikdamas per 1,09 sekundės, antroje vietoje to paties modelio implementacija veikianti nešiojamame kompiuteryje. „Wav2Vec2 XLS-R“ architektūra keliomis šimtosios dalimis atsilieka nuo antros vietos, o nešiojamame kompiuteryje veikianti ši architektūra atsilieka jau ~1 sekunde, tai yra beveik dvigubai ilgesnis apdorojimo laikas. „Google Speech to Text API“ sprendimas užtruko ~13 sekundžių apdoroti vieną minutę garso įrašo, tai lėmė ir tai, jog garso įrašas yra perduodamas per taikomąją programų sąsają interneto tinklu. Kas įveda dar papildomą laiką failo perdavimui, ne tik apdorojimui.

**23 lentelė** Metodo vidutinis greitis atliekant spėjimus su testavimo duomenų imtimi. Skaičiuojant kiek vidutiniškai metodas trunka atpažinti vieną minutę garso įrašo, skaičiuojant sekundėmis.

| | **Nešiojamas kompiuteris** | **Serveris** | **“Google Speech to Text API”** |
|---|---|---|---|
| „Wav2Vec2 Base“ | 1,19 s | 1,09 s | |
| „Wav2Vec2 XLS-R“ | 2,20 s | 1,22 s | |
| „Google Speech to Text API“ | | | 13,00 s |

Žodyno paieškos trukmės lentelėje (žr. **24 lentelė**) galima matyti, kiek užtruko žodyno paieškos pataisymas visam testavimo duomenų rinkiniui. Iš laiko trukmės vidurkio testavimo duomenų imčiai galima matyti, jog ilgiausiai paieška buvo atliekama su „Wav2Vec2 Base“ modelio spėjimais, trumpiausiai su „Google Speech to Text API“ sprendimu. Taip galėjo nutikti todėl, kad komercinis sprendimas savyje jau turi žodžių pataisymo funkcionalumą ir žodžiai nėra iškraipomi, o „Wav2Vec2 Base“ sprendimo spėjimai nebūtinai atitinka tvarkingą žodį ir paieškos algoritmui reikia atlikti daugiau veiksmų norint surasti atitikmenį. Taip pat šiuolaikinio žodyno paieška truko vidutiniškai ~5 kartus lėčiau nei „Babel“ žodyno, tokia tendencija galėjo būti dėl to, kad šiuolaikinis žodynas (1850478 žodžių ir dažnio porų) yra ~82 kartus didesnis nei „Babel“ (22413 žodžių ir dažnio porų) žodynas.

**24 lentelė** Žodyno paieškos algoritmo „SymSpell“ paieškos veikimo laikas su testavimo žodynu. Skaičiuojama, kiek trunka pataisyti visus žodžius transkripcijose sekundėmis.

| **Bandymo nr.** | **Modelio pavadinimas** | **Šiuolaikinis žodynas** | **„Babel“ žodynas** |
|---|---|---|---|
| **1** | „Wav2Vec2 Base“ | 6,61 s | 1,30 s |
| | „Wav2Vec2 XLS-R“ | 3,90 s | 0,79 s |
| | „Google Speech to Text API“ | 1,77 s | 0,52 s |
| **2** | „Wav2Vec2 Base“ | 6,40 s | 1,11 s |
| | „Wav2Vec2 XLS-R“ | 3,89 s | 0,77 s |

| | | | |
|---|---|---|---|
| | „Google Speech to Text API“ | 1,65 s | 0,47 s |
| **3** | „Wav2Vec2 Base“ | 6,40 s | 1,11 s |
| | „Wav2Vec2 XLS-R“ | 4,14 s | 0,78 s |
| | „Google Speech to Text API“ | 1,66 s | 0,47 s |
| **4** | „Wav2Vec2 Base“ | 6,51 s | 1,11 s |
| | „Wav2Vec2 XLS-R“ | 3,99 s | 0,77 s |
| | „Google Speech to Text API“ | 1,65 s | 0,47 s |
| **5** | „Wav2Vec2 Base“ | 6,41 s | 1,09 s |
| | „Wav2Vec2 XLS-R“ | 3,99 s | 0,76 s |
| | „Google Speech to Text API“ | 1,69 s | 0,46 s |
| | | | |
| **Vidurkis** | „Wav2Vec2 Base“ | 6,466 s | 1,14 s |
| | „Wav2Vec2 XLS-R“ | 3,982 s | 0,77 s |
| | „Google Speech to Text API“ | 1,687 s | 0,48 s |

Lentelėje atspindinčioje laiko trukmę algoritmui surasti tūkstančio žodžių arba vienos minutės garso įrašo atitikmenis (žr. **25 lentelė**), matosi, kiek kurio algoritmo spėjimai truko surasti žodžių atitikmenis minėtoms imtims. Visiems sprendimams žodyno paieška truko maždaug 10 kartų ilgiau tūkstančiui žodžių, nei vienai minutei.

**25 lentelė** Vidutinė laiko trukmė „SymSpell“ algoritmo paieškai, tūkstančiui žodžių ir vienai minutei garso įrašo. Skaičiuojant, kad testavimo žodyno ilgis ~7 valandos.

| | **Žodynas** | **„Wav2Vec2 Base“** | **„Wav2Vec2 XLS-R“** | **„Google Speech to Text API“** |
|---|---|---|---|---|
| **Laikas tūkstančiui žodžių apdoroti** | „Šiuolaikinis“ | 0,15 s | 0,09 s | 0,05 s |
| | „Babel“ | 0,027 s | 0,018 s | 0,014 s |
| **Laikas vienai minutei apdoroti** | „Šiuolaikinis“ | 0,015 s | 0,009 s | 0,003 s |
| | „Babel“ | 0,0027 s | 0,0018 s | 0,0009 s |

Apjungus modelio spėjimo laiką su žodyno paieška vienai minutei (žr. **26 lentelė**) galima sakyti, jog pridėta žodyno paieška beveik neprailgina spėjimo laiko. Kadangi laikas ilgėja šimtosiomis sekundės dalimis.

**26 lentelė** Bendras sprendimų laikas vienai minutei garso įrašo atpažinti.

| **Pridėta žodyno paieška** | **„Wav2Vec2 Base“** | | | | **„Wav2Vec2 XLS-R“** | | | | **„Google Speech to Text API“** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Kompiuteris** | | **Serveris** | | **Kompiuteris** | | **Serveris** | | **Kompiuteris** | |
| | Be žodyno | Su žodynu | Be žodyno | Su žodynu | Be žodyno | Su žodynu | Be žodyno | Su žodynu | Be žodyno | Su žodynu |
| „Šiuolaikinis“ | 1,19 s | 1,205 | 1,09 s | 1,105 | 2,20 | 2,209 | 1,22 s | 1,229 | 13,00 s | 13,003 |
| „Babel“ | 1,19 s | 1,193 | 1,09 s | 1,093 | 2,20 | 2,202 | 1,22 s | 1,222 | 13,00 s | 13,001 |

## Išvados

1. Darbe buvo ištirtas dirbtiniu intelektu grįstų algoritmų panaudojimas ir jų tikslumas literatūroje. Tiksliausia rasta architektūra skirta garso įrašų į tekstą vertimui „Wav2Vec2“. Papildomai buvo atlikta technologijų analizė skirta šio algoritmo architektūrai įgyvendinti. Pasirinkta modelį ruošti „Python“ programavimo kalba, dėl kalbos patogumo ir didelio vartotojų kiekio. Taip pat buvo panaudota „HuggingFace“ modelių biblioteka palengvinanti modelio realizaciją.
2. Buvo paruošti du lietuviškos telefoninės kalbos atpažinimo modeliai „Wav2Vec2 Base“, „Wav2Vec2 XLS-R“. „Wav2Vec2 XLS-R“ modelis yra pranašesnis, nes turi daugiau pritaikomų parametrų, taip pat yra iš anksto apmokytas su didesniu kiekiu garso įrašų iš įvairių kalbų (ne tik anglų) ir yra skirtas „Wav2Vec2“ modelį lengviau pritaikyti įvairesniam spektrui kalbų.
3. Tikslumo matavimai buvo atliekami lyginant apmokytus lietuviškų telefoninių garso įrašų vertimo į tekstą modelius lyginant su egzistuojančiu komerciniu garso į tekstą vertimo sprendimu „Google Speech To Text API“. Pagal k - skirsnių analizę buvo nustatytas statistiškai reikšmingas skirtumas, pagal „Wilcoxon“ porų testą, tarp visų modelių. Abu apmokyti „Wav2Vec2“ modeliai pagal „Nemanyi“ testą veikė geriau nei komercinis sprendimas, kuris nepateko į kritinio atstumo rėžius. Matuojant žodžių klaidos dažnio (WER) metriką ant testavimo duomenų imties „Wav2Vec2 Base“ modelis veikė ~2% geriau, nei „Google Speech To Text API”, o „Wav2Vec2 XLS-R“ modelis pranoko komercinį sprendimą, beveik 30%.
4. Pasiūlytas „SymSpell“ sprendimas, siekiant pataisyti metodų garso įrašų vertimo į tekstą klaidas. Buvo panaudotos dvi metodo variacijos panaudojant skirtingus dažnių žodynus. Šiuolaikinės lietuvių kalbos žodyną, bei žodyną sudarytą iš „Babel“ treniravimo duomenų imties transkripcijų žodžių. Su k – skirsnių testu pagal „Wilcoxon“ porų testą statistiškai nereikšmingas skirtumas „Google Speech To Text API“ buvo tarp originalaus metodo ir prie jo pridėto „SymSpell“ algoritmo su „Babel“ žodyno paieška. „Wav2Vec2“ modeliuose statistiškai nereikšmingas skirtumas buvo tarp originalaus modelio ir šiuolaikinio žodyno paieška. Tad galima spėti jog „Wav2Vec2“ modeliams „Babel“ žodyno duomenys yra tinkamesni, kadangi yra iš vienodo konteksto, kaip ir mokymo ar testavimo duomenys – vartojami žodžiai yra panašūs tarpusavy. Atliekant tyrimus metodams su testavimo rinkiniu, komercinio sprendimo spėjimų „SymSpell“ algoritmas nepagerino. Tačiau „Wav2Vec2“ modelių rezultatus pagerino vidutiniškai ~2% su šiuolaikinės lietuvių kalbos dažnių žodynu ir ~5% su „Babel“ dažnių žodynu.

## Tolimesni tyrimai ir darbai

1. Patikrinti modelio tinkamumą garsynui „Liepa“, kuris skirtingai nei „IARPA Babel“ nėra sudarytas iš telefoninių skambučių ir taip pasižiūrėti, kiek tinkamas telefoninių skambučių modelis paprastiems garso įrašams. Taip pat pridedant triukšmą ar užtildant „Liepa“ garsyno įrašus.
2. Permokyti modelį, papildant garso įrašais, kurie yra ilgesni nei 5 s, juos išskaidant, kad jų trukmė būtų tinkama mokymui. Skaidant garso įrašus turėtų būti atsižvelgta į tai, kad tariamas žodis nebūtų perkirstas.

## Literatūros sąrašas


[1] Hannun A. , Case C., Casper J., Catanzaro B., Diamos G., Elsen E., Prenger R., Satheesh S., Sengupta S., Coates A., Y. Ng A. *Deep Speech: Scaling up end-to-end speech recognition*. 2015. [interaktyvus] [žiūrėta 2022-01-10] Prieiga per internetą: https://arxiv.org/pdf/1412.5567.pdf

[2] Graves A., Fernandez S., Gomez F., Schmidhuber J. *Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks.* 2006. [interaktyvus] [žiūrėta 2022-01-10] Prieiga per internetą: http://www.cs.toronto.edu/~graves/icml_2006.pdf.

[3] Morais R. *A Journey to <10% Word Error Rate*. 2017. [interaktyvus] [žiūrėta 2022-01-05] Prieiga per internetą: https://hacks.mozilla.org/2017/11/a-journey-to-10-word-error-rate/

[4] Ishaq Z. *History of computer and its generations.* 2019. [interaktyvus] [žiūrėta 2022-01-05] Prieiga per internetą: https://www.researchgate.net/publication/336700280_History_of_computer_and_its_generations

[5] Salimbajevs A., Kapočiūtė-Dzikienė J. *General-Purpose Lithuanian Automatic Speech Recognition System.* 2018. [interaktyvus] [žiūrėta 2022-01-16] Prieiga per internetą: https://www.vdu.lt/cris/bitstream/20.500.12259/59642/2/ISBN9781614999119.PG_150-157.pdf

[6] Zhang Y., Qin J., Park D. S., Han W., Chiu C., Pang R., Le Q., Wu Y. *Pushing the Limits of Semi-Supervised Learning for Automatic Speech Recognition.* 2020. [interaktyvus] [žiūrėta 2022-01-16] Prieiga per internetą: https://arxiv.org/pdf/2010.10504v1.pdf

[7] Khalid A., Sarfaraz A., Ahmed S., Malik F. *Prevalence of Stress among Call Center Employees.* 2013. [interaktyvus] [žiūrėta 2022-04-23] Prieiga per internetą: https://www.researchgate.net/publication/308203140_Prevalence_of_Stress_among_Call_Center_Employees

[8] Subramaniam L., Faruquie T., Ikbal S., Godbole S., Mohania M. *Business Intelligence from Voice of Customer.* 2009. [interaktyvus] [žiūrėta 2022-04-23] Prieiga per internetą: https://ieeexplore.ieee.org/abstract/document/4812540

[9] *Garsynas LIEPA-2.* [interaktyvus] [žiūrėta 2022-04-23] Prieiga per internetą: https://xn--ratija-ckb.lt/liepa-2/infrastrukturines-paslaugos/garsynas/

[10] Smagowska B. *Noise at Workplaces in the Call Center.* 2010. [interaktyvus] [žiūrėta 2022-04-23] Prieiga per internetą: https://acoustics.ippt.pan.pl/index.php/aa/article/viewFile/250/240

[11] *Babel.* [interaktyvus] [žiūrėta 2022-04-23] Prieiga per internetą: https://www.iarpa.gov/index.php/research-programs/babel

[12] *IARPA Babel Lithuanian Language Pack IARPA-babel304b-v1.0b.* [interaktyvus] [žiūrėta 2022-04-23] Prieiga per internetą: https://catalog.ldc.upenn.edu/LDC2019S03

[13] Salimbajevs A. *Creating Lithuanian and Latvian Speech Corpora from Inaccurately Annotated Web Data.* 2018. [interaktyvus] [žiūrėta 2022-01-17] Prieiga per internetą: https://www.aclweb.org/anthology/L18-1454.pdf

[14] Bhatt S., Jain A., Dev A. *Continuous Speech Recognition Technologies—A Review.* 2021. [interaktyvus] [žiūrėta 2022-01-17] Prieiga per internetą: https://link.springer.com/content/pdf/10.1007/978-981-15-5776-7.pdf

[15] Wang S., Li G. *Overview of end-to-end speech recognition.* 2019. [interaktyvus] [žiūrėta 2022-01-06] Prieiga per internetą: https://iopscience.iop.org/article/10.1088/1742-6596/1187/5/052068/pdf

[16] Itoh N., Kurata G., Tachibana R., Nishimura M. *A Metric for Evaluating Speech Recognizer Output based on Human-Perception Model*. 2015. [interaktyvus] [žiūrėta 2022-01-16] Prieiga per internetą: https://www.isca-speech.org/archive/interspeech_2015/papers/i15_1285.pdf

[17] Lileikytė. R, Lamel L., Gauvain J., Gorin A. *Conversational telephone speech recognition for Lithuanian.* 2018. [interaktyvus][žiūrėta 2022-01-17] Prieiga per internetą: https://www.sciencedirect.com/science/article/pii/S0885230816300523

[18] Gales M., Knill K., Ragni. *Unicode-based graphemic systems for limited resource languages.* 2015. [interaktyvus] [žiūrėta 2022-01-17] Prieiga per internetą: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7178960&tag=1

[19] Kasparaitis P. *Lithuanian Speech Recognition Using the English Recognizer*. 2008. [interaktyvus] [žiūrėta 2022-01-17] Prieiga per internetą: https://pdfs.semanticscholar.org/a147/f7038b71decb1cbf84d95062523772bf3558.pdf?_ga=2.196251116.1741263857.1610898416-2132255563.1610898416

[20] Sipavičius D., Maskeliūnas R. *"Google" Lithuanian Speech Recognition Efficiency Evaluation Research.* 2016. [interaktyvus] [žiūrėta 2022-01-17] Prieiga per internetą: https://www.researchgate.net/publication/308495820_Google_Lithuanian_Speech_Recognition_Efficiency_Evaluation_Research

[21] Nassif A. B., Shahin I., Attili, I., Azzeh, M., Shaalan, K. *Speech recognition using deep neural networks: A systematic review.* 2019. [interaktyvus] [žiūrėta 2022-01-17] Prieiga per internetą: https://www.researchgate.net/publication/330815113_Speech_Recognition_Using_Deep_Neural_Networks_A_Systematic_Review

[22] Amodei D.; Rishita A., Battenberg E.; Case C.; Casper J.; Catanzaro B.; Chen J.; Chrzanowski M.; Coates A.; Diamos G.; Elsen E.; Engel J.; Fan L.; Fougner C.; Han T.; Hannun A.; Jun B.; LeGresley P.; Lin L.; Narang S.; Ng A.; … Zhu Z. *Deep Speech 2: End-to-End Speech Recognition in English and Mandarin*. 2015. [interaktyvus][žiūrėta 2020-12-16] Prieiga per internetą: https://arxiv.org/pdf/1512.02595v1.pdf

[23] Baevski A., Zhou H., Mohamed A., Auli M., *wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations.* 2020. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://arxiv.org/pdf/2006.11477v3.pdf

[24] Schalkwyk, J., Beeferman, D., Beaufays, F., Byrne, B., Chelba, C., Cohen, M., Garret, M., Strope, B.: *Google Search by Voice: A case study.* 2010. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://research.google/pubs/pub36340/

[25] Ali A., Renals S. *Word Error Rate Estimation for Speech Recognition: e-WER.* 2018. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://www.aclweb.org/anthology/P18-2004.pdf

[26] Nguyen G., Dlugolinsky S., Bobak M., Tran V., Garcia A. L., Heredia I., Malik P., Hluchy L. *Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey.* 2019. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://link.springer.com/article/10.1007/s10462-018-09679-z?sap-outbound-id=E40D852155DBE1169062005B52A5B1209C5E32EA&utm_source=hybris-campaign&utm_medium=email&utm_campaign=000_KUND01_0000013886_SRCS_Centralized_10462&utm_content=EN_internal_30984_20190819&mkt-key=005056A5C6311ED999AA0A5933FFAAE7

[27] Dean J., Corrado G. S., Monga R., Chen K., Devin M., Le Q. V., Mao M. Z., Ranzato M. A., Senior A., Tucker P., Yang K., Ng A. Y. *Large Scale Distributed Deep Networks.* 2012. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://storage.googleapis.com/pub-tools-public-publication-data/pdf/40565.pdf

[28] He H., *The State of Machine Learning Frameworks in 2019.* 2019. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominates-industry/

[29] „Google trends PyTorch vs Tensorflow“. [žiūrėta 2020-01-17] *Prieiga per internetą:* https://trends.google.com/trends/explore?date=today%205-y&q=tensorflow,pytorch

[30] Mihajlovic S., Kupusinac A., Ivetic D., Berkovic I. *The Use of Python in the field of Artifical Intelligence.* 2020. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: http://www.tfzr.uns.ac.rs/itro/FILES/33.PDF

[31] Bhashin H. *Python Basics: A Self-Teaching Introduction.* 2019. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://www.researchgate.net/publication/331299272_Python_Basics_A_Self-Teaching_Introduction

[32] Hemanth D. J., Estrela V. V. *Deep Learning for Image Processing Applications.* 2017. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://books.google.lt/books?hl=lt&lr=&id=vsFVDwAAQBAJ&oi=fnd&pg=PR1&dq=image+processing+deep+learning&ots=-1CdGwr9pk&sig=JR8T3HRlaU5QCd59fyOswi6fE4Q&redir_esc=y#v=onepage&q&f=false

[33] Socher R., Bengio Y., Manning C. D. *Deep Learning for NLP (without Magic).* 2012. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://www.aclweb.org/anthology/P12-4005.pdf

[34] Nadkarni M. P., Machado L. O., Chapman W. W. *Natural language processing: an introduction.* 2011. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://watermark.silverchair.com/18-5-544.pdf

[35] „Tensorflow I/O“ [interaktyvus] [žiūrėta 2020-01-17] *Prieiga per internetą:* https://www.tensorflow.org/io

[36] „Audio I/O And Pre-Processing with TorchAudio“ [interaktyvus] [žiūrėta 2020-01-17] *Prieiga per internetą:* https://pytorch.org/tutorials/beginner/audio_preprocessing_tutorial.html

[37] „Mozilla DeepSpeech dokumentacija“ [interaktyvus] [žiūrėta 2020-01-17] *Prieiga per internetą:* https://deepspeech.readthedocs.io/en/latest/

[38] „OpenSeq2Seq dokumentacija“ [interaktyvus] [žiūrėta 2020-01-17] *Prieiga per internetą:* https://nvidia.github.io/OpenSeq2Seq/html/speech-recognition.html#speech-recognition

[39] Ravanelli M., Parcoller T., Bengio J. *The PyTorch-Kaldi Speech Recognition Toolkit.* 2019. [interaktyvus] [žiūrėta 2020-01-17] Prieiga per internetą: https://arxiv.org/pdf/1811.07453.pdf

[40] Wang C., Tang Y., Xutai M., Okhonko D., Pino J. *Fairseq S2T: Fast Speech-to-Text Modeling with FAISEQ.* 2020. [interaktyvus][žiūrėta 2020-01-17] Prieiga per internetą: https://www.aclweb.org/anthology/2020.aacl-demo.6.pdf

[41] Siegel J.S. Basic Concepts, and Overview of Sources, Quality of Data, and Methods. In: Demographic and Socioeconomic Basis of Ethnolinguistics. 2018. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: https://link.springer.com/chapter/10.1007/978-3-319-61778-7_1

[42] Radford A., Narasimhan K., Salimans T., Sutskever I. *Improving Language Understanding by Generative Pre-Training.* 2018. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: *https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf*

[43] Misra I., Maaten L. *Self-Supervised Learning of Pretext-Invariant Representations.* 2019. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: *https://arxiv.org/pdf/1912.01991v1.pdf*

[44] He K., Fan H., Wu Y., Xie S., Girschick R. *Momentum Contrast for Unsupervised Visual Representation Learning.* 2019. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: *https://arxiv.org/pdf/1911.05722.pdf*

[45] Liptchinsky V., Synnaeve G., Collobert R. *Letter-Based Speech Recognition With Gated ConvNets.* 2019. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: *https://arxiv.org/pdf/1712.09444.pdf*

[46] Collobert R., Puhrsch C., Synnaeve G. *Wav2Letter: an End-to-End ConvNet-based Speech Recognition System.* 2016. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: *https://arxiv.org/pdf/1609.03193.pdf*

[47] Palaz D., Collobert R., Doss M. *Estimating Phoneme Class Conditional Probabilities from Raw Speech Signal using Convolutional Neural Networks.* 2018. [interaktyvus][žiūrėta 2022-05-10] Prieiga per internetą: *https://arxiv.org/pdf/1304.1018.pdf*

[48] Dauphin Y., Fan A., Auli M., Grangier D. *Language Modeling with Gated Convolutional Networks.* 2017. [interaktyvus][žiūrėta 2022-05-15] Prieiga per internetą: *https://arxiv.org/pdf/1612.08083.pdf*

[49] Wang. H., Yang W., Zhao Z., Luo T., Wang J., Tang Y. *Rademacher dropout: An adaptive dropout for deep neural network via optimizing generalization gap.* 2019. [interaktyvus][žiūrėta 2022-05-15] Prieiga per internetą: *https://www.sciencedirect.com/science/article/pii/S0925231219306265*

[50] Panayotov V., Chen G., Povey D., Khudanpur S. *LibriSpeech: An ASR Corpus Based On Public Domain Audio Books.* 2015. [interaktyvus][žiūrėta 2022-05-15] Prieiga per internetą:

*https://www.semanticscholar.org/paper/Librispeech%3A-An-ASR-corpus-based-on-public-domain-Panayotov-Chen/34038d9424ce602d7ac917a4e582d977725d4393*

[51] Li J., Lavrukhin V., Ginsburg B., Leary R., Kuchaiev O., Cohen J. M., Nguyen H., Gadde R. T. *Jasper: An End-to-End Convolutional Neural Acoustic Model.* 2019. [interaktyvus][žiūrėta 2022-05-15] Prieiga per internetą: *https://arxiv.org/pdf/1904.03288v3.pdf*

[52] Salazar J., Liang D., Nguyen T., Kirchhoff K. *Masked Language Model Scoring.* 2020. [interaktyvus][žiūrėta 2022-05-15] Prieiga per internetą: https://www.aclweb.org/anthology/2020.acl-main.240.pdf

[53] Hugging face karkaso dokumentacija „Wav2vec2-xls-r-300m“ modelio aprašymas. [interaktyvus][žiūrėta 2022-12-30] Prieiga per internetą: https://huggingface.co/facebook/wav2vec2-xls-r-300m

[54] Babu A., Wang C., Tjandra A., Lakhotia K., Xu Q., Goyal N., Singh K., Platen P., Saraf Y., Pino J., Baevski A., Conneau A., Auli M., *XLS-R: Self-Supervised Cross-Lingual Speech Rrepresentation Learning at Scale*. 2021. [interaktyvus][žiūrėta 2022-12-30] Prieiga per internetą: https://arxiv.org/pdf/2111.09296.pdf

[55] Mihai B., *Sampling rate and aliasing on a virtual laboratory.* 2009. [interaktyvus][žiūrėta 2022-01-10] Prieiga per internetą: https://www.researchgate.net/publication/40422576_Sampling_rate_and_aliasing_on_a_virtual_laboratory

[56] Linguistic Data Consortium. *IARPA Babel Lithuanian Language Pack IARPA-babel304b-v1.0b*. 2019. [interaktyvus] [žiūrėta 2022-01-10] Prieiga per internetą: https://catalog.ldc.upenn.edu/LDC2019S03

[57] Salazar J., Kirchhoff K., Huang Z. *Self-Attention Networks For Connectionist Temporal Classification in Speech Recognition.*2019. [interaktyvus] [žiūrėta 2022-01-16]. Prieiga per internetą: https://arxiv.org/pdf/1901.10055.pdf

[58] Scheidl H., Fiel S., Sablatnig R. *Word Beam Search: A Connectionist Temporal Classification Decoding Algorithm.* 2018. [interaktyvus] [žiūrėta 2022-01-16] Prieiga per internetą: https://repositum.tuwien.at/bitstream/20.500.12708/924/2/Scheidl%20Harald%20-%202018%20-%20Word%20Beam%20Search%20A%20Connectionist%20Temporal%20Classification...pdf

[59] *Sequence Modeling With CTC.* 2017. [interaktyvus] [žiūrėta 2022-01-16] Prieiga per internetą: https://distill.pub/2017/ctc/

[60] *SymSpell.*Prieiga per internetą: https://symspellpy.readthedocs.io/en/latest/index.html

[61] Sobrino D., Soberanis M., Chin I., Cetina V. *Fixing Errors of the Google Voice Recognizer through Phonetic Distance Metrics.* 2021. [interaktyvus] [žiūrėta 2022-05-22] Prieiga per internetą: https://arxiv.org/pdf/2102.09680.pdf

[62] Mon E., Thu Y., Yu T., Oo A. *SymSpell4Burmese: Symmetric Delete Spelling Correction Algorithm (SymSpell) for Burmese Spelling Checking.* 2021. [interaktyvus] [žiūrėta 2022-05-22] Prieiga per internetą: https://ieeexplore.ieee.org/document/9678171

[63] Utka A. *Wordlist of the Contemporary Corpus of Lithuanian language.* 2016. [interaktyvus] [žiūrėta 2022-05-22] Prieiga per internetą: https://clarin.vdu.lt/xmlui/handle/20.500.11821/8

[64] Demšar J. *Statistical Comparisons of Classifiers over Multiple Data Sets.* 2006. [interaktyvus] [žiūrėta 2022-05-22] Prieiga per internetą: https://www.jmlr.org/papers/volume7/demsar06a/demsar06a.pdf

**Informacijos šaltinių sąrašas**

**Priedai**