Kauno technologijos universitetas

Informatikos fakultetas

Sintetinės ir manipuliuotos kalbos atpažinimo garso įrašuose tyrimas

Magistro baigiamasis projektas

Modestas Butnorius

Projekto autorius

asist. Audrius Nečiūnas

Vadovas

Kaunas, 2026

Kauno technologijos universitetas

Informatikos fakultetas

Sintetinės ir manipuliuotos kalbos atpažinimo garso įrašuose tyrimas

Magistro baigiamasis projektas

Dirbtinio intelekto informatika (6211BX007)

Modestas Butnorius

Projekto autorius

asist. Audrius Nečiūnas

Vadovas

prof. Dalia Čalnerytė

Recenzentė

Kaunas, 2026

Kauno technologijos universitetas

Informatikos fakultetas

Modestas Butnorius

Sintetinės ir manipuliuotos kalbos atpažinimo garso įrašuose tyrimas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Modestas Butnorius

Patvirtinta elektroniniu būdu

Butnorius, Modestas. Sintetinės ir manipuliuotos kalbos atpažinimo garso įrašuose tyrimas. Magistro baigiamasis projektas / projektui vadovavo asist. Audrius Nečiūnas; Kauno technologijos universitetas, informatikos fakultetas.

Studijų kryptis ir studijų krypčių grupė: Informatikos mokslai, Informatika (B01).

Reikšminiai žodžiai: klastoto garso aptikimas, kalbos generavimas, priešiška ataka, triukšmo mažinimas, kalbos aktyvumo aptikimas, ansamblinis klasifikavimas.

Kaunas, 2026. 95 p.

Santrauka

Šiame darbe tiriamos klastotų balso įrašų generavimo ir aptikimo technologijos. Eksperimentų metu išbandyti įvairūs kalbos generavimo metodai, o jų sugeneruoti įrašai įvertinti naudojant egzistuojantį realiai taikomą garso klastočių aptikimo sprendimą. Darbe ypatingas dėmesys skirtas priešiškoms atakoms, kurios pastaruoju metu tapo itin aktualia problema garso klastočių aptikimo srityje. Tyrimo metu buvo išbandytos kelios priešiško triukšmo generavimo technologijos bei skirtingos triukšmo mažinimo strategijos. Taip pat buvo analizuojami kalbos aktyvumo aptikimo metodai, skirti iš garso įrašų išskirti kalbos segmentus. Galiausiai, buvo ištirtos įvairios garso įrašų klasifikavimo architektūros, įskaitant CNN, RNN ir transformerių modelius.

Remiantis atliktų eksperimentų rezultatais, sukurtas apjungtas garso klastočių aptikimo sprendimas, sudarytas iš trijų pagrindinių komponentų: triukšmo mažinimo, kalbos segmentų išskyrimo bei klasifikavimo algoritmo. Klasifikavimo dalyje naudojamas ansamblinis metodas, apjungiantis dešimt skirtingų dirbtinio intelekto modelių, apmokytų naudojant skirtingus garso požymius, kurių prognozės sujungiamos į vieną galutinį sprendimą. Darbe pasiūlytas sprendimas geba efektyviai aptikti klastotus garso įrašus net ir esant priešiškam triukšmui bei pasiekia geriausius rezultatus pagal DCF metriką, lyginant su kitais sprendimais, pateiktais specialiai šiam uždaviniui skirtame iššūkyje.

Butnorius, Modestas. Research of Detection of Synthetic and Manipulated Speech in Audio Recordings. Master‘s Final Degree Project / supervisor dr. assist. Audrius Nečiūnas; Faculty of Informatics, Kaunas University of Technology.

Study field and study field group: Computer science, Informatics (B01).

Keywords: audio spoof detection, speech generation, adversarial attacks, noise reduction, voice activity detection, ensemble classification.

Kaunas, 2026. 95 p.

Summary

This thesis investigates technologies for generating and detecting spoofed voice recordings. During the experiments, various speech generation methods were investigated, and the recordings they produced were evaluated using an existing real-world audio spoofing detection solution. Attention was dedicated to adversarial attacks, which have recently become a highly relevant problem in the field of audio spoofing detection. Several adversarial noise generation techniques and different noise reduction strategies were evaluated during the research. In addition, speech activity detection methods designed to extract speech segments from audio recordings were analyzed. Finally, various audio classification architectures were investigated, including CNN, RNN, and transformer-based models.

Based on the results of the conducted experiments, a combined audio spoofing detection solution was developed, consisting of three main components: noise reduction, speech segment extraction, and a classification algorithm. The classification component employs an ensemble approach combining ten different artificial intelligence models trained using different audio features, whose predictions are merged into a single final decision. The solution proposed in this thesis is capable of effectively detecting spoofed audio recordings even in the presence of adversarial noise and achieves the best results according to the DCF metric when compared with other solutions presented in a challenge specifically designed for this task

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

Santrumpos:

ASV (angl. *Automatic Speaker Verification* – Automatinis kalbėtojo tapatybės patvirtinimas) – sistema, skirta automatiškai nustatyti arba patvirtinti asmens tapatybę pagal jo balso charakteristikas.

TTS (angl. *Text-to-Speech* – Teksto vertimas į kalbą) – technologija, leidžianti automatiškai generuoti žmogaus balsą iš tekstinės informacijos.

VC (angl. *Voice Conversion* – Balso konvertavimas) - tai metodas, skirtas pakeisti vieno žmogaus balsą taip, kad jis skambėtų kaip kito asmens balsas, išlaikant pradinį kalbos turinį.

CNN (angl. *Convolutional Neural Network* – Konvoliucinis neuroninis tinklas) – tai giluminio mokymosi modelis, dažniausiai naudojamas vaizdų, garso signalų ir kitų struktūrizuotų duomenų požymių išgavimui bei klasifikavimui.

RNN (angl. *Recurrent Neural Network* – Rekurentinis neuroninis tinklas) –neuroninio tinklo architektūra, skirta nuoseklių duomenų apdorojimui, gebanti išlaikyti informaciją apie ankstesnius įvesties elementus.

DCF (angl. *Detection Cost Function* – Aptikimo kainos funkcija) – metrika, naudojama klasifikavimo sistemų veikimui vertinti, atsižvelgianti į skirtingų klaidų tipų kainą ir jų tikimybes.

EER (angl. *Equal Error Rate* – Vienodų klaidų rodiklis) – biometrinių ir klasifikavimo sistemų vertinimo metrika, nusakanti tašką, kuriame klaidingo priėmimo ir klaidingo atmetimo rodikliai yra lygūs.

ROF (angl. *Rudin-Osher-Fatemi* – Rudino-Ošerio-Fatemi metodas) – triukšmo mažinimo metodas, naudojamas signalų ir vaizdų apdorojime, pagrįstas variacinio optimizavimo principu, siekiant sumažinti triukšmą išsaugant svarbias signalo savybes. Dažniausiai šis metodas vadinamas triukšmo mažinimo pagal bendrąją variaciją (dispersiją).

Terminai:

Klastotas balsas (angl. *Spoofed audio*) – dirbtinai sugeneruotas, modifikuotas arba manipuliuotas garso įrašas, skirtas imituoti tikrą žmogaus balsą ar suklaidinti balso atpažinimo sistemas.

Tikras balsas (angl. *Bona-fide audio*) – natūralus, nemodifikuotas žmogaus balso įrašas, nelaikomas dirbtinai sugeneruotu ar suklastotu.

Priešiška ataka (angl. *Adversarial attack*) – metodas, kuriuo į duomenis ar signalus įterpiami nedideli, žmogui sunkiai pastebimi pakeitimai, siekiant suklaidinti mašininio mokymosi modelį ar sumažinti jo tikslumą.

Transformeris (angl. *Transformer*) – giluminio mokymosi architektūra, paremta dėmesio mechanizmu (angl. *attention mechanism*), plačiai naudojama natūralios kalbos, garso ir kitų sekų duomenų apdorojimui.

Įvadas

Darbo problema:

Tobulėjant dirbtinio intelekto technologijoms, sparčiai vystosi ir garso generavimo metodai, leidžiantys sukurti itin realistiškai skambantį žmogaus balsą. Šiuolaikiniai balso sintezės bei balso konvertavimo metodai gali generuoti kalbą, kuri klausytojui tampa sunkiai atskiriama nuo tikro žmogaus balso. Nors šios technologijos gali būti pritaikomos naudingose srityse, pavyzdžiui, virtualių asistentų kūrime, filmų įgarsinime ar prieinamumo sprendimuose, jos taip pat sudaro prielaidas piktnaudžiavimui. Vienas iš aktualiausių piktnaudžiavimo pavyzdžių yra balso imitavimas, kai dirbtinai sugeneruotas arba modifikuotas balsas naudojamas apsimetant kitu asmeniu. Tokios technologijos gali būti naudojamos dezinformacijai skleisti, sukčiavimui vykdyti ar biometrinių autentifikavimo sistemų apsaugai apeiti. Dėl sparčios šių metodų pažangos paprastiems vartotojams tampa vis sudėtingiau įvertinti internete randamų garso ir vaizdo įrašų autentiškumą, o tradiciniai saugumo metodai ne visada geba patikimai aptikti dirbtinai sugeneruotą turinį.

Darbo aktualumas:

Dėl problemoje paminėtų priežasčių pastaraisiais metais didelis dėmesys skiriamas klastoto balso aptikimo (angl. *audio spoof detection*) tyrimams. Šios srities tikslas – sukurti metodus, galinčius automatiškai atskirti tikrą žmogaus balsą nuo dirbtinio intelekto metodais sugeneruoto arba pakeisto balso. Garso klastočių aptikimas tampa ypač svarbus balso biometrikos, skaitmeninio saugumo ir informacijos patikimumo užtikrinimo srityse. Verta paminėti, kad dėl nuolat tobulėjančių balso klastojimo technologijų ir technikų negalimas vienas geriausias aptikimo sprendimas. Pastoviai reikia ieškoti naujų algoritmų ir strategijų klastotiems garsams aptikti.

Darbo tikslas – sukurti metodą, leidžiantį patikrinti, ar balso įrašas buvo sugeneruotas dirbtinai, ar kokiu nors kitu būdu manipuliuotas.

Darbe bus analizuojami egzistuojantys metodai, skirti dirbtinio intelekto pagrindu sugeneruoto balso aptikimui, vertinamas jų efektyvumas bei siekiama pasiūlyti tikslesnį sprendimą garso klastočių aptikimo uždaviniui spręsti.

Darbo uždaviniai:

1. atlikti klastoto balso aptikimo uždavinio literatūrinę apžvalgą;
2. suformuoti sprendimą garsų klasifikavimui;
3. patikrinti priešiškos atakos įtaką balso klasifikavimui ir jos apėjimo metodus;
4. atlikti geriausio balso klasifikavimo metodo parinkimo eksperimentus;
5. pagal atliktų eksperimentų rezultatus, realizuoti suformuotą sprendimą;
6. ištestuoti ir palyginti galutinį sprendimą su jau egzistuojančiais sprendimais.

Darbo naujumas:

Tyrime siūlomas sprendimas susideda iš trijų komponentų, kurių kiekvieno veikimo principas skiriasi nuo egzistuojančių ir praktikoje dažnai taikomų sprendimų:

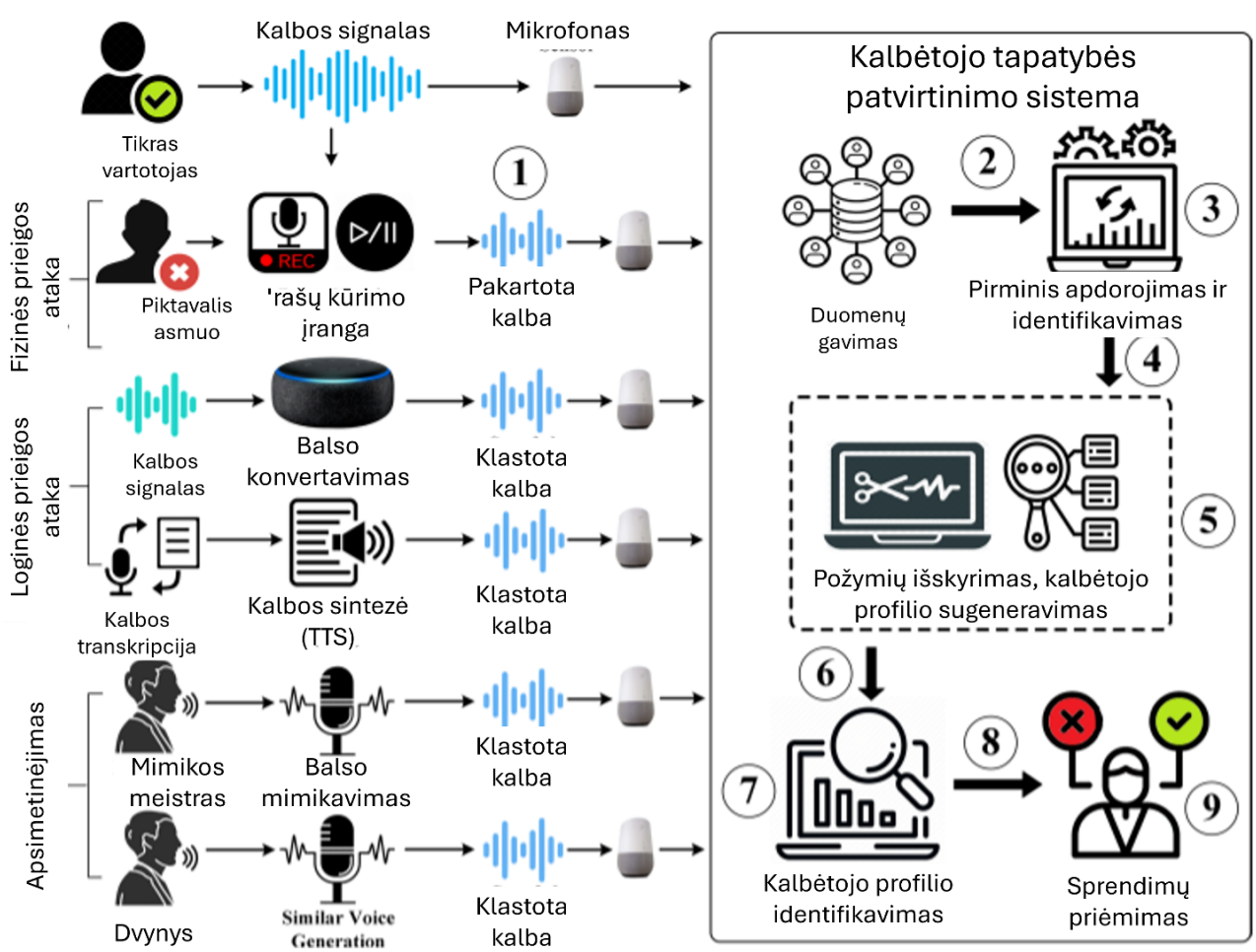
1. priešiško triukšmo mažinimas pasitelkus ROF algoritmą;
2. garso failo padalijimas išskiriant atskirus girdimus žodžius;
3. klasifikavimas balsavimo principu, kur atskirai suklasifikuoti išskirti žodžiai nurodo galutinę viso garso failo klasę.

Dokumento struktūra:

Darbas susideda iš penkių pagrindinių dalių: klastotos kalbos aptikimo literatūros apžvalgos, kuriamo sprendimo vizijos, garso paruošimo eksperimentų, garso klasifikavimo eksperimentų ir apjungto sprendimo. Pirmoje dalyje atliekama visa su šia tema susijusių literatūros šaltinių apžvalga. Šiame skyriuje analizuojama pati balso aptikimo uždavinio problema, su kokiais iššūkiais susiduria ją sprendžiantys sprendimai ir tų sprendimų veikimas. Papildomai, analizuojamos kitos tyrimui aktualios temos, kaip priešiška ataka, žodžių išskyrimas ir balso generavimo technologijos. Antrojoje dalyje aprašoma darbe kuriamo sprendimo vizija. Čia galima rasti šio sprendimo išsikeltus reikalavimus ir vertinimo kriterijus. Trečiojoje dalyje aprašomi visi, su garso failų apdorojimu ir paruošimu susiję, atlikti eksperimentai ir jų rezultatai. Čia aprašomas, darbui pasirinktas, duomenų rinkinys ir kaip jis buvo papildytas. Be duomenų rinkinio, ši dalis labiausiai fokusuojasi į du pagrindinius aspektus – priešišką ataką ir garso padalijimą. Ketvirtoji dalis yra skirta patiems garso klasifikavimo eksperimentams. Abi eksperimentų dalys pradžioje turi bendrą aprašymą, kuris nurodo, kaip atlikti eksperimentai yra susiję vieni su kitais. Galiausiai, paskutinėje dalyje aprašomas darbe įgyvendintas galutinis sprendimas. Čia galima rasti šio sprendimo testavimo rezultatus ir jo palyginimą su jau egzistuojančiais sprendimais. Galiausiai, darbas užbaigiamas išvadomis, kur aptariamos svarbiausios darbo dalys ir pastebėjimai, kurie iškilo atliekant kiekvieną užsibrėžtą užduotį.

1. Klastotos kalbos aptikimo literatūrinė apžvalga

Pati generuoto balso aptikimo užduotis yra dalis balso apsaugos nuo parodijos uždavinio [1]. Praktikoje šis uždavinys naudojamas automatinėse kalbėtojo tapatybės patvirtinimo sistemose (angl. *Automatic speaker verification*), kurios nustato, ar kalba koks nors specifinis asmuo, kurio biometriniai balso duomenys yra žinomi sistemoje. Balso parodijavimas yra uždavinys, kuris standartiškai bando apeiti šias sistemas [2] ir susideda iš trijų pagrindinių atakų tipų (žr. **1 pav.**).



1 pav. Balso parodijavimo atakų tipai [1]

Pirmoji iš šių atakų yra fizinės prieigos ataka [1]. Atliekant šią ataką, piktavalis žmogus kokiu nors būdu gauna parodijuojamo asmens garso įrašą. Norint prieiti prie kokios nors saugomos sistemos, žmogus kalbėtojo atpažinimo sistemoje paleidžia turimą garso įrašą, tikėdamasis, kad pasitelkus garso įrašą pavyks įsilaužti į sistemą. Dėl savo veikimo pobūdžio, ši ataka yra lengvai aptinkama, kadangi pakanka apmokyti modelį, kuris turi išmokti aptikti, ar girdimas garsas leidžiamas per kokį nors garsiakalbį.

Antrasis atakos tipas yra apsimetinėjimas (angl. *Impersonation*) [1]. Ši ataka inicializuojama realaus asmens, kuris turi panašų balsą į imituojamą asmenį. Toks asmuo gali būti žinomo žmogaus dvynys arba tiesiog koks nors mimikos meistras. Šiai atakai aptikti sistema turi gebėti labai tiksliai atskirti žinomo žmogaus kalbėjimo manierą, toną ir visas kitas savybes, kurios padeda atskirti skirtingų žmonių balsus.

Galiausiai, trečia ir šiam darbui aktualiausia ataka yra loginės prieigos ataka [1, 3]. Ši ataka įvairiais sintetiniais balso generavimo metodais sugeneruoja pasirinkto kalbėtojo balsu sakomą frazę. Pastaruoju metu pradėjo smarkiai tobulėti dirbtinio intelekto algoritmai, kurie gali sugeneruoti vis geresnius ir realistiškesnius garso įrašus. Dėl to reikia vis naujesnių ir sudėtingesnių kalbėtojo atpažinimo sistemų, kurios gali aptikti, ar girdimas garsas yra sugeneruotas dirbtiniu būdu. Papildomai, balsus generuojantys algoritmai gali pasitelkti technikas, kurios manipuliuoja sugeneruotą balsą taip, kad būtų galima apeiti specifines kalbėtojo atpažinimo sistemas. Viena iš labiau paplitusių technikų yra priešiška ataka (angl. *Adversarial attack*) [4], kuri į garso įrašą įterpia šiek tiek triukšmo, kurio pagalba galima apgauti pasirinktas kalbėtojo tapatybės patvirtinimo sistemas.

Apsaugos nuo parodijos uždaviniui kas kelis metus yra organizuojamas iššūkis *ASVspoof* [5]. Šis iššūkis paskutinį kartą buvo organizuojamas 2024 metais. Iš esmės visi atvirai prieinami egzistuojantys sprendimai, apsaugos nuo parodijos uždaviniui, naudoja šio iššūkio duomenų rinkinius ir savo veiksmingumui įvertinti naudoja iššūkyje aprašomas metrikas. Iššūkiams kuriami sprendimai organizavimo metu pasiekia labai gerus rezultatus, tačiau po tam tikro laiko, atsiradus naujesniems balso generavimo algoritmams ir technologijoms, tie patys sprendimai tampa neveiksmingi ir lengvai apeinami [6].

1.1. Balso generavimo technologijos

Balso generavimo technologijos gali būti išskirstytos į dvi pagrindines technikas [1]: kalbos sintezę (angl. *Speech synthesis*) ir balso konvertavimą (angl. *Voice conversion*).

1.1.1. Kalbos sintezė

Pirmoji technika yra garso sintezė su balso klonavimu [7, 8, 9]. Šio tipo algoritmai tam tikrą teksto įvestį paverčia garso įrašu, kur tam tikru balsu sakomas įvestas tekstas. Šio tipo algoritmai gali būti tiesiogiai apmokyti pasirinkto kalbėtojo balsu. Tačiau pastaruoju metu pradėjo atsirasti įvairūs algoritmai, kurie su tekstine įvestimi papildomai naudoja garso įrašo įvestį, kuri yra naudojama kaip generuojamo balso pavyzdys. Tokie algoritmai gali sugeneruoti didelę įvairovę įvairių balsų. Didžioji dalis teksto sintezės modelių gali būti padalintos į dvi dalis: autoregresinius ir difuzinius.

1.1.2. Autoregresiniai modeliai

Standartiniai teksto generavimo algoritmai yra autoregresiniai (angl. *autoregressive)*. Tai reiškia, kad generuojant kokią nors garso seką modelis pirma turi sugeneruoti sekoje pirmiau einančią garso signalo vertę, prieš generuojant sekantį garso signalo momentą.

Vienas iš naujesnių autoregresinių kalbos sintezės su balso klonavimu algoritmų yra *XTTS* [7] (angl. *eXtended Text-To-Speech*) architektūra. Algoritmas sugeba sugeneruoti itin tikroviškai skambantį balsą įvairiomis kalbomis. Norėdamas sintezuoti balsą, modelis naudoja teksto įvestį ir pavyzdinį kalbėtojo garso failą. Garso failas gali būti bet kokio ilgio, tačiau ilgesnis garso failas užtikrina geresnę sugeneruoto garso failo kokybę. Bendram naudojimui pakanka 5-10 sekundžių trukmės failo. Vienas didžiausių šio modelio privalumų yra tai, kad jis palaiko kelias kalbas balso generavimui. Pagal numatytus nustatymus modelis palaiko 16 kalbų, tačiau pamodifikavus algoritmo failus galima pridėti daugiau. Šis algoritmas naudoja kodelius (angl. *encoder*), dekoderius (angl. *decoder*) ir *GPT* transformerius [10] (angl. *transformer*), kurie išmoksta žodyno faile žinomas fonemas priskirti prie

realių balse girdimų garsų. Algoritmui paduodamas tekstas yra tokenizuojamas ir kiekvienas tokenas (žetonas) priskiriamas išmoktam garsui. Galiausiai, iš pavyzdinio balso failo išgautos kalbėtojo informacijos vektorius ir tekstui priskirti garsai yra apjungiami ir, pasitelkiant *GPT-2* [11] koderį, gaunamas galutinis garso įrašo vektorius, kuris yra paduodamas vokoderiui [12] (angl. *vocoder*), kuris vektorių paverčia garso failu.

1.1.3. Difuziniai modeliai

Dėl savo veikimo principo autoregresinio kalbos generavimo būdo greitaveika yra lėta, bandant generuoti ilgesnius garso įrašus. Dėl to buvo pradėta ieškoti greitesnių garso sintezės sprendimų. Galiausiai, buvo atrasta, kad difuziją taikantys modeliai gali sėkmingai sugeneruoti garso seką. Dažniausiai su kokiu nors difuziniu vaizdų generavimo modeliu bandoma sugeneruoti dvimatę garso reprezentaciją, kaip spektrogramą, po to sugeneruotam vaizdui atliekama korekcija, kuri leidžia sugeneruotą paveikslėlį paversti garsu.

Vienas tokių modelių yra *ProDiff* [13], kuris kalbai generuoti pritaiko difuzijos ir generatyvinius konkuruojančius tinklus (angl. *Generative adversarial networks* arba *GAN*). Modelis yra apmokomas specifinio kalbėtojo balsu, dėl to norint sugeneruoti kitokio kalbėtojo balsą, modelį reikia apmokyti iš naujo arba atlikti derinimą (angl. *Fine tune*) jau apmokytam modeliui. Modelis įvestą tekstą paverčia fonemų reprezentacijomis, iš kurių suformuojama pradinė spektrograma. Į šią spektrogramą įterpiamas triukšmas, kuris vėliau difuzijos modelių pagalba po kelių triukšmo mažinimo (angl. *Denoising*) etapų yra išvalomas. Galiausiai, gaunama švari spektrograma, kuri reprezentuoja pasirinkto kalbėtojo balsu sakomus tekstu įvestus žodžius. Modelio mokyme naudojamas diskriminatorius, kuris įvertina sugeneruotą spektrogramą ir teikia grįžtamąjį ryšį pačiam generavimo modeliui. Galiausiai, spektrograma pateikiama vokoderiui [12], kuris spektrogramą paverčia garso įrašu.

1.1.4. Balso konvertavimas

Antroji technika yra balso konvertavimas [14]. Šios technikos veikimo principas yra pakeisti jau egzistuojančiame garso įraše girdimo balso ypatybes. Kaip ir su garso sinteze, modeliai gali būti apmokomi specifiniu balsu arba naudoti pavyzdinius garso įrašus balsui suklastoti. Vienas iš populiariausių kalbos konvertavimo metodų yra *RVC* [14, 15] (angl. *Retrieval-based-Voice-Conversion*). Algoritmas perkelia tam tikro nurodyto kalbėtojo balso charakteristikas į įvestą garso įrašą. Šiam uždaviniui atlikti naudojamas koderis, kuris iš įvesties balso išgauna kalbos lingvistinius požymius ir pašalina originaliam kalbėtojui būdingą informaciją. Algoritme naudojamas paieškos mechanizmas atranda panašias akustines ypatybes iš pasirinkto žinomo kalbėtojo balso duomenų rinkinio ir pateikia reikalingus kalbėtojo įterpinius (angl. *embeddings*), kurie yra derinami su išskirtomis lingvistinėmis savybėmis. Kad modelis gerai apibendrintų skirtingus balsus, modelyje naudojamas didelio masto mokymas iš įvairių duomenų rinkinių. Naudojant paieška grįstus metodus, *RVC* pasiekiamas aukštos kokybės balso konvertavimas, todėl galima tikroviškai ir išraiškingai susintezuoti įvairių kalbėtojų balsus. Verta paminėti, kad įvairių kalbėtojų balso įterpiniai yra atvirai prieinami. Šiam modeliui yra sukurta atskira kalbėtojų įterpinių repozitorija [16], kur šio tyrimo vykdymo metu galima rasti virš 27900 įvairių kalbėtojų balso informaciją.

1.2. Žodžių išskyrimo technologijos

Kadangi tyrime bus bandoma sukurti sprendimą, kuris tikrins, ar jam pateiktuose garso įrašuose girdima klastota kalba, verta patyrinėti technologijas, kurios iš garso įrašų gali išskirti atskirus sakomus žodžius.

1.2.1. Balso aktyvumo aptikimas

Balso aktyvumo aptikimas (angl. *voice activity detection* arba *VAD*) yra garso apdorojimo metodas, kurio pagrindinė paskirtis yra nustatyti, ar garso signale yra žmogaus kalbos segmentai, ar kitokie kalbos neturintys segmentai, tokie kaip tyla, muzika ar kitas pašalinis triukšmas. Tradicinės *VAD* [17] sistemos standartiškai veikia analizuodamos trumpus garso kadrus, iš jų išskirdamos paprastus akustinius požymius, kaip signalo energiją (amplitudę), nulio perėjimo dažnį ar spektrinę entropiją. Vėliau šie požymiai lyginami su iš anksto nustatytomis slenkstinėmis vertėmis arba statistiniais modeliais, ir kiekvienas garso kadras suklasifikuojamas kaip kalba arba ne kalba. Naujesni *VAD* metodai [18] naudoja įvairius mašininio mokymo algoritmus, kaip konvoliucinius arba rekurentinius neuroninius tinklus, kurie garso požymius išmoksta atskirti tiesiogiai iš duomenų. Galutinis metodo rezultatas standartiškai yra dvejetainių sprendimų seka, nurodanti laiko intervalus, kuriuose garso signalų segmentuose yra kalba arba jos nėra.

1.2.2. Garso įrašų transkribavimas

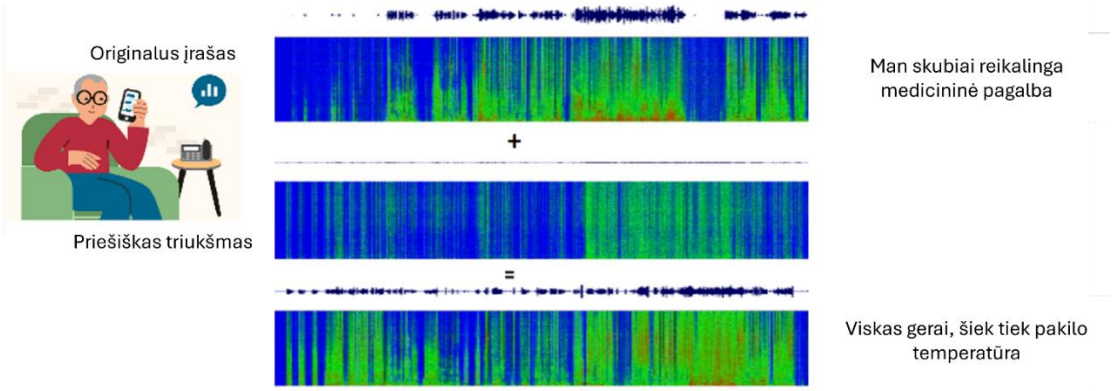
Kitas būdas garso įrašuose išskirti atskirus sakomus žodžius yra garso transkribavimo modeliai. Šių modelių pagrindinė paskirtis yra atrasti, kas tiksliai sakoma garse, tačiau dažnai modeliai kaip papildomą išvestį grąžina atskirų žodžių tarimo laiko intervalus. Vienas tokio modelio pavyzdys yra *Whisper* [19]. Šis algoritmas yra didelio masto automatinio kalbos atpažinimo (angl. *automatic speech recognition* arba *ASR*) modelis, skirtas transkribuoti ir versti garso įrašuose įvairiomis kalbomis ir sąlygomis sakomus žodžius. Modelio architektūra pagrįsta transformerių kodavimo ir dekodavimo etapais. Įvestas garso įrašas iš pradžių paverčiamas melų (melas – matavimo vienetas) skalės spektrograma, kuri paskui paduodama kodavimo sluoksniams. Kodavimo žingsnyje spektrograma užkoduojama ir yra paverčiama aukštos svarbos latentinių savybių seka, kuri atspindi tiek lokalią, tiek ilgalaikę garso sekos informaciją. Tada dekoderis autoregresyviai generuoja atitinkamą teksto išvestį. Kiekvienam žodžiui prognozuoti dekoderis naudoja kryžminio dėmesio mechanizmą, kuris patikrina visas koduotojo išvestis ir iš jų randa, kurios garso dalys yra svarbiausios sekančio žodžio prognozei. Mokymo metu be garso ir transkripcijų susiejimo modelis papildomai yra mokomas atlikti pagalbines užduotis, kaip kalbos vertimą į kitas kalbas arba kalbamos kalbos identifikavimą. Toks mokymo būdas padidina modelio patikimumą ir jo daugiakalbį lankstumą.

1.3. Priešiška ataka

Vienas iš pagrindinių iššūkių, su kuriais šiandien susiduria automatinės kalbėtojo tikrinimo sistemos, yra priešiškų atakų (angl. *adversarial attack*) grėsmė [4, 5, 20, 21, 22, 23]. Šios atakos yra bandymai manipuliuoti mašininio mokymo modelių išvestis, šiek tiek modifikuojant šiems modeliams pateikiamus įvesties duomenis. Šiame skyriuje detaliau aprašomas jų veikimas ir kaip jos gali būti sprendžiamos.

1.3.1. Priešiškas triukšmas

Priešiškos atakos yra dažnai taikomos vaizdų generavimo uždaviniuose, tačiau jas taip pat galima taikyti teksto generavimo, natūralios kalbos apdorojimo ir balso atpažinimo uždaviniuose. Generuoto balso aptikimo sistema apibendrinta žemiau (žr. **2 pav.**). Su pasirinktu balso generavimo modeliu yra sugeneruojamas garso įrašas. Tada į sugeneruotą įrašą įterpiami žmogaus ausiai minimaliai girdimi trikdžiai arba triukšmas. Šis veiksmas smarkiai patobulina balso sugeneruotą balso klastotę. Pateikus manipuliuotą generuotą balsą, kalbėtojo aptikimo sistema išveda neteisingą rezultatą.



2 pav. Priešiškos atakos pavyzdys balso apdorojime [4]

Prieš bandant atlikti priešišką ataką, asmuo turi sužinoti informaciją apie mašininio mokymo modelį, kurį bandoma apeiti. Žinant modelio architektūrą, mokymo parametrus, svorius ar apmokymo duomenis, galima suformuluoti tokią strategiją, kuri geriausiai sugebės apeiti tą modelį. Tada su išgauta informacija galima parengti optimalią triukšmo generavimo strategiją. Tokia atakos kategorija vadinama baltos dėžės ataka. Jei piktavalis asmuo nežino jokios informacijos apie atakuojamą modelį, jis vis tiek gali gauti naudingos informacijos iš to modelio išvesčių. Modeliui galima pateikti bet kokį garso įrašą ir stebėti, kas bus išvedama. Prieš pateikiant naują garso įrašą, į jį įterpiamas triukšmas, ir toliau stebima, kas bus gauta. Pagal gautą išvestį triukšmas yra šiek tiek pamodifikuojamas ir vėl vykdomas praeitas žingsnis iki tol, kol pasirinktas modelis bus įveiktas. Tokia strategija vadinama juodos dėžės ataka [4, 20].

Priešiškos atakos gali būti specifikuotos, kur tikimasi iš modelių išgauti specifinį rezultatą. Papildomai, atakos gali būti nespecifikuotos, kur tikimasi, kad bus gautas bet koks kitas rezultatas negu pasirinktas. Šiam darbui aktualios specifikuotos atakos, kadangi tyrime bus bandoma sukurti sistemą, kuri aptiks, ar jai pateiktas garso failas buvo tikrai sugeneruotas taikant dirbtinio intelekto metodus, ar ne.

Galiausiai, triukšmui generuoti galima naudoti kelias skirtingas technikas [4]. Viena jų yra genetiniai algoritmai. Šiuo metodu generuojamos triukšmų populiacijos. Sugeneruotos populiacijos nariai yra įvertinami, ir tada su geriausiais populiacijos nariais kuriamos naujos, geresnės triukšmų populiacijos. Šios operacijos kartojamos tol, kol pasiekiamas norimas populiacijos narys. Kita technika yra gradientine paieška paremti metodai. Šiai technikai reikalinga informacija apie mašininio mokymo modelį, kurį bandoma apgauti. Taikant greitojo gradiento ženklo metodą [21] (angl. *Fast Gradient Sign Method* arba *FGSM*) apskaičiuojamas tinkamas triukšmas, taikant paties modelio gradientą. Pati *FGSM* ataka gali būti aprašyta pagal žemiau pateiktą formulę (žr. 1 formulė):

$$\boldsymbol{x_{adv}} = \boldsymbol{x} + \epsilon * sign(\boldsymbol{\nabla}_x loss(\boldsymbol{\theta}, \boldsymbol{x}, y)) \quad (1)$$

čia $\boldsymbol{x}$ yra duomenų įvestis, $\boldsymbol{\theta}$ yra modelio parametrai, y – modelio išvestis, ϵ – kokia nors maža vertė, $\boldsymbol{x}_{adv}$ – priešiška ataka paveiktas garso signalas.

Atakai reikia turėti prieigą prie paties klasifikatoriaus, kuriam pateikiama garso įvestis x. Pagal modelio išvestį ir modelio parametrus apskaičiuojama tam tikra pasirinkta klaidos funkcija. Tada suskaičiuojamas šios klaidos funkcijos gradientas, kurio toliau naudojama tik ženklo funkcija. Funkcijos išvestis sudauginama su tam tikra mažos vertės ϵ ir yra pridedama prie originalaus garso. Taip išgaunama triukšminga informacija, kurią pateikus klasifikatoriui su parametrais θ, išgaunamas rezultatas yra artimesnis priešingai (binariniai) išvesčiai nuo originalios y. Esant didesnėms ϵ vertėms, ataka yra agresyvesnė ir išvestis yra labiau nutolusi nuo originalios išvesties, tačiau peržiūrėjus pamodifikuotą informaciją, akivaizdžiai matoma, kad ji buvo manipuliuota. Dėl to naudojama maža ϵ vertė, ir pati ataka taikoma daug kartų iki, kol klasifikatorius nebesugeba teisingai suklasifikuoti originalios įvesties x.

Galiausiai, galima taikyti optimizavimu paremtus metodus. Šiam metodui, kaip ir gradientu paremtuose metoduose, reikia žinoti pilną modelio architektūros ir parametrų informaciją. Šiuo metodu galima naudoti bet kokį optimizavimo algoritmą (pvz. Gradientinį nusileidimą), kad per kelias iteracijas surastume patį mažiausią galimą triukšmą, kuris sugeba apgauti norimą sistemą.

1.3.2. Priešiško triukšmo mažinimas

Apsaugos nuo priešiškos atakos metodai gali būti paskirstyti į proaktyvius ir reaktyvius [4, 24]. Reaktyviai gynybai mokomi specifiškai priešiškoms atakoms aptikti skirti modeliai. Šie modeliai sėkmingai susidoroja su egzistuojančiomis priešiškomis atakomis, bet atsiradus naujoms atakoms reikia mokyti naują modelį. Kitoks reaktyvus gynybos būdas yra natūralios kalbos apdorojimo metodai, skirti pateiktų garso įrašų transkripcijoms išgauti. Tokie sprendimai naudoja specialias metrikas, kurios įvertina transkripcijų teisingumą. Kadangi priešiška ataka į garso įrašą įterpia triukšmą, tikėtina, kad transkripcijos modeliai pradės daryti klaidas bandant transkribuoti triukšmo paveiktus žodžius. Jei pateiktam garso įrašui apskaičiuojamas slenkstinės vertės neviršijantis „teisingumo“ įvertis, jį galima traktuoti kaip suklastotą.

Proaktyvia gynyba bandoma tobulinti pačias kalbėtojo aptikimo sistemas. Vienas šių būdų yra priešiška ataka modifikuotus garso įrašus pridėti į sistemos apmokymo duomenų imtį [25]. Taip tikimasi, kad modelis išmoks teisingai klasifikuoti net manipuliuotas balso klastotes. Kitas proaktyvios gynybos būdas yra patikimų garso sekų paieška garso failuose. Neradus patikimos sekos, galima teigti, kad ir pateiktas garso įrašas yra nepatikimas.

Galiausiai, galima bandyti sumažinti triukšmą iš garso signalų, prieš juos pateikiant klasifikatoriams. Tai galima daryti taikant įvairius įprastus triukšmo mažinimo metodus, bet jie dažnai nepasiteisina, kadangi priešiškas triukšmas skiriasi nuo paprasto gausinio (angl. *Gaussian*) ar balto triukšmo [26]. Egzistuoja keli eksperimentai, kuriuose bandyta pritaikyti principinių komponenčių analizę (angl. *principal component analysis* arba PCA) [27, 28]. Šiuo metodu siekiama sutraukti esminius garso požymius į principines komponentes. Po to mažesnis kiekis principinių komponenčių naudojamas garso atkūrimui, tikintis, kad pats mažinamas triukšmas egzistuoja žemesnėse, mažiau informacijos turinčiose komponentėse, tokiu būdu sumažinant triukšmą iš garso failo.

Kitas populiarus sprendimas, kuris pasiekia gerus rezultatus su priešiško triukšmo mažinimu, yra Rudino-Ošerio-Fatemi metodas (angl. *Rudin-Osher-Fatemi* arba *ROF)* [29, 30]. Šis algoritmas siekia

pašalinti visus nenatūralumus iš garso, siekiant kuo labiau išlaikyti originalią garso struktūrą. Metodas atlieka energijos minimizavimą funkcijos, kuriame lyginamas atitikimas triukšmingam garsui. Staigūs svyravimai yra baudžiami taikant bendrosios variacijos (dispersijos) reguliarizaciją. Dėl to šis metodas literatūroje dažnai vadinamas triukšmo mažinimu pagal bendrąją variaciją (angl. *Total Variation Denoising*).

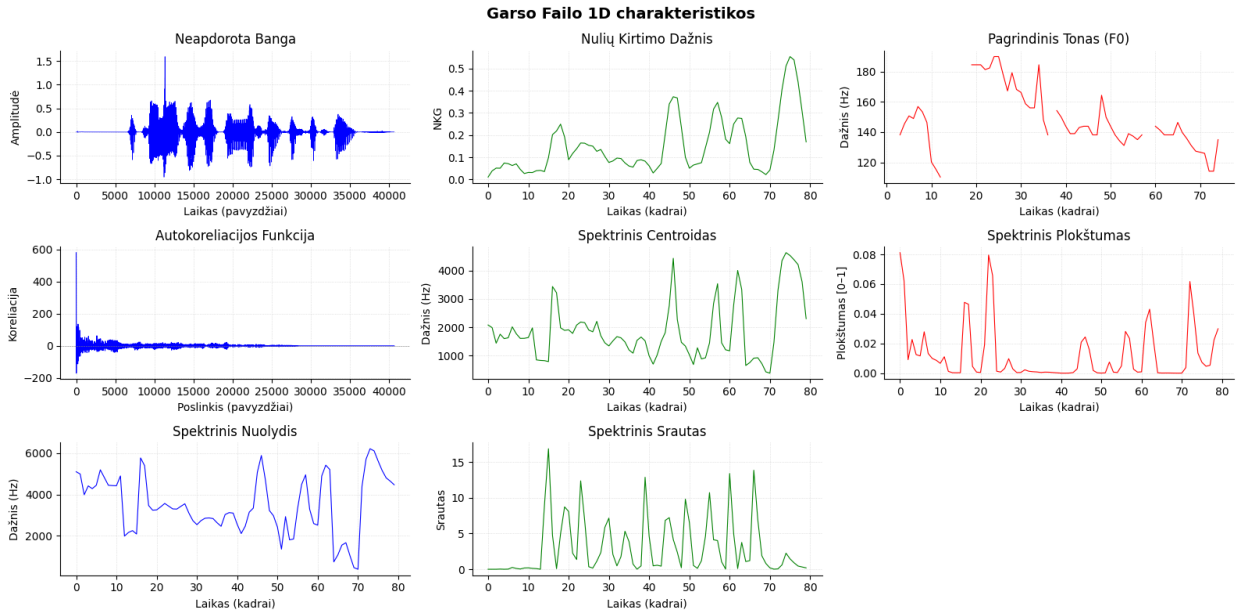
Galiausiai, dažnai taikomi neuroninius tinklus naudojantys sprendimai. Triukšmo mažinimui dažnai naudojami autoenkoderiai, kurie suspaudžia ir paskui vėl išskleidžia garsą, tikintis, kad bus atkurtas netriukšmingas garsas [31]. Šis metodas sugeba pasiekti gerus rezultatus, tačiau jam reikia daug tikro ir triukšmingo garso pavyzdžių porų. Panašiai veikia U-Net [32, 33] tipo modeliai, kurie taipogi gali būti taikomi triukšmo mažinimui, tačiau jie susiduria su tuo pačiu duomenų kiekio iššūkiu. Galiausiai, egzistuoja kiti neuroniniais tinklais paremti sprendimai [34, 35, 36].

1.4. Garso požymiai

Prieš pradedant klasifikuoti garso įrašus, juos pirma reikia apdoroti. Šio proceso metu garsas yra transformuojamas į kitokią, daugiau naudingos informacijos turinčią, formą. Šiame skyriuje bus aptarti toliau tyrime naudojami garso požymiai ir transformacijos, su kuriais vėliau bus mokomi įvairūs klasifikavimo sprendimai.

1.4.1. Vienmačiai požymiai

Paprasčiausios garso savybės yra vienmatės, kaip laiko eilutės. Šie požymiai kalbos signalą atvaizduoja kaip vieną reikšmių seką laiko atžvilgiu [37]. Skaičiavimų požiūriu šios savybės yra efektyvios ir dažnai naudojamos pagrindinėms laikinėms arba statistinėms garso signalų charakteristikoms išgauti. Skirtingai nuo aukštesnių dimensijų reprezentacijų, vienos dimensijos požymiai dažniausiai apibūdina specifines bangos formos arba dažnių pasiskirstymo požymius, naudojant vieną reikšmę arba vektorių kiekvienam kadrui (iškarpai). Dėl savo paprastumo ir interpretuojamumo šie požymiai yra plačiai naudojami kalbos apdorojimo ir garso klasifikavimo uždaviniuose. Žemiau pateiktos kelios šiame tyrime išbandytos garso požymių reprezentacijos (žr. **3 pav.**).



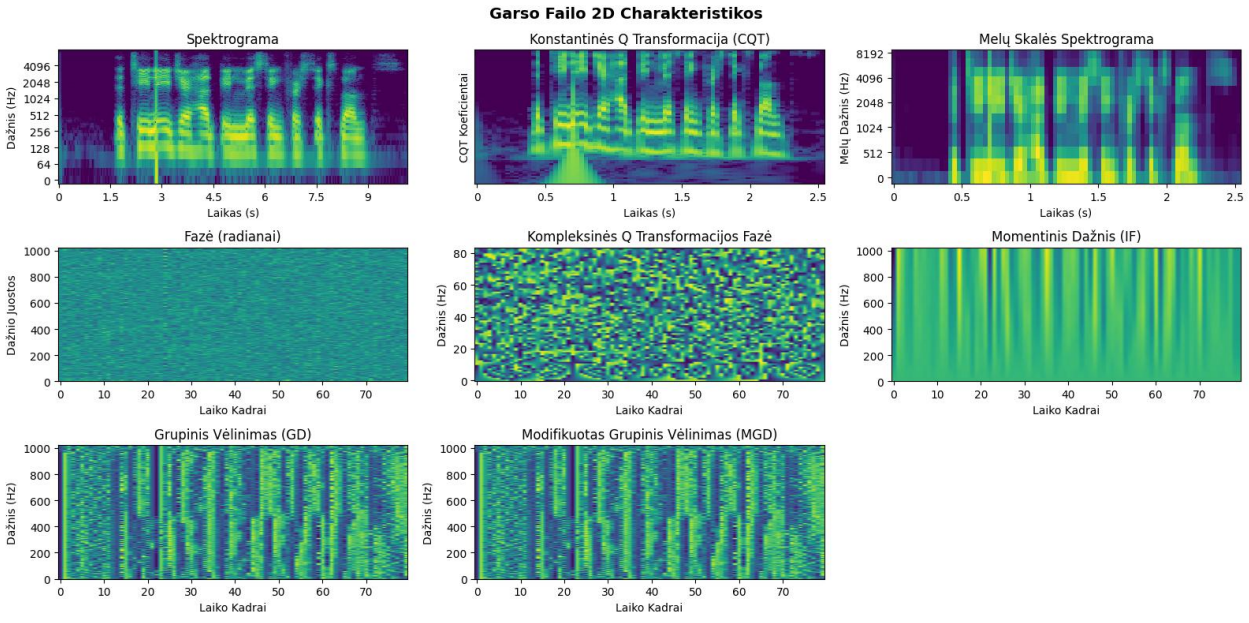
3 pav. Tyrime tirtų vienmačių savybių vizualizacijos

Vienas dažniausiai naudojamų vienmačių požymių yra žaliavinė neapdorota garso banga (angl. *raw waveform*) – tai yra pats pradinis garso signalas laiko srityje, atspindintis amplitudės pokyčius laikui bėgant. Skirtingai nei kitos šiame skyriuje aptariamos savybės, neapdorota garso banga nėra išgaunama transformacijomis, o naudojama tiesiogiai kaip modelio įvestis – būtent taip ji panaudojama *RawNet* tipo architektūrose [38]. Kita laiko srities savybė – nulio kirtimo dažnis (angl. *zero-crossing rate* arba *ZCR*), kuris parodo, kaip dažnai signalas kerta nulinę amplitudės reikšmę, ir leidžia įvertinti signalo triukšmingumą bei balsingumą. Autokoreliacija matuoja signalo panašumą į laiko atžvilgiu paslinktą jo kopiją, taip atskleidžiant signalo periodiškumą, tai yra esmė pagrindiniam tonui (angl. *pitch* arba *F0*) apskaičiuoti. Pagrindinis tonas apibūdina žemiausią periodinį garso signalo dažnį ir yra tiesiogiai susijęs su suvokiamu balso aukštumu, todėl jis ypač svarbus analizuojant natūralios ir sintetinės kalbos skirtumus [39].

Be laiko srities savybių, kalbos analizėje plačiai naudojami ir spektriniai deskriptoriai, kurie apibūdina signalo energijos struktūrą dažnių srityje [40]. Pirmas toks deskriptorius yra spektrinis centroidas (angl. *spectral centroid*), kuris nurodo, kurioje dažnių srityje sutelkta didžioji dalis signalo energijos. Spektrinis plokštumas (angl. *spectral flatness*) matuoja, kiek spektro energija yra tolygiai pasiskirsčiusi per visus dažnius. Kuo reikšmė artimesnė vienetui, tuo signalas labiau primena baltą triukšmą, o žema reikšmė rodo ryškią toninę struktūrą. Spektrinis nuolydis (angl. *spectral roll-off*) apibrėžia dažnį, žemiau kurio sukaupta didžioji dalis (dažniausiai 85%) visos signalo energijos, taip apibūdinant spektro pasiskirstymo asimetriją [41]. Galiausiai, spektrinis srautas (angl. *spectral flux*) matuoja spektro pokyčius ir yra ypač jautrus staigiai besikeičiantiems signalo momentams. Tai šią savybę daro ypač naudingą aptinkant vokoderio sintezės artefaktus, kurie pasireiškia netipiniais spektriniais šuoliais.

1.4.2. Spektrinės ir fazinės transformacijos

Didesniam informacijos kiekiui išgauti garso požymiai dažnai yra paverčiami į dvimates formas. Tokios reprezentacijos leidžia vienu metu analizuoti tiek laiko, tiek dažnio srities informaciją, todėl yra ypač naudingos modeliuojant sudėtingus akustinius reiškinius. Žemiau pateiktos kelios šiame darbe tirtos dvimačių savybių reprezentacijos (žr. **4 pav.**).



4 pav. Tyrime tirtų spektrinių ir fazinių savybių vizualizacijos

Viena paprasčiausių ir plačiausiai naudojamų garso transformacijų yra spektrograma. Ji atvaizduoja garso signalo energijos pasiskirstymą laiko ir dažnio srityse, kur ryškesnės spalvos atitinka aukštesnę signalo amplitudę tam tikrame dažnio ir laiko vertėje. Spektrograma yra gaunama taikant Furjė transformaciją [42], kurios metu garso signalas yra padalijamas į persidengiančius trumpus langus, o kiekviename iš jų apskaičiuojamas dažnių spektras. Tokiu būdu gaunama laiko ir dažnio priklausomybė, leidžianti analizuoti, kaip signalo spektrinės savybės kinta laike [43]. Žemiau pateikta standartinės Furjė transformacijos formulė (žr. 2 formulė).

$$S_x(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi f t} dt \tag{2}$$

čia $x(t)$ yra garso bangos reikšmė laiko momentu t, $2\pi f$ yra signalo dažnis radianais per sekundę. $S_x(f)$ yra Furjė transformacijos išvestis dažnių srityje. Praktikoje naudojama šios transformacijos modifikacija vadinama greitąją Furjė transformacija (angl. *Fast Fourier Transform* arba *FFT,* kartais *Short Time Fourier Transform* arba *STFT*), kuri per pusę sumažina šiai transformacijai reikalingų skaičiavimų skaičių.

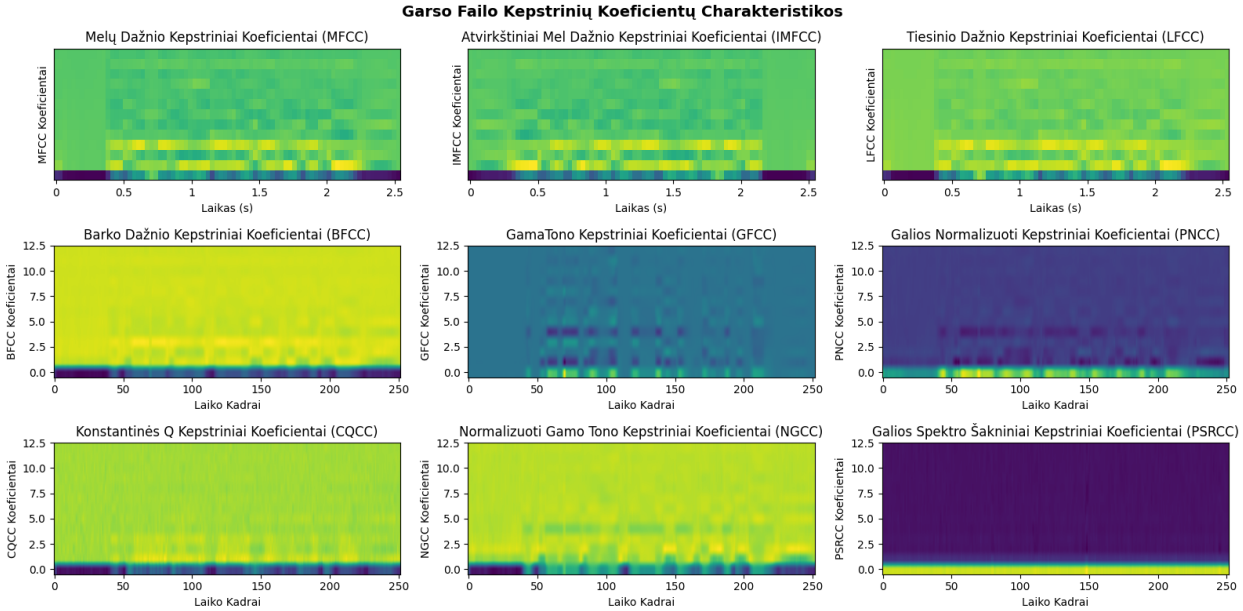
Remiantis spektrogramos principu, dažnai naudojami ir jos variantai, tokie kaip melų skalės spektrograma. Melų skalės spektrograma yra spektrogramos transformacija, kurioje dažnių ašis perskaičiuojama į melų skalę, kuri labiau atitinka žmogaus klausos suvokimą. Žmogaus klausa yra jautresnė žemiems dažniams ir mažiau jautri aukštiems, todėl melų skalė suspaudžia aukštųjų dažnių sritį ir išplečia žemųjų dažnių detalumą [44]. Galutinis rezultatas geriau atspindi tas garso dalis, kurios iš tikro yra girdimos žmogaus ausiai. Kita garso analizėje populiari spektrogramos versija yra Q konstantų transformacija (angl. *constant Q transform* arba *CQT*) gauta spektrograma. CQT spektrogramos dažnių skalė yra logaritminė, o kiekvieno dažnių intervalo skiriamoji geba yra proporcinga jo centriniam dažniui. Tai reiškia, kad žemų dažnių srityje pasiekiama didesnė dažninė raiška, o aukštų dažnių srityje – didesnė laikinė raiška. Dėl šios savybės CQT ypač gerai atspindi muzikinius ir kalbos signalus [45].

Tęsiant spektrinių reprezentacijų analizę, svarbu paminėti ir fazės informaciją, kuri dažnai yra ignoruojama tradiciniuose garso požymiuose, tačiau gali suteikti reikšmingos papildomos informacijos. Fazė nusako signalo bangos padėtį jos periodo iškarpoje tam tikru laiko momentu ir apibūdina, kaip skirtingi dažnio komponentai atkuria garsą laike. Skirtingai nuo amplitudės spektro, kuris aprašo energijos pasiskirstymą, fazė išlaiko struktūrinę informaciją apie signalo generavimo procesą. Klastotos kalbos aptikimo uždaviniuose fazinė informacija yra ypač svarbi, nes sintetinės kalbos ir balso konversijos metodai dažnai sukuria subtilius fazės netolygumus arba nenatūralią fazinių komponentų sąveiką. Šie artefaktai gali būti sunkiai pastebimi vien tik amplitudės srityje, tačiau fazės analizė leidžia juos efektyviau identifikuoti. Dėl šios priežasties fazės pagrindu sukurti požymiai, tokie kaip grupinis vėlinimas (angl. *group delay* arba *GD*) ar momentinis dažnis (angl. *instantaneous frequency* arba *IF*), dažnai naudojami siekiant padidinti klastoto balso aptikimo sprendimų atsparumą apgaulei [46, 47].

1.4.3. Kepstriniai požymiai

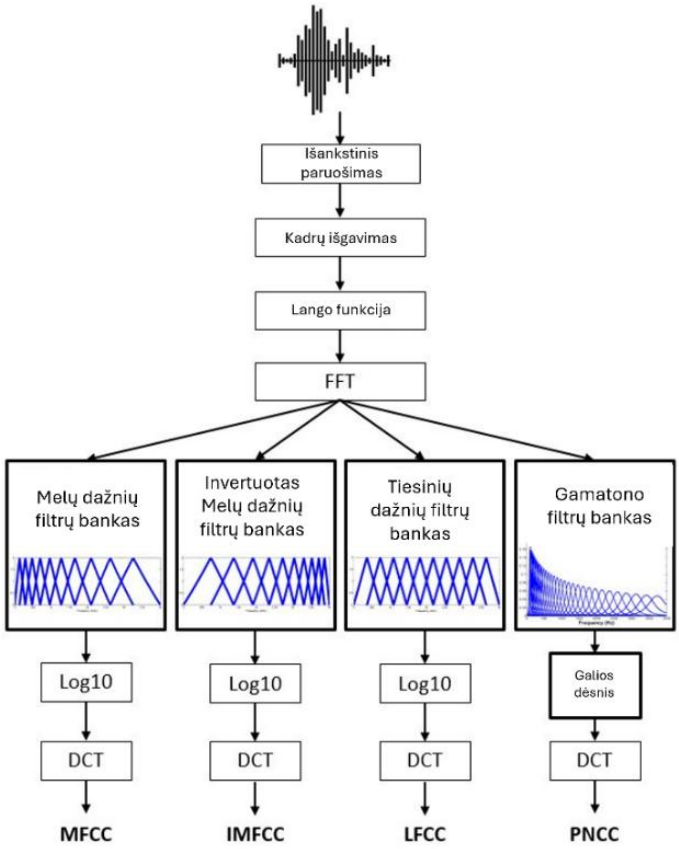
Dvimatėms garso reprezentacijoms taipogi galima atlikti transformacijas ir išgauti dar daugiau papildomos garso informacijos. Tokie požymiai vadinami kepstriniais (žr. **5 pav.**). Kepstrinai požymiai yra gaunami taikant logaritminę transformaciją galios spektre ir vėliau atliekant atvirkštinę Furjė transformaciją. Taip gaunama nauja reprezentacija, kuri aprašo spektrinio apvalkalo (angl.

envelope) struktūrą. Kitaip tariant, kepstrinės savybės leidžia atskirti lėtai kintančias signalo charakteristikas, susijusias su balsų takų forma spektrogramoje, nuo greitai kintančių komponentų, susijusių su šaltiniu ar triukšmu. Dėl šios priežasties jos plačiai naudojamos kalbos atpažinimo, kalbėtojo identifikavimo bei klastotos kalbos aptikimo uždaviniuose.



5 pav. Tyrime tirtų kepstrinių savybių vizualizacijos

Nors aukščiau pavaizduotos savybės (žr. **5 pav.**) skiriasi savo skaičiavimo detalėmis, jos visos remiasi panašia bendra schema [48], kuri pateikta žemiau (žr. **6 pav.**).



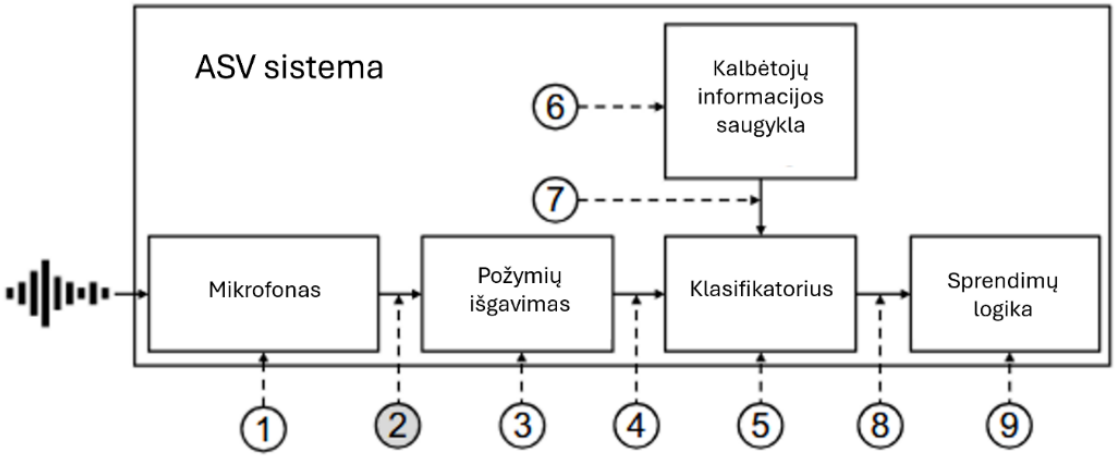
6 pav. Skirtingų kepstrinių savybių gavimas [48]

Pirmiausia neapdorotas garso signalas praeina per išankstinio pabrėžimo (angl. *pre-emphasis*) filtrą, kuris sustiprina aukštesnių dažnių komponentes ir taip pagerina tolesnio apdorojimo kokybę. Vėliau signalas yra padalinamas į trumpus persidengiančius segmentus (kadrus), kurie vėliau yra padauginami iš lango funkcijos (pvz., *Hanning* arba *Hamming*), siekiant sumažinti spektrinius nuotėkius. Kiekvienam kadrui taikoma Furjė transformacija, kuria laikinių duomenų signalas paverčiamas galios spektru ir taip gaunamas pagrindas tolesniam filtravimui. Čia skirtingi kepstriniai požymiai išsiskiria: galios spektras perduodamas į skirtingus filtrų bankus (angl. *filter banks*), kurie kiekvienas modeliuoja žmogaus klausos sistemą skirtingu būdu. Po filtravimo kiekvienas filtrų banko išėjimas transformuojamas logaritmu. Tam tikros kepstrinės savybės gaunamos šiame žingsnyje atliekant kitokią transformaciją, pavyzdžiui, galios normalizuoti kepstriniai koeficientai (angl. *Power Norm Cepstral Coefficients* arba *PNCC*), gaunami filtro banko išėjimą transformuojant pagal galios dėsnį. Galiausiai, visais būdais gauti rezultatai perduodami diskretinei kosinusų transformacijai (angl. *discrete cosine transform* arba *DCT*), kuri dekoreliuoja filtrų bankų išėjimus ir suglaudina informaciją į kompaktišką koeficientų vektorių – tai ir yra galutiniai kepstriniai požymiai.

Klastotos kalbos aptikimo užduotyje dažniausiai naudojami keli kepstriniai požymiai. Pirmasis iš jų yra tiesiniai kepstriniai dažnių koeficientai (angl. *linear frequency cepstral coefficients* arba *LFCC*) [49]. Šie koeficientai pasižymi tuo, kad jų dažnių skalė yra tolygiai pasiskirsčiusi skirtingai, nei melų skalė, ir nesuspaudžia aukštųjų dažnių srities. Tai ypač svarbu klastojimo aptikimo kontekste, kadangi daugelis sintetinės kalbos artefaktų pasireiškia būtent aukštųjų dažnių srityje, kurią *MFCC* tipo savybės linkusios nuvertinti. Panašiu principu grindžiamas ir Melų dažnių atvirkštinių kepstrinių koeficientų (angl. *inverse mel frequency cepstral coefficients* arba *IMFCC*) naudojimas – apversta melų skalė sustiprina aukštųjų dažnių sritį, todėl šios savybės taip pat išlaiko jautrumą sintetinės kalbos artefaktams [50]. Galiausiai, paskutinis kalbos klastojimo aptikimo uždavinyje plačiai naudojamas kepstrinis požymis yra Q konstantų transformacijos kepstriniai koeficientai (angl. *constant Q transform cepstral coefficients* arba *CQCC*). Šis požymis remiasi ne Furjė transformacija, o praeitame poskyryje minėtos *CQT* transformacijos pagrindu. *CQT* gebėjimas išlaikyti pastovų dažnio ir raiškos santykį leidžia *CQCC* efektyviau aprašyti kalbos natūralią struktūrą, o sintetinės kalbos sistemos kaip tik sunkiau sugeba šią struktūrą tiksliai atkartoti, todėl klastojimo artefaktai tampa geriau pastebimi. *CQCC* buvo pasiūlyta būtent klastojimo aptikimo uždaviniui spręsti [51], ir ši savybė iki šiol pasiekia labai gerus rezultatus *ASVspoof* iššūkiuose (angl. *benchmark*) [1, 5].

1.5. Garso klasifikavimo įrankiai

Kaip skyriaus įvade jau buvo minėta, praktinis klastočių aptikimo sistemų panaudojimas yra automatinėse kalbėtojo tapatybės patvirtinimo sistemose [3, 5, 52] (angl. *automatic speaker verification* arba *ASV*) (žr. **7 pav.**). Visos šios sistemos veikia panašiai. Pirmiausia tam tikru būdu sistemai paduodamas garsas, kuriame kalba koks nors asmuo. Paduodamas garsas yra apdorojamas ir iš jo ištraukiami tam tikri požymiai. Tada jie yra pateikiami klasifikatoriui, kuris apskaičiuoja tikimybę, kad garso įraše girdimas numatytas asmuo. Kadangi klasifikatorius turėtų galėti atpažinti daug įvairių asmenų, šios sistemos dažnai turi numatytųjų žmonių balso duomenų saugyklą, iš kurios klasifikatoriui pateikiama specifinių žmonių balsinė informacija, kurią klasifikatorius naudoja tikslesniam rezultatui apskaičiuoti.



7 pav. Automatinės kalbėtojo patikrinimo sistemos schema [3]

Šiam tyrimui svarbiausia dalis yra pats garso failo klasifikavimo procesas, tikrinantis, ar failas buvo sugeneruotas sintetiniu būdu. Taigi, dvi svarbiausios dalys yra garso paruošimas ir pats klasifikavimo algoritmas. Prieš tai minėtame *ASVSpoof* iššūkyje, didžioji dalis pasiūlytų sprendimų susideda iš būtent šių dviejų dalių. Dalis sprendimų bando išgauti daug įvairių savybių ir tada, jas apjungus, pateikti klasifikatoriui. Tokie sprendimai dažnai išveda gerus rezultatus, tačiau jie reikalauja daug daugiau skaičiavimo resursų ir dėl to yra daug lėtesni už kitus sprendimus, kurie fokusuojasi tik į vieną savybę. Be to, nemažai egzistuojančių sprendimų naudoja daug modelių ir paskui balsavimo būdu nusprendžia ar failas yra suklastotas, ar ne. Šio tipo sprendimai taipogi pasiekia geresnius rezultatus už pavienius klasifikatorius [5, 53]. Tačiau, kadangi naudojami keli modeliai, tokie sprendimai naudoja daugiau resursų ir veikia lėčiau. Toliau šiame skyriuje bus aprašyti kelios klastotos kalbos aptikimui dažniau naudojamos klasifikavimo technologijos.

1.5.1. Konvoliuciniai neuroniniai tinklai

Konvoliuciniai neuroniniai tinklai [54] yra mašininio mokymo algoritmas, kuris dažniausiai yra taikomas vaizdų apdorojimo srityje, tačiau juos galima panaudoti ir garsų apdorojimui. Neapdorotam garsui galima naudoti vienos dimensijos konvoliucijos operaciją [55], kadangi garso banga (angl. *waveform*) standartiškai programiškai atvaizduojama skaičių masyvu. Papildomai, garsą galima transformuoti į dvimatę formą (pvz. spektrogramą arba *MFCC)* ir jai taikyti dviejų dimensijų konvoliucijos operaciją.

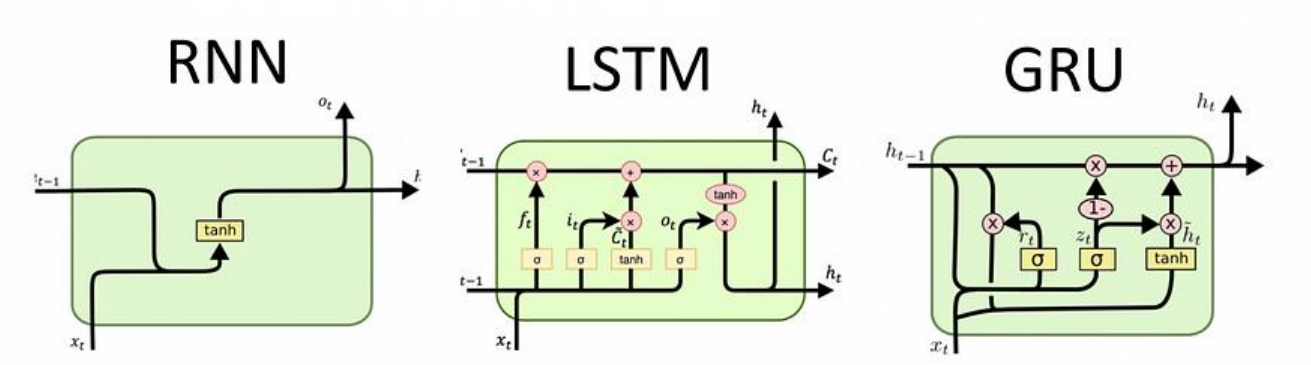
Konvoliuciniai neuroniniai tinklai naudoja jau prieš tai minėtą konvoliucijos operaciją [54]. Ši operacija sumažina duomenyse esančios informacijos kiekį, išlaikant svarbiausią informaciją. Tai daroma pasitelkiant filtrus, kurie yra slenkami paveikslėliu. Filtrai dažniausiai yra kvadratinės matricos ir gali būti bet kokio dydžio. Kiekvienas filtro celė turi svorį, kuris yra sudauginamas su atitinkamu tos filtro celės uždengtu paveikslėlio pikseliu. Po to apskaičiuojamas visų celių narių ir juos atitinkančių pikselių sandaugų vidurkis kuris naudojamas, kaip nauja pikselio vertė sumažintame paveikslėlyje. Kai su filtru praslenkama pro visą paveikslėlį, gaunamas sumažintas paveikslėlis, kuriame išsaugoma visa svarbiausia informacija. Tokios operacijos yra ypač svarbios bandant klasifikuoti neapdorotą garsą, kadangi, kaip jau prieš tai minėta, jo skaitmeninė reprezentacija yra skaičių masyvas, kuris standartiškai būna labai didelis. Net vienai sekundei reprezentuoti gali būti naudojami dešimtys tūkstančių skaitmenų. Garso duomenų rinkiniai standartiškai naudoja 16 kHz diskretizacijos dažnį (angl. *sample rate*), kas parodo, kad garso vertės per sekundę pasikeičia šešiolika

tūkstančių kartų, ir kiekvienas pasikeitimas yra pažymimas garso faile. Be konvoliucijos operacijos reiktų naudoti milžiniškus klasifikatorius, kurie naudotų milžiniškus kiekius skaičiavimo resursų, norint prognozuoti vos kelių sekundžių ilgio įrašo klasę.

Klastoto balso aptikimo uždaviniuose ypač dažnai naudojamos įvairios *ResNet* (angl. *Residual Network*) tipo modelių architektūros [3, 55, 56]. Šio tipo modeliai naudoja šuolių jungtis (angl. *skip connection*), kurios tam tikrų sluoksnių išvestis papildomai sujungia su tinkle giliau esančiais sluoksniais, taip praleisdamas tam tikrus sluoksnius. Ši savybė išsprendžia vieną svarbiausių neuroninių tinklų iššūkių – nykstamo dydžio gradiento problemą. Kadangi, taikant šias jungtis, gradientas mažėja daug lėčiau, galima apmokyti daug gilesnius neuroninius tinklus, kurie idealiai išmoks iš garso išgauti daugiau informacijos ir taip padaryti tikslesnę prognozę. Šio modelio tipo modifikacija *Res2Net* [57] pačiame naujausiame jau minėtame *ASVspoff* [5] iššūkyje pasirodė geriausiai. Šioje modifikacijoje tinklu keliaujantys signalai (duomenys) yra padalijami į kelias grupes, ir tada kiekvienai grupei skaičiavimai yra atliekami atskirai, iki kol galutinės kiekvienos grupės išvestys vėliau vėl yra sujungiamos.

1.5.2. Rekurentiniai neuroniniai tinklai

Rekurentiniai neuroniniai tinklai [58, 59] yra mašininio mokymo algoritmas, dažniausiai taikomas laiko eilučių analizėje, dėl to jie puikiai tinka garso analizei. Tinklai naudoja rekurentines celes (žr. **8 pav.** Skirtingų rekurentinių tinklų celių pavyzdžiai [60]), kurios specialiai pritaikytos laike kintančių duomenų apdorojimui. Kiekviena celė turi dvi įvestis: duomenims tame laiko momente ir praeito laiko momento celės išvestą rezultatą. Celė apdoroja šias įvestis ir išveda rezultatą dabartiniam laiko momentui. Šis įvertis pateikiamas sekančiai celei, kuri toliau skaičiuoja rezultatą sekančiam laiko momentui. Pilnai apdorojus visą garso įrašo seką, šių celių rezultatai pateikiami pilnai sujungtam sluoksniui, kuris atlieka garso klasifikavimą.



8 pav. Skirtingų rekurentinių tinklų celių pavyzdžiai [60]

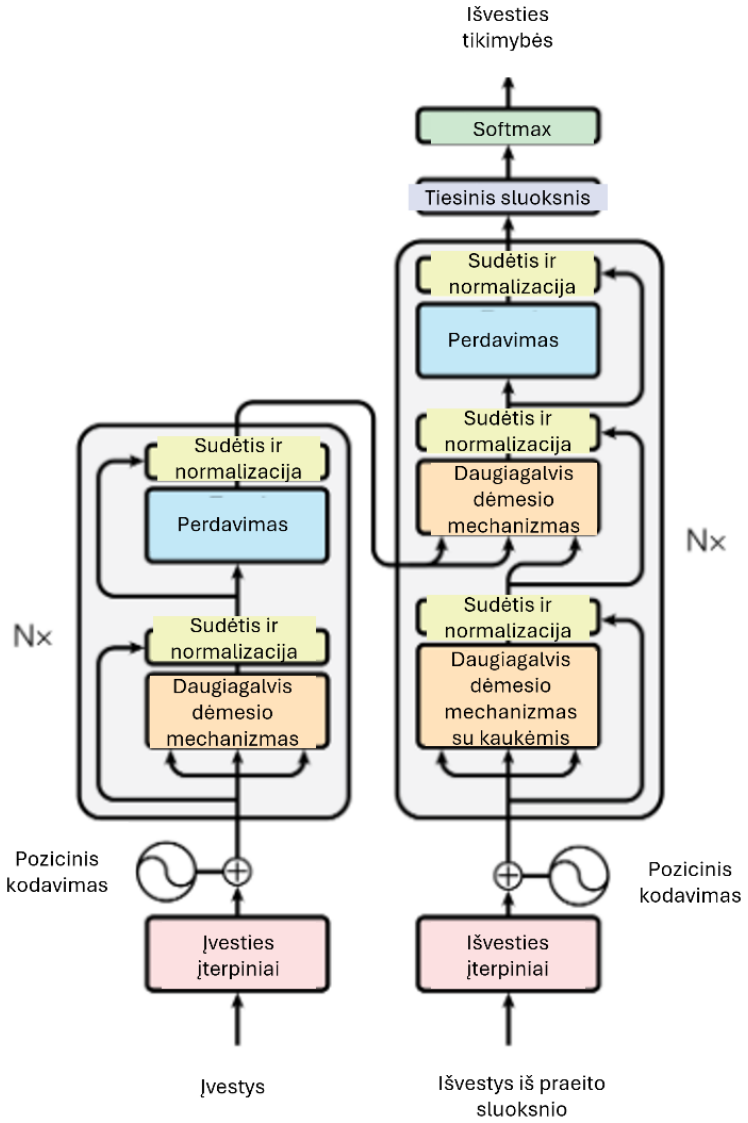
Šiuos tinklus ypač smarkiai paveikia nykstančio dydžio gradiento problema, todėl jie negali apdoroti labai ilgų garso sekų. Šiai problemai spręsti buvo pasiūlytos įvairios modifikacijos [58, 59, 60]. Kelios iš jų yra ilgalaikės – trumpalaikės atminties tinklai (angl. *long-short term memory* arba *LSTM*) ir uždari rekurentiniai vienetai (angl. *gated recurrent unit* arba *GRU*). Šios modifikacijos įveda technikas, skirtas pašalinti neaktualią informaciją, kas leidžia ilgiau išlaikyti duomenų sekoje rastą svarbią informaciją ir taip prailgina laiko sekų ilgį, kurį galima sėkmingai apdoroti.

Rekurentiniai neuroniniai tinklai, klastoto balso aptikimo srityje, senesniuose *ASVspoof* iššūkiuose šie tinklai sugebėdavo pasiekti pakankamai gerus rezultatus [61, 62, 63], tačiau dabar jie naudojami

daug rečiau, iš esmės dėl to, nes šias užduotis daug geriau atlieka rekurentinių neuroninių evoliuciniai deriniai – transformeriai.

1.5.3. Transformeriai

Kitas garso analizei puikiai tinkantis algoritmas yra transformeriai [64]. Šio tipo modeliai naudoja dėmesio mechanizmą, kuris leidžia vienu metu analizuoti visos duomenų sekos elementus vienu metu. Tai leidžia transformeriams veiksmingiau aptikti tiek trumpalaikes, tiek ilgalaikes duomenų priklausomybes sekoje ir tuo pačiu žymiai pagreitina duomenų apdorojimo laiką, kadangi skaičiavimai gali būti išlygiagretinti. Dėl šių priežasčių transformeriai veikia daug efektyviau nei prieš tai minėti rekurentiniai neuroniniai tinklai, kurie nuoseklius duomenis visada turi apdoroti žingsnis po žingsnio. Žemiau pateikta standartinė šio modelio architektūra (žr. **9 pav.**).



9 pav. Transformerio architektūra [64]

Transformerių architektūra susideda iš vienas ant kito išdėstytų kodavimo ir dekodavimo sluoksnių, kurie savyje turi dėmesio mechanizmus ir pilnai sujungtus sluoksnius. Kiekvienas kodavimo

sluoksnis turi po vieną dėmesio mechanizmą, dekodavimo sluoksnis – po du. Šių dėmesio mechanizmų dėka transformeriai išmoksta atpažinti kontekstinius ryšius tarp sekos elementų.

Nuo pat atsiradimo transformeriai tapo vyraujančia architektūra daugelyje mašininio mokymosi sričių. Jie gali būti naudojami natūralios kalbos apdorojime, teksto generavime, kompiuterinėje regoje ir t.t. Kalbos apdorojimo uždaviniuose transformeriai pasiekia labai gerus rezultatus kalbos, kalbėtojo ir klastoto balso atpažinimo srityse. Šiuolaikinėse klastotos kalbos atpažinimo sistemose vis dažniau įterpiamos transformeriais arba tiesiog dėmesiu paremtos architektūros. Jos dažnai sujungiamos su savarankiško mokymosi metodais, kuriais siekiama padidinti atsparumą sintetinių ir konvertuotų balsų atakoms. Vienas tokio sprendimo pavyzdys yra *WavLM* [65]. Šis modelis sujungia konvoliucinį požymių kodavimą su transformerio konteksto tinklu. Tas leidžia modeliui gerai išskirti įvairias garso savybes ir sujungti jas su ilgalaike duomenų laiko priklausomybių informacija. Modelis išmoksta reprezentuoti kalbą, mokymo metu naudojant didelį kiekį nesužymėtų garso įrašų, atliekant užmaskuotos garso dalies prognozės uždavinius. Galiausiai, modelis išmoksta apskaičiuoti tokias garso reprezentacijas, kurios kitiems dirbtinio intelekto modeliams suteikia daugiau konteksto, nei standartinės garso savybės. Šis algoritmas yra labai plačiai taikomas klastotos kalbos atpažinimo užduotyje, kur jis taipogi pasiekia labai gerus rezultatus.

1.5.4. Gausiniai mišinių modeliai

Gausinių mišinių modelis [66] (angl. *Gaussian mixture model* arba *GMM*) yra neprižiūrimojo mokymosi algoritmas, kuris dažniausiai yra naudojamas klasterizavimo uždaviniams spręsti, kur kiekvienam duomenų imties elementui skaičiuojamos tikimybės, kad tas elementas priklauso kokiam nors klasteriui. Gausinis mišinys yra funkcija, sudaryta iš kelių Gausinių skirstinių (angl. *Gaussians*), kurie modelyje veikia kaip klasteriai, kurie turi savo vidurkius, kovariaciją ir svorius. Panašiai kaip *K-Means* [67] algoritme reikia parinkti, kiek pradinių klasterių turime ir kokie yra jų pradiniai parametrai. Modelio apmokymui naudojamas lūkesčių maksimizavimo (angl. *Expectation-Maximization*) algoritmas, kuris iteratyviai priskiria duomenis klasteriams ir atnaujina modelio parametrus. Apmokius modelį, jam galima paduoti naujus duomenis ir modelis apskaičiuoja priklausymą, nurodančias duomenų priklausomumą vienam ar kitam klasteriui.

Šis algoritmas dažnai naudojamas kalbėtojų atpažinimo (angl. *speaker verification*) užduotyje. Šiai užduočiai standartiškai naudojami keli *GMM* modeliai (kiekvienam kalbėtojui naudojamas atskiras modelis), kuriems paduodami apdoroti garso požymiai, kaip *MFCC*. *GMM* modeliai išveda tikimybes, kad garse girdimo kalbėtojo balso požymiai priklauso to modelio Gausiniams komponentams, ir pagal tas tikimybes nusprendžiama, kuris kalbėtojas kalba. Klastotos kalbos aptikimo uždavinyje pakanka apmokyti du modelius – vieną klastotai ir kitą tikrai kalbai apdoroti. Istoriškai *GMM* modeliai buvo vienas iš geriausių pasirinkimų, norint aptikti klastotą balsą [3, 68, 69]. Senesnės kalbos sintezės technologijos buvo daug primityvesnės ir dėl to, jomis generuoto klastoto balso įrašuose atsirasdavo aiškūs artefaktai, kuriuos *GMM* modeliai galėjo puikiai išskirti klasterizuojant *CQCC* garso požymius [3, 69]. Dabar *GMM* modeliai klastoto balso atskyrimo uždaviniui nebėra taikomi, kadangi, smarkiai patobulėjus garso sintezės algoritmams, šių modelių vidutinis tikslumas dabar yra per žemas. Papildomai, įvairūs kiti mašininio mokymo modeliai išgauna daug geresnius rezultatus šioje užduotyje [5, 70, 71].

1.5.5. RawNet

Šios modelių šeimos paskirtis yra garso failuose girdimų balso požymių išgavimas [38, 72]. Modelis naudoja 1D konvoliucinius neuroninius tinklus, su kuriais modelis iš 3 sekundžių ilgio garso įrašų išgauna garso įraše girdimo balso požymius ir juos išveda vektoriaus reprezentacija. Modelis originaliai buvo mokytas kalbėtojų atskyrimo užduočiai su *VoxCeleb* duomenų rinkiniu, tačiau jis puikiai tinka ir klastotos kalbos atskyrimui nuo tikros [73]. Kadangi pats *RawNet3* modelis nevykdo klasifikavimo, reikia apmokyti atskirą klasifikatorių, kuris šio modelio išvedamus vektorius atskirtų į atskiras klases. Dažniausiai naudojamas vienas pilnai sujungtas sluoksnis, kuris turi 256 įvestis (*RawNet3* vektoriaus ilgis) ir 2 išvestis tikroms ir klastotoms garso klasėms [72]. Klasifikatoriams mokyti rekomenduojama naudoti Softmax su adityvia kampine riba (angl. *Additive Angular Margin Softmax* arba *AAM-softmax)* aktyvacijos funkciją [74] arba *DINO* mokymo karkasą [72]. Iš šių dviejų būdų geresnius rezultatus pasiekia *AAM-Softmax* funkcija [74].

AAM-Softmax yra paprastos *Softmax* funkcijos modifikacija, kuri mokymo metu prideda papildomus skaičiavimus, kurie padeda modeliui išmokti geriau atskirti skirtingas klases. Žemiau pateikta *AAM-Softmax* aktyvacijos formulė (žr. 3 formulė).

$$L_A = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\left(cos\left(\theta_{y_i},i\right)+m\right)}}{e^{s\left(cos\left(\theta_{y_i},i\right)+m\right)} + \sum_{j=1,j\neq y_i}^{N} e^{s\,cos\left(\theta_{y_i},i\right)}} \quad (3)$$

čia $\cos(\theta_j, i)$ yra pilnai sujungto sluoksnio išvesties ir klasės kosinuso panašumas, m yra pridedama riba, s yra mastelio koeficientas. Apmokymo žingsnio metu modeliui pasunkinamas rezultato apskaičiavimas siekiant, kad modelis išmoks geriau atskirti skirtingoms klasėms priklausančius duomenis. Validavimo žingsnyje ir naudojant patį modelį ši formulė netaikoma, o tiesiog naudojamas apmokytas pilnai sujungtas sluoksnis su paprastu *Softmax* klasifikatoriumi. Tačiau kadangi šis sluoksnis buvo apmokytas taikant *AAM-Softmax*, jis sukuria didesnius skirtumus tarp atskirų klasių ir taip sugeba geriau atskirti skirtingas klases. Šis klastoto balso klasifikavimo modelis teoriškai sugeba susidoroti su klastotos kalbos aptikimo užduotimi, *ASVspoof5* iššūkyje senesnė *RawNet2* architektūra buvo naudota kaip vienas iš bazinių modelių kitiems sprendimams įvertinti.

1.5.6. AASIST

AASIST (angl. *Audio Anti-Spoofing using Integrated Spectro-Temporal Graph Attention Networks*) yra specifiškai klastotos kalbos atpažinimo užduočiai sukurtas sprendimas [75]. Algoritmas kalbą modeliuoja kaip grafą, kuriame viršūnės atitinka koduotos spektrogramos vietinius laiko-dažnio segmentus, o briaunos atspindi ryšius laiko ir dažnio dimensijose. Algoritme naudojamas pamodifikuotas *RawNet2* modelis, kuris užkoduotus garso požymius paverčia į dvimatę formą, panašią į spektrogramą. Šis grafais pagrįstas sprendimas leidžia modeliui vienu metu atkurti ir smulkius akustinius netolygumus, ir bendrus struktūrinius modelius, būdingus suklastotam arba sintezuotam balsui. Po to modelis pritaiko grafų dėmesio mechanizmą, kuris išmoksta atskirti tikro ir klastoto užkoduoto balso požymius. Tai leidžia modeliui atrasti regionus, kuriuose egzistuoja garso generavimo metodų palikti skiriamieji požymiai kaip vokoderių sukurti artefaktai arba fazės neatitikimai. Dėl šios priežasties šis modelis veiksmingai aptinka šiuolaikinių TTS ir VC modelių generuotus garso įrašus.

Šis algoritmas yra plačiai naudojamas generuotos kalbos aptikimo užduotyse. *ASVspoof5* [5] iššūkyje šis modelis panašiai kaip ir *RawNet2* buvo taikomas kaip bazinis modelis. Nemaža dalis tame iššūkyje pasirodžiusių sprendimų naudojo šio modelio modifikacijas [53, 57, 76].

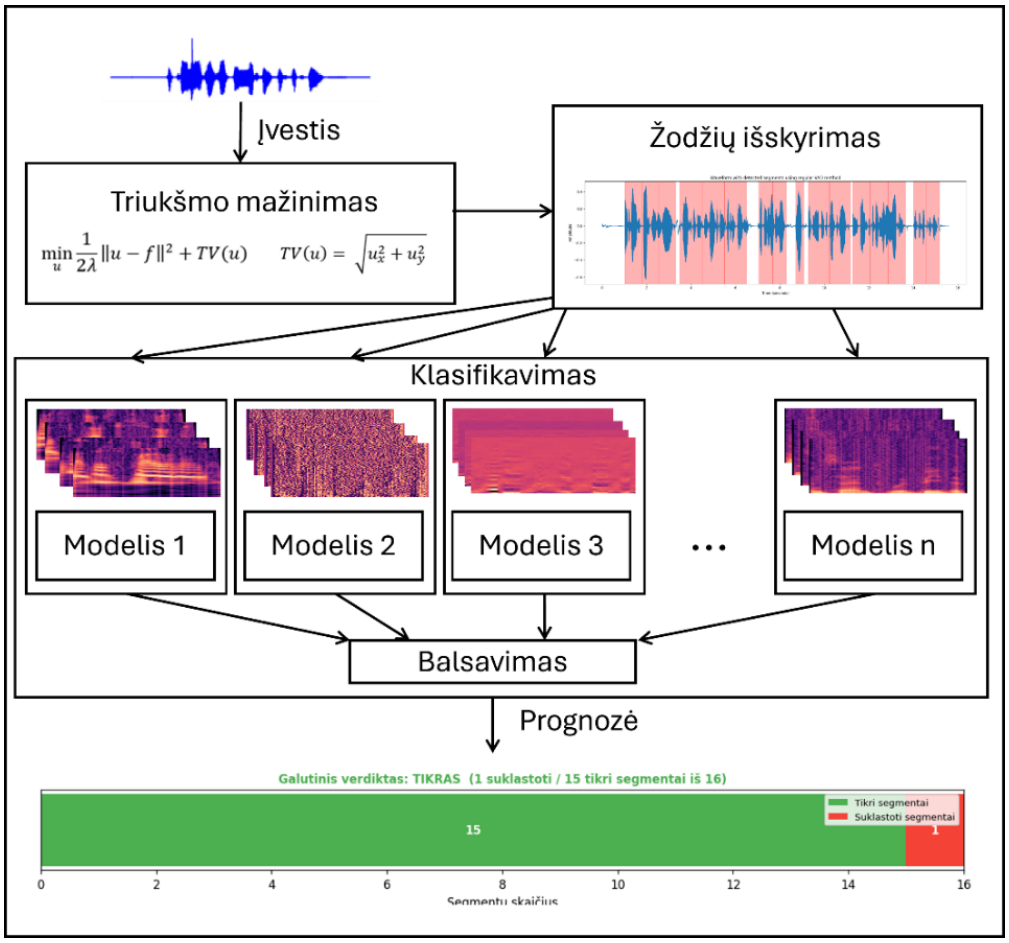
1.6. Literatūrinės apžvalgos apibendrinimas

Šiame skyriuje detaliau paminėti pagrindiniai iššūkiai, su kuriais susiduriama klastotos kalbos aptikimo užduotyje. Iš karto buvo pastebėti du pagrindiniai iššūkiai. Pirmasis iššūkis yra priešiška ataka, kurios pagalba galima sėkmingai apeiti egzistuojančias kalbėtojo atpažinimo sistemas. Antrasis iššūkis yra tai, kad, greitai tobulėjant balso sintezavimo technologijoms, egzistuojantys klastoto balso atpažinimo algoritmai greitai pasensta. Iš priešiškos atakos iššūkio sprendimo būdų analizės paaiškėjo dvi geriausios strategijos – pirmoji iš jų yra modelių mokymo duomenų imtyje naudoti priešišku triukšmu paveiktų garsų pavyzdžius. Antrasis sprendimo būdas yra bandyti iš failo sumažinti priešišką triukšmą, prieš jį pateikiant klasifikatoriui.

Analizuojant egzistuojančius sprendimus buvo pastebėta, kad dabar populiariausios klastotos kalbos aptikimo sistemos dažniausiai naudoja konvoliucinius neuroninius tinklus ir transformerius. Naujesni sprendimai standartiškai naudoja specialius modelius, kurie iš garso įrašų išgauna klasifikatoriams aktualios informacijos vektorius, kurie šiems modeliams padeda tiksliau atskirti klastotus ir tikrus garso įrašus. Iš standartinių garso savybių transformacijų, populiariausios savybės yra tos, kurios pabrėžia aukštus garso signalo dažnius, kur kalbos sintezės modeliai dažnai sukuria artefaktus, kuriuos šios savybės išskiria geriau.

2. Kuriamo klastotos kalbos aptikimo sprendimo vizija

Šiame skyriuje aprašoma, ką tikimasi sukurti tyrimo pabaigoje. Tyrime bus bandoma sukurti tokį klasifikavimo algoritmą, kuris sugebėtų aptikti, ar jam pateiktame garso įraše egzistuoja klastotas ar kaip nors pamodifikuotas balsas. Žemiau pateikta planuojamo sprendimo veikimo logika (žr. **10 pav.**).



10 pav. Planuojamo sprendimo vizija

Sprendimas naudos garso įrašo įvestį, kuri iš pradžių bus apdorota tam tikra triukšmo mažinimo technika, kuri šiam sprendimui bus parinkta po triukšmo mažinimo eksperimentų. Šis žingsnis tikimasi ženkliai sumažinti garso įraše egzistuojantį priešišką triukšmą, kad sekančiuose žingsniuose parinkta garso klasifikavimo technika sugebėtų atskirti, kad įraše girdimas balsas yra klastotas.

Sekančiame žingsnyje bus bandoma iš garso įrašo išskirti girdimus žodžius. Analizuojant kelius egzistuojančius sprendimus, buvo pastebėta, kad jie kaip įvestį gali priimti tik tam tikro, nurodyto ilgio garso įrašus. Jei jiems bandoma pateikti didesnio ilgio garso įrašą, jis dažnai yra padalijamas nurodyto ilgio intervalais, ir tada kiekvienas intervalas suklasifikuojamas atskirai. Tada galutinė prognozė yra visų išvestų prognozių vidurkis. Pagrindinė tokio garso dalijimo strategijos problema yra tai, kad gali atsirasti tokie garso fragmentai, kuriuose negirdima jokia kalba, arba pats dalijimas įvyksta per žodžio vidurį. Tokio sprendimo pavyzdys yra literatūrinės analizės metu tirtas *RawNet3* [72] modelis. Kadangi ir šiame tyrime bus naudojami klasifikatoriai, kurie galės priimti tik griežtai nustatyto ilgio garso įrašus, dėl to ir šiame tyrime reikalinga kokia nors garso dalijimo strategija. Kadangi bandoma atrasti, ar pati girdima kalba yra suklastota, dalijimui bus išskiriami atskiri žodžiai.

Taip bus pašalinta klasifikatoriams mažiau aktuali informacija ir papildomai kiekvienas girdimas žodis nebus perskeltas per kelis segmentus.

Trečiasis ir paskutinis žingsnis yra padalintų garso įrašų klasifikavimas. Šiam žingsniui planuojama naudoti daug įvairių klasifikavimo modelių, kurie komandinio balsavimo principu išves galutinę klastoto garso įrašo tikimybės prognozę. Toks klasifikavimo sprendimas buvo pasirinktas dėl dviejų pagrindinių priežasčių:

- Keli modeliai komandiniu principu turėtų išvesti tikslesnę prognozę nei vienas atskiras modelis. Kadangi kiekvienas modelis gali fokusuotis į skirtingas garso įrašo dalis, bendra kelių tokių modelių prognozė turėtų geriau padengti visą iš garso įrašo išgaunamą informaciją. Metodas yra statistiškai patikimesnis.
- Toks sprendimas turėtų būti atsparesnis priešiškoms atakoms. Standartiškai priešiška ataka bandoma sukurti tokį triukšmą, kuris sugebėtų apgauti vieną specifinį klasifikatorių. Jeigu turime daugiau modelių, triukšmo generavimo užduotis tampa daug sudėtingesnė, kadangi dabar reikia sukurti tokį triukšmą, kuris sugeba apgauti daugiau nei pusę ar daugiau klasifikavimui naudojamų modelių.

Papildomai, geresniam priešiško triukšmo aptikimui garso įrašuose, šių modelių mokymui bus pritaikyta proaktyvi strategija, į modelių apmokymo duomenų imtį pridėjus priešišku triukšmu paveiktų garso įrašų, tikimasi, kad modeliai išmoks teisingai klasifikuoti net triukšmingus duomenis. Tokia klasifikavimo strategija dėl didesnio modelių skaičiaus turėtų veikti lėčiau nei vienas didelis modelis. Dėl to patys atskiri modeliai turėtų būti daug paprastesni. Galiausiai, toks sprendimas leidžia lengvesnį tobulinimą ateityje. Kadangi sprendimas susideda iš kelių atskirų komponentų, jei vienas komponentas pradeda veikti prasčiau, visada galima jį pašalinti, atnaujinti ar pakeisti, nesugadinant kitų likusių komponentų. Tas galioja ir klasifikavimo žingsnyje, kur visada galima pridėti naują tikslesnį, atnaujinti arba, jei reikia, pašalinti prasčiau veikiančius egzistuojančius modelius.

2.1. Sprendimo reikalavimai

Pagrindinis šio projekto tikslas yra atrasti geresnius sprendimus, leidžiančius aptikti dirbtinio intelekto įrankiais suklastotą balsą. Šiuo atveju geresnis sprendimas būtų tas, kuris tiksliau per protingą laiko tarpą gali atlikti prognozę balso autentiškumui nustatyti. Kadangi anksčiau atlikta literatūrinė analizė parodė, kad bet koks dabar atrastas sprendimas tikėtinai bus įveiktas ateityje atsiradus naujoms technologijoms, tyrime ieškomo sprendimo tikslumui užtikrinti ilgesniam laikui buvo nuspręsta naudoti kelis modelius, kurie balsavimo principu išveda prognozę. Galutiniam sprendimui užsibrėžti tokie reikalavimai:

- Bendras modelių balsavimu pasiektas tikslumas turėtų būti didesnis už 95% su modeliams nematytais duomenimis.
- Bendras modelių balsavimo pasiektas DCF įvertis turėtų būti mažesnis už 0,15 su nematytais duomenimis.
- Bendras modelių balsavimo pasiektas EER% įvertis turėtų būti mažesnis už 5% su nematytais duomenimis.
- Kiekvienas individualus klasifikatorius turėtų fokusuotis į skirtingas garso detales (pvz., vienas modelis labiau akcentuojasi į aukštus garso dažnius, kitas į žemus).
- Kiekvienas atskiras klasifikavimo modelis gali naudoti aiškinamojo dirbtinio intelekto (angl. *Explainable artificial intelligence* arba *XAI*) metodus, jų prognozėms paaiškinti.
- Kiekvienas individualus klasifikatorius naudoja skirtingus garso požymius arba modelius.

- Kiekvienas individualus modelis prognozę turėtų atlikti per neilgiau nei 0,5 sekundės. Taip siekiama užtikrinti sprendimo greitaveiką.
- Kiekvienas individualus modelis turi priimti garso bangos formos duomenis. Jei modelis naudoja kitokią garso transformaciją, jis savyje turi turėti visas reikalingas funkcijas garso bangos transformacijai atlikti.
- Iš kiekvieno individualaus modelio turi būti galimybė išgauti garso požymių reprezentacijas, kurios buvo naudojamos garso klasifikavime.
- Bendras sprendimas turi būti atsparus priešiškai atakai.
- Bendras sprendimas turi galėti priimti įvairaus ilgio garso failus.

2.2. Vertinimo kriterijai

Kadangi pradinė planuojama sistema susideda iš trijų pagrindinių dalių, kurios turi atlikti iš principo skirtingas užduotis, visoms trims taikomi skirtingi vertinimo kriterijai. Šiame skyriuje aprašomas klasifikatorių, triukšmo ir garso dalijimo eksperimentų vertinimas.

2.2.1. Klasifikatorių vertinimas

Klasifikatorių vertinimui naudojamos kelios metrikos. Pirma iš jų yra klasifikavimo tikslumas, kuris parodo, kokia dalis klasifikuotų garso įrašų buvo suklasifikuoti teisingai. Papildomai, tikrinamas jautrumas, preciziškumas ir F1. Pastarosios metrikos darbe naudojamos kaip pagalbiniai vertinimai. Detalesniems modelių mokymo ir testavimo rezultatams paaiškinti naudojamos mokymo ir validavimo imčių paklaidos ir tikslumo kitimo kreivės. Detalesniems modelių rezultatų paaiškinimams naudojamos sumaišymo matricos.

Antra svarbi metrika yra *DCF*. Ši metrika yra viena iš pagrindinių *ASVspoof* [5] iššūkiuose naudojamų metrikų modeliams įvertinti. Dėl šios priežasties ją naudosime ir šiame tyrime, kadangi ji leistų šiame darbe sukurtą sprendimą tiesiogiai palyginti su jau egzistuojančiais. Iššūkio organizatoriai *DCF* skaičiavimui apibrėžė tokią formulę (žr. formulė 4):

$$DCF(\tau_{cm}) = \beta * P_{miss}^{cm}(\tau_{cm}) + P_{fa}^{cm}(\tau_{cm}) \tag{4}$$

čia $P_{miss}^{cm}(\tau_{cm})$ yra neteisingai suklasifikuotų teisinguose garso įrašuose procentinė dalis, $P_{fa}^{cm}(\tau_{cm})$ yra neteisingai suklasifikuotų klastotų garso įrašų procentinė dalis, β yra konstanta, kuri pagal iššūkio organizatorius turėtų būti lygi 1,9. Galiausiai, τ_{cm} yra slenkstinė užtikrintumo vertė, kurią peržengus keičiama klastoto balso prognozė. *DCF* yra procentinė neteisingai suklasifikuotų įrašų dalis, kur neteisingai suklasifikuoti tikri įrašai turi didesnį svorį. Tie modeliai, kurių tikslumo ir *DCF* suma yra didesnė už vienetą, parodo, kad modelis klysta abiejose klasėse, ir kuo didesnis *DCF*, tuo dažniau tikri įrašai klasifikuojami kaip klastotės. Dažnai literatūroje šis įvertis yra pažymimas *minDCF* arba *t-DCF*. Abu šie įverčiai naudoja slenkstinę vertę τ_{cm}, *t-DCF* parodo standartinį *DCF* įvertį esantiems specifinėms τ_{cm} reikšmėms. Įvertis *minDCF* yra tiesiog visų patikrintų *t-DCF* verčių minimumas. Šiame tyrime τ_{cm} yra fiksuojama kaip 0,5 ir toliau tyrime naudojama, kaip konstanta.

Papildomai, šią problemą sprendžiančioje literatūroje dažnai sutinkama metrika yra *EER%.* Šis įvertis yra panašus į tikslumą ir parodo, kaip dažnai modelis klysta. Metrikai apskaičiuoti naudojama *DCF* įvertyje jau minėta slenkstinė užtikrintumo vertė τ_{cm}. Bandoma surasti tokią slenkstinę vertę, kur neteisingai suklasifikuotų tikrų ir klastotų įrašų dažniai yra lygūs. *EER%* yra modelio daromų klaidų dažnis esant tai slenkstiniai užtikrintumo vertei. Šis įvertis buvo naudojamas senesniuose *ASVspoof*

iššūkiuose, tačiau pačiame naujausiame šios metrikos buvo atsisakyta, ir dabar modeliams vertinti naudojamos kelios skirtingos *DCF* metrikos, bet kadangi šis įvertis intuityviai parodo, kaip gerai veikia modelis, ji vis tiek dažnai naudojama įvairiems siūlomiems sprendimams vertinti. Ši metrika bus panaudota ir šiame tyrime, tačiau tai bus daroma tik pačiuose paskutiniuose tyrimo etapuose.

Paskutinė svarbi metrika yra modelių greitaveika. Specifiškai tikriname, per kiek laiko vidutiniškai užtrunka atlikti atitinkamą garso transformaciją, ir per kiek laiko išgautą požymį užtrunka suklasifikuoti su kokiu nors klasifikavimo algoritmu.

2.2.2. Triukšmo mažinimo vertinimas

Triukšmo eksperimentai susideda iš dviejų dalių: generavimo ir mažinimo. Abiejų dalių vertinimui reikalingas bazinis (angl. *baseline*) modelis, su kuriuo galima palyginti, kaip pasikeičia modelio veikimo rezultatai, jam pateikus triukšmingus ir valytus duomenis. Specifiškai fokusuojamasi į bazinio modelio tikslumą ir *DCF*.

Triukšmo generavimo eksperimentams įvertinti pateikiami sugeneruoto triukšmo paveikti garso failai ir palyginamos klasifikavimo rezultatų metrikos tarp švarių ir triukšmingų garso failų prognozių.

Triukšmo mažinimo eksperimentuose pritaikius įvairius metodus, bandoma sumažinti praeitame žingsnyje sugeneruotą triukšmą. Kaip ir su generavimo eksperimentais valyti garso failai pateikiami baziniam modeliui, ir po to gauti rezultatai lyginami su standartiniais švariais rezultatais ir triukšmingais rezultatais. Kadangi realiomis sąlygomis kuriamas sprendimas dažniausiai turėtų priimti švarius duomenis, papildomai bus patikrinta, kokie įverčiai gaunami modeliui pateikus švarius duomenis, kurie prieš tai buvo paveikti triukšmo mažinimo metodo. Tai leidžia patikrinti, ar triukšmo mažinimo metodas nesugadina švarių garso failų.

2.2.3. Garso dalijimo vertinimas

Šią dalį yra sudėtinga objektyviai įvertinti. Atliekant eksperimentus bandoma surasti tokį garso įrašų dalijimo būdą, kur padalintuose garso įrašo segmentuose girdėtųsi skirtingi žodžiai. Dalijimo vertinimui naudojamas tas pats garso įrašas, kuriame tariama daug įvairaus ilgio žodžių. Padalijus garso įrašą, rankiniu būdu patikrinami visi atskiri išskirti segmentai ir patikrinama, ar:

- atskirtuose garso failo segmentuose girdisi tik vienas ar keli trumpesni pilni žodžiai;
- atskiruose garso failo segmentuose nesigirdi kitiems garso failo segmentams priklausančios žodžių dalių;
- nėra tokio segmento, kur girdimas tik pašalinis triukšmas ar tyla.

Papildomai, kadangi ši sistemos dalis nėra tokia svarbi, jai tikimasi bus sueikvota mažiausiai skaičiavimo resursų. Dėl to atsižvelgiama į metodų greitaveiką, t.y. per kiek laiko atliekamas pilnas garso failo padalijimas segmentais.

2.3. Eksperimentų aplinkos

Visi šio tyrimo eksperimentai buvo vykdomi tokioje aplinkoje:

- Windows 11 operacinė sistema
- Intel(R) Core i7-9750H CPU @ 2.60GHz
- 16GB operatyviosios atminties
- Nvidia GeForce GTX 1650 GPU

- 4GB dedikuotos GPU atminties su papildomais bendros 8GB GPU atminties iš CPU.
- Cuda 11.8

Visi eksperimentai buvo atliekami *python* 3.10 aplinkoje. Eksperimentams naudotos tokios bibliotekos:

- numpy (1.26.4) – darbo su skaičiais biblioteka.
- scipy (1.11.4) – biblioteka skirta įvairioms matematinėms operacijoms, šiame darbe ji naudojama ieškant tarpų tarp atskirų žodžių garso įrašuose.
- pandas (2.2.1) – naudojama skaitiniams duomenims saugoti ir apdoroti.
- librosa (0.10.0) – darbo su garso įrašais biblioteka. Naudojama, garso failų nuskaitymui ir apdorojimui.
- pydub (0.25.1) – darbo su garso įrašais biblioteka. Naudota įvairiuose garso failų dalijimo eksperimentuose.
- matplotlib (3.8.2) – grafikų braižymo biblioteka.
- spafe (0.3.3) – biblioteka, kuri turi daug funkcijų skirtų įvairioms garso savybėms išgauti.
- tqdm (4.67.1) – biblioteka, kuri buvo naudota patikrinti eksperimentų greitaveika.
- torch (2.1.1+cu118) – mašininio mokymo biblioteka, skirta neuroninių tinklų kūrimui, mokymui ir naudojimui. Biblioteka turi integraciją su *cuda*, kuri leidžia skaičiavimus atlikti grafiniame procesoriuje.
- soundfile (0.12.1) – darbo su garso failais biblioteka. Darbe naudojama garso failų saugojimui.
- webrtcvad (2.0.10) – biblioteka, kuri turi VAD metodo implementaciją.
- TTS (0.22.0) – karkasas turintis daug įvairių balso generavimo modelių. Darbe naudojamas balso įrašo generavimų eksperimentuose.
- transformers (4.46.1) – karkasas skirtas įvairių mašininio mokymo modelių iš *HuggingFace* platformos mokymui ir naudojimui. Darbe naudojamas balso įrašo generavimų eksperimentuose.

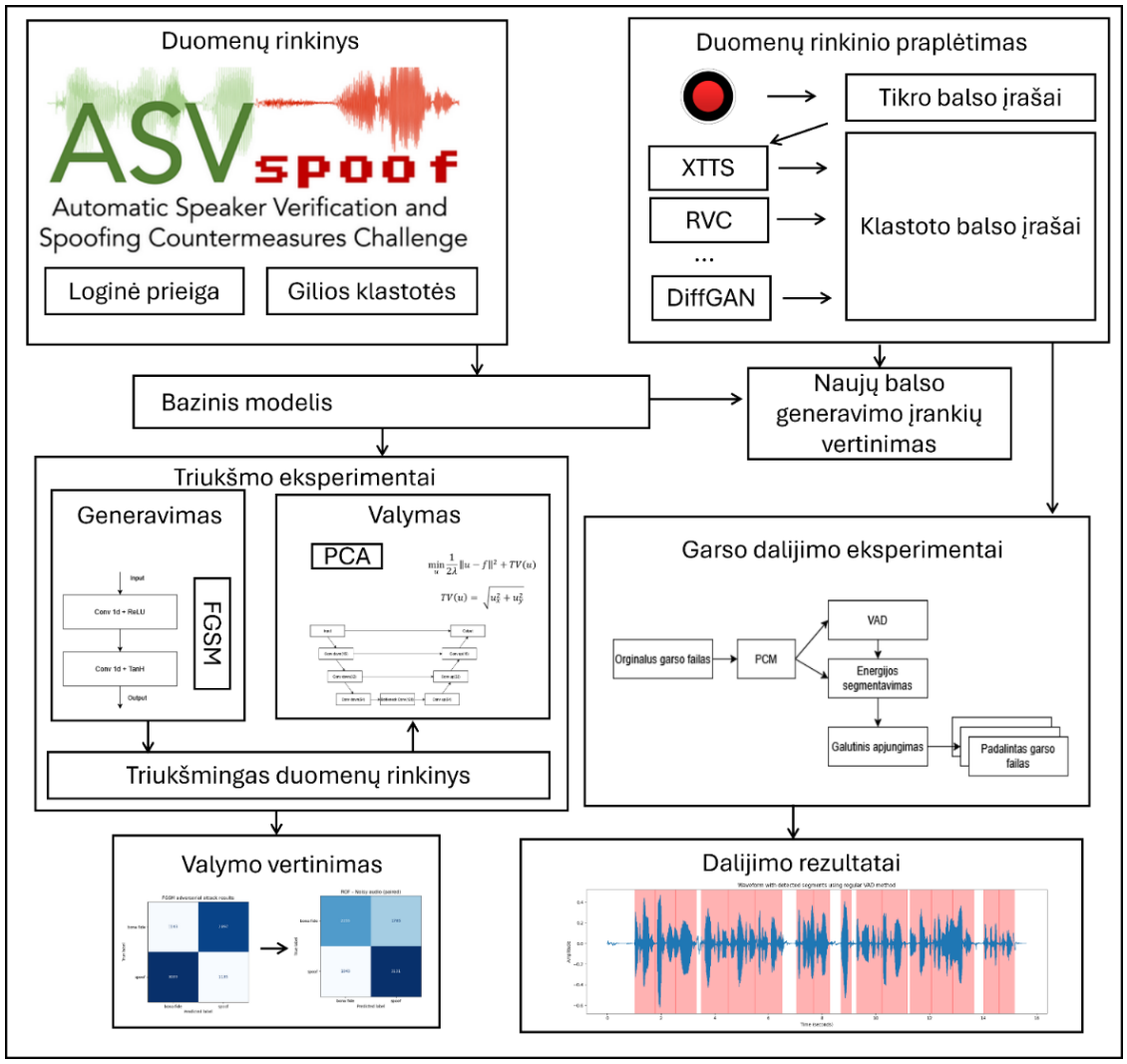
Garso failų klasifikavimo, priešiško triukšmo generavimo eksperimentams ir apskritai mašininio mokymo modelių mokymui buvo naudota *KTU AI notebook* aplinka, kurioje buvo naudotos tos pačios paminėtos bibliotekos.

2.4. Sprendimo vizijos apibendrinimas

Šiame skyriuje aprašyta bendra tyrime kuriamo sprendimo vizija. Ji turėtų susidėti iš trijų komponentų – triukšmo mažinimo, garso dalijimo išskiriant atskirus žodžius ir klasifikavimo su daug modelių balsavimo principu. Po to užsibrėžti kiekvieno komponento vertinimo kriterijai ir bendro apjungto sprendimo reikalavimai. Galiausiai, aprašytos tyrime naudojamos aplinkos.

3. Garso paruošimo eksperimentai

Šiame skyriuje aprašomi visi šiam darbui atlikti eksperimentai, kurie susiję su galutiniam sprendimui pateikiamų duomenų apdorojimu. Žemiau pateiktas grafikas parodo, kaip tyrime vykdyti garso paruošimo eksperimentai susiję vieni su kitais (žr. **11 pav.**).



11 pav. Garso paruošimo eksperimentai

Prieš pradedant vykdyti eksperimentus, pasirinktas duomenų rinkinys. Su šiuo duomenų rinkiniu apmokytas bazinis modelis (angl. *baseline model*), kuris yra ypač svarbus triukšmo eksperimentams, kadangi juose modelis naudojamas sugeneruotiems triukšmams ir išbandytiems triukšmo mažinimo metodams įvertinti. Papildomai, buvo atlikti balso įrašų generavimo eksperimentai, kurie vėliau irgi buvo įvertinti pritaikius šį modelį. Sugeneruoti garso failai buvo panaudoti garso dalijimo eksperimentams, kur buvo išbandytos kelios technikos, skirtos atskirų girdimų žodžių išgavimui iš ilgesnio garso įrašo.

3.1. Duomenų rinkinys

Šiame darbe buvo pasirinkta naudoti duomenų rinkinį *ASVspoof* [5]. Naujausia šio duomenų rinkinio versija yra *asvspoof2024*, tačiau, kadangi ji šio tyrimo vykdymo metu nėra atvirai prieinama, buvo pasirinkta naudoti *asvspoof2021* duomenų rinkinio versiją [77]. Šis duomenų rinkinys padalintas į tris dalis: fizinės prieigos (angl. *physical access*), loginės prieigos (angl. *logical access*) ir gilių

klastočių (angl. *deepfake*). Kadangi darbe tiriamas sintetinio balso aptikimas, darbe naudojamos tik loginės prieigos ir gilių klastočių dalys. Šias dalis bendrai sudaro 793395 įvairaus ilgio garso įrašų failų *flac* formatu. Visi failai yra vieno kanalo (angl. *mono sound*) ir naudoja 16 kHz diskretizacijos dažnį. Trupiausias garso įrašas yra apie 0,4 sekundžių trukmės, ilgiausias apie 29,3 sekundžių trukmės. Iš visų garso failų apie 41069 yra tikrų balsų failai. Visi likę failai (752326 failų) yra balsų klastotės.

Kadangi duomenų rinkinyje klasės yra nesubalansuotos, tyrime buvo pasirinkta naudoti visus tikro balso failus. Balso klastočių failams buvo sukurta atskira imtis, sudaryta iš 41069 atsitiktinai parinktų klastočių failų. Visuose darbe vykdytuose eksperimentuose buvo naudotas šis sumažintas duomenų rinkinys. Jis sudarytas iš 82138 garso failų, kur pusė jų yra tikro balso failai, o kita pusė klastotės.

Modelių mokymo metu sumažintas duomenų rinkinys buvo dalintas į apmokymo, validavimo ir testavimo imtis santykiu 60:30:10%. Visuose mokymuose naudota ta pati atsitiktinių skaičių generatoriaus pradinė sėkla 25, dėl to visos imtys visada turi tuos pačius failus tarp skirtingų modelių mokymų ir eksperimentų, kas leidžia tiksliau palyginti išgautus rezultatus. Verta paminėti, kad šiame rinkinyje nėra priešiška ataka paveiktų garso failų. Dėl to norint atlikti triukšmo eksperimentus, pirma juos reikės susigeneruoti.

3.2. Bazinis modelis

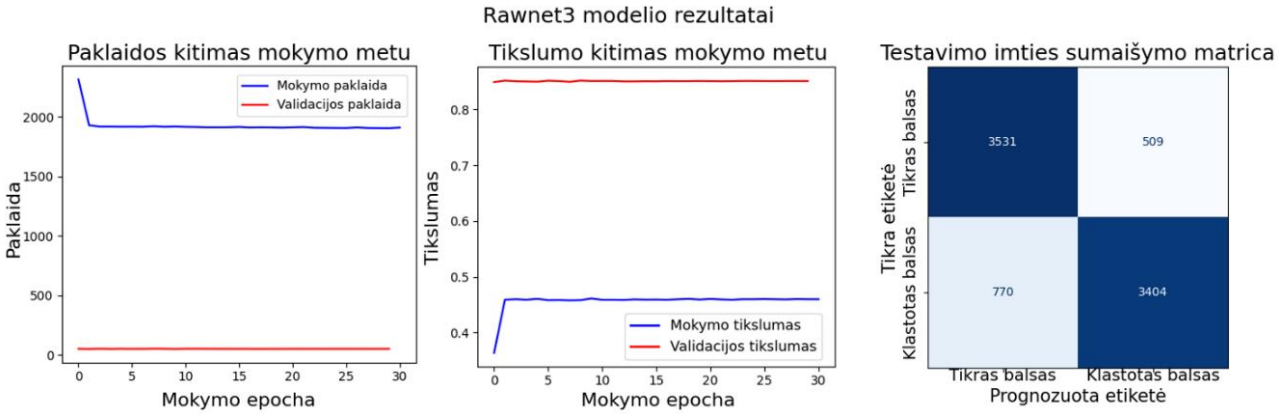
Sekantiems šio skyriaus eksperimentams įgyvendinti reikalingas bazinis modelis. Šis metodas reikalingas generuotų klastočių ir priešiško triukšmo patikrai. Papildomai, šio modelio rezultatus vėliau galima lyginti su šiame projekte pasiūlyto sprendimo rezultatais. Šiam tikslui įgyvendinti buvo pasirinktas *RawNet3* modelis [72]. Šio modelio implementacija ir jo naudojimo kodas yra atvirai prieinamas oficialioje RawNet *GitHub* Repozitorijoje [78]. Modelio svoriai taipogi yra atvirai prieinami per *HuggingFace* aplinką [79]. Papildomai, *ASVspoof5* iššūkyje vienas iš bazinių modelių yra senesnis šios šeimos modelis *RawNet2*. Pasak modelio autorių, naujesnis šios šeimos modelis pasiekia geresnius klasifikavimo rezultatus [72], ir dėl to šio tyrimo vykdymo metu jis yra aktualesnis.

Kaip buvo minėta 1.5.5 skyriuje, RawNet šeimos modeliai yra skirti balso savybių išgavimui. Kadangi atvirai pateikti duomenys išveda tik vektorizuotas balso reprezentacijas, reikia apmokyti klasifikatorių, kuris šią išvestį paverstų galutine prognoze. Tam įgyvendinti buvo pasirinkta apmokyti *AAM-Softmax* klasifikatorių [74]. Pats klasifikatorius yra labai paprastas ir susideda iš vieno pilnai sujungto sluoksnio, kuris ir išveda galutinę prognozę taikant softmax funkciją. *AAM-Softmax* yra kitoks būdas skaičiuoti modelių išvedamus logitus (angl. *logits*), kuris galiausiai išmoko pilnai sujungtą sluoksnį geriau atskirti skirtingas klases. Šiame darbe naudota žemiau pateikta *AAM-Softmax* iteracijos mokymo procedūra.

```
Procedure AAMSoftmax
// input x – garso požymių vektorius
// input labels – tikimasi prognozė
// input w – pilnai sujungto sluoksnio svoriai
// input margin – pridedamoji kampinė riba (angl. additive angular margin)
// input scale – logitų skalės faktorius, prieš juos pateikiant softmax funkcijai
// input optimizer – svorių optimizuotojas
// input new_w = atnaujinti svoriai

begin
  // požymių ir svorių vektoriai normalizuojami
  x_norm = normalize(x)
  w_norm = normalize(w)

  // tarp požymių vektoriaus ir normalizuotų svorių apskaičiuojamas kosinuso panašumas
```

```
  cosine = linear_projection (x_norm, w_norm)
  sine = sqrt (1 – clamp (cosine² between 0 and 1))

  // sinuso kampo panašumo slenkstinė riba pritaikoma tikrajai klasei
  phi = cosine * cos(margin) – sine * sin(margin)

  // atliekamas one-hot kodavimas išvestims
  one_hot = zero matrix same shape as cosine
  for each i in labels:
     one_hot[i, label[i]] = 1

  // apskaičiuojami logitai
  logits = (one_hot * phi) + ((1- one_hot) * cosine)
  logits = logits * scale

  // atnaujinami svoriai
  loss = cross_entropy_loss(logits, labels)
  new_w = optimizer.step(backpropogation(loss), w)
end
```

Klasifikatoriaus mokyme naudota kryžminės entropijos paklaida, kuri naudojo ADAM optimizatorių su 0,001 pradiniu mokymosi greičiu. Papildomai, naudotas kosinuso atkaitinimo reguliarizacijos funkcija (*CosineAnnealingLR*), kuris per visą mokymą mokymo greitį pastoviai mažina iki 0,000005. Mokymas buvo vykdytas 30 epochų ir jame naudotas 64 poaibio dydis (angl. *batch size*). Ankstyvas mokymo stabdymas nebuvo taikomas, nes tariama konvergencija galėjo būti laikina, o po paklaidos padidėjimo modelis galėjo dar geresnius rezultatus. Norint naudoti apmokytą klasifikatorių, sukuriamas naujas pilnai sujungtas sluoksnis, kuriam užkraunami apmokyto sluoksnio svoriai. Žemiau pateiktas bazinio RawNet3 modelio mokymo rezultatas (žr. **12 pav.**).



12 pav. RawNet3 modelio mokymo rezultatai

Kaip matyti, modelis per pirmas kelias epochas sukonvergavo. Kadangi klasifikuojami 256 matmenų vektoriai, modeliui nereikia ilgai mokytis, kad būtų išmokta atskirti skirtingoms klasėms priklausančias garso reprezentacijas. Kadangi apmokymo ir validavimo metu modelių išvestys apskaičiuojamos skirtingai, gautos neįprastos prognozės kitimo kreivės. Galima priminti, jog apmokymo metu *AAM-Softmax* taiko kompleksiškesnę funkciją, lyginant su *Softmax* (žr. **3 formulė**. Kadangi validavimo metu skaičiuojamas tik *Softmax*, modelio validacijos tikslumas pastoviai yra daug aukštesnis už apmokymo tikslumą.

Pagal testavimo imties prognozių sumaišymo matricą matyti, kad modelis išmoko atskirti skirtingas klases, tačiau vis tiek dažnai daromos neteisingos prognozės. Testavimo imtyje modelis pasiekė 0,844 tikslumą ir 0,211 *DCF* įvertį. Tyrimo tikslams šis modelis tinka, kad pamatytume, kaip egzistuojanti technologija susidoroja su esamais balso generavimo įrankiais ir priešiška ataka.

3.3. Balso įrašų generavimas

Papildomam bazinio modelio vertinimui šiame tyrime buvo išbandyti keli naujesni ir populiaresni balso generavimo įrankiai. Literatūros analizės metu buvo pastebėta, kad šioje sferoje egzistuojantys sprendimai greitai pasensta. Kadangi naudojamas 2021 metų duomenų rinkinys, po šių metų sukurtos balso generavimo technologijos turėtų sėkmingiau apgauti anksčiau minėtą bazinį modelį.

Prieš atliekant garso generavimą su *Bandicam* programine įranga buvo įrašyti 116 tikro balso failų. Po to pasitelkus žemiau pateiktas garso sintezės ir balso klonavimo technologijas buvo generuojami klastoto garso failai:

- *Tacotron2* [80] – sugeneruota 110 klastotų garso failų pasitelkus standartinį modelį, kuris yra pasiekiamas per *python TTS* karkasą.
- *yourTTS* [81] – kaip ir su praeitu modeliu sugeneruota 110 failų su standartiniu *TTS* karkaso modeliu. Papildomai, panaudoti prieš tai įrašyti tikri balso failai specifinio pasirinkto balso generavimui.
- *XTTSv2* [7] – identiška situacija *yourTTS* modeliui.
- *SpeechT5* [82] – su specialiai modeliui sukurtu wav2vec modeliu išgauti du skirtingi specifinio balso informacijos x-vektoriai. Vienas naudojo vieną pavyzdinį garso failą, kitas dešimt. Po to abu vektoriai panaudoti sugeneruoti po 110 klastotų garso įrašų tiek garso sintezės, tiek balso klonavimo būdu.
- *RVC* [83] – iš balso duomenų repozitorijos [16] pasirinkta trijų specifinių balso požymių informacija. Tada su šiais balsais balso klonavimo būdu buvo sugeneruota po 114 įrašų.
- *UnitSpeech* [84, 85] – pakoregavus (angl. *fine-tune*) modelio dekoderį, garso sintezės būdu sugeneruota 110 klastoto balso failų.
- *ProDiff* [13, 86] – su standartiniu modeliu sugeneruota 111 garso failų.
- *DiffGAN-TTS* [87, 88] – identiška situacija *ProDiff* modeliui.

Iš viso su visais šiais modeliais buvo sugeneruota 1431 klastoto garso įrašų. Didžiajai daliai išbandytų modelių buvo vykdomi papildomi veiksmai, kurie buvo skirti parinkto balso sugeneravimui. *Tacotron2*, *ProDiff* ir *DiffGAN-TTS* modeliams taikyti standartiniai modeliai, kurie generuoja garso įrašus standartiniais modelių autorių parinktais balsais.

3.3.1. Generavimo rezultatai

Žemiau pateikti visų išbandytų modelių sugeneruotų balso įrašų prognozavimo rezultatai (žr. **1 lentelė**), pasitelkus bazinį *RawNet3* modelį.

1 lentelė. Generuotų balso įrašų klasifikavimo rezultatai

Modelis	**Modelio išleidimo metai**	**Generavimui vykdyti papildomi veiksmai**	**Teisingos prognozės**	**Teisingų prognozių dalis; %**
Tikras balsas	-	-	108/116	93,1
Tacotron2	2018	Ne	104/110	94,5
SpeechT5 TTS	2022	Taip	220/220	100
SpeechT5 VC		Taip	209/210	99,5
yourTTS	2023	Taip	94/110	85,4
DiffGAN-TTS	2022	Ne	92/111	82,8

UnitSpeech	2023	Taip	83/110	75,5
ProDiff	2022	Ne	89/111	80,1
RVC 1	2023	Taip	95/114	83,3
RVC 2		Taip	114/114	100
RVC 3		Taip	72/110	65,5
XTTS v2	2024	Taip	64/111	57,6

Akivaizdi apmokyto klasifikatoriaus tikslumo kitimo tendencija, didėjant garso generavimo technologijos išleidimo metams. Su naujausiu tirtu *XTTSv2* modeliu pasiektas mažiausias tikslumas. Verta paminėti, kad dalis šių garso generavimo technologijų yra paremtos senesnėmis technologijomis, kurios buvo naudotos *ASVspoof2021* duomenų rinkinio sudarymui. Dėl šios priežasties šiuo duomenų rinkiniu mokytas *RawNet3* modelis sugebėjo tiksliau suklasifikuoti anksčiau išleistų balso generavimo technologijų generuotus garso failus.

Skirtingi x-vektoriai *SpeechT5* modeliuose nepasiteisino, nes beveik visi sintezuoti garso failai buvo suklasifikuoti teisingai. Paklausius šiuo modeliu generuotus garso failus galima pasakyti, kad su vienu pavyzdiniu failu sukonstruoto x-vektoriaus sintezuoti failai yra aukštesnės kokybės. Daug pavyzdinių garso failų naudojęs x-vektorius sugeneravo tokius failus, kur labai aiškiai girdisi pauzės tarp skirtingų garsų. Kalba šiuose failuose skamba sulaužyta.

Papildomai, verta paminėti kad dalies šių modelių (pvz, *RVC*, *SpeechT5*), generuojamų garso failų kokybė priklauso nuo modeliui apmokyti ir taikyti pateiktų garso failų. Labai aiškiai tai matyti antrajame *RVC* modelyje. Tikėtina, kad šio modelio naudota balso informacija buvo sukurta taikant prastos kokybės duomenis. Dėl to klasifikatorius sugebėjo teisingai suklasifikuoti visus *RVC2* sintezuotus failus. Kituose dviejuose *RVC* modeliuose naudoti aukštesnės kokybės duomenys ir dėl to klasifikatorius sunkiau aptinka klastotes. Visose technologijose kurioms naudoti triukšmingi arba per tylūs pavyzdiniai balso failai, sugeneruojami prastesnės kokybės garso failai.

Autoregresinių modelių generuotų garso įrašų kokybė buvo geresnė nei difuziją naudojusių modelių. Modeliai, naudoję pavyzdinius garso failus (*yourTTS* ir *XTTSv2*) sugeneruoja tokią kalbą, kurios struktūra yra panaši į pavyzdiniuose failuose girdimą. Jeigu pavyzdiniame faile tarp žodžių yra ilgesnės pauzės, sugeneruotame faile tarp žodžių irgi bus panašaus ilgio pauzės. Jokių pavyzdinių failų nenaudojantys modeliai sugeneruoja kalbą, pasižyminčia nenatūraliai aukštu kalbėjimo tempu.

Galiausiai, verta paminėti, kad nemaža dalis šių modelių (*ProDiff*, *DIffGAN*, *UnitSpeech*, *XTTS*, *yourTTS*) sugeneruoja 22050 Hz diskretizacijos dažnį naudojančius garso signalus. Prieš pateikiant generuotus failus klasifikatoriui, juos pirma reikėjo perskaičiuoti, kad jie atitiktų standartinį 16 kHz dažnį.

3.4. Priešiško triukšmo generavimas

Priešiškai atakai išbandyti pirmiausia buvo bandyta sugeneruoti priešišką triukšmą, skirtą manipuliuoti duomenų rinkinio garso failus. Tyrime tai buvo daroma dviem skirtingais būdais: taikant specialiai mokytus autoenkoderius ir pritaikant *FGSM* ataką bandant optimizuoti originalius garso įrašus.

3.4.1. Autoenkoderiai

Autoenkoderiams išbandyti buvo sukurtos dvi paprastos architektūros. Pirma susideda iš dviejų 1D konvoliucinių sluoksnių, kur pirmasis iš įvestos garso bangos išgauna 32 požymius, kurie vėl yra apjungiami antrajame sluoksnyje. Antroji architektūra identiška pirmajai, tačiau tarp jos dviejų egzistuojančių sluoksnių įterpiami dar du konvoliuciniai sluoksniai, kur pirmasis 32 požymius išplečia iki 64, o antrasis vėl sumažina iki 32. Papildomai, šioje architektūroje naudojamas poaibio normalizavimas (angl. *batch norm*). Abi architektūros visuose sluoksniuose naudoja ReLU aktyvacijos funkcijas. Išeities sluoksniuose naudojama hiperbolinio tangento (TanH) aktyvacijos funkcija, skirta išvesties sutraukimui į [-1;1] intervalą. Pirmosios architektūros konvoliuciniai sluoksniai naudoja tokius parametrus:

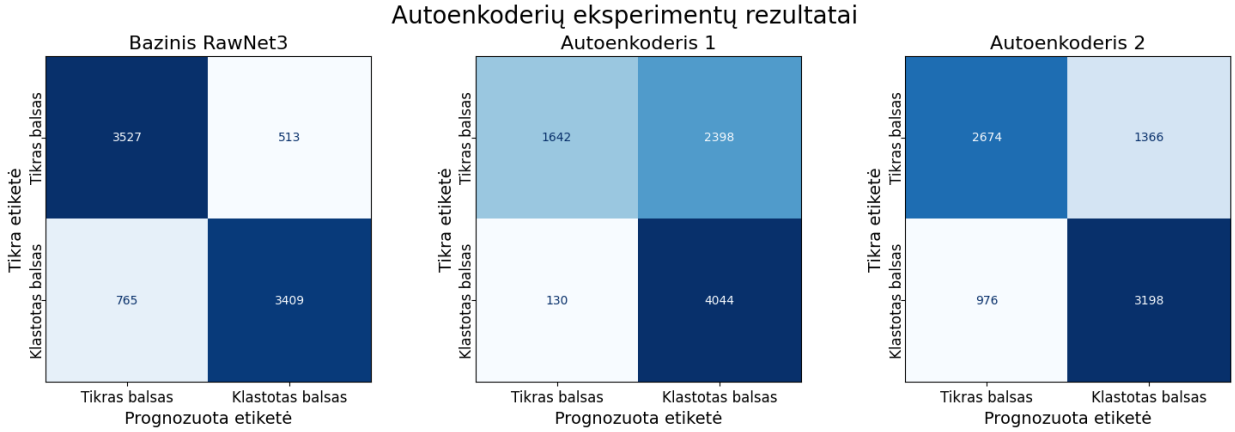
- branduolio dydis (angl. *kernel size*) 3;
- kamšalas (angl. *padding*) 1.

Antrojoje architektūroje pridėti sluoksniai naudoja tokius parametrus:

- branduolio dydis 15;
- kamšalas 7.

Abiejų modelių mokymas vyko konkuruojančių neuroninių tinklų principu. Autoenkoderio sugeneruotas triukšmas pritaikomas atitinkamam pateiktam garso failui, kuris yra pateikiamas baziniam *RawNet3* klasifikatoriui. Tarp klasifikatoriaus išvestų logitų ir pasirinktų klasių apskaičiuojama kryžminės entropijos paklaida. Pirmojoje architektūroje pasirinkta naudoti priešingą klasę tikrajai. Antrojoje pasirinkta klasė visada yra tikras balsas. Su apskaičiuota paklaida atnaujinami autoenkoderio svoriai ir toliau tęsiamas mokymas. Mokymai vykdyti per vieną epochą ir naudotas ADAM optimizavimas su 0,001 mokymo greičiu.

Žemiau pateikti autoenkoderių eksperimentų rezultatai (žr. **13 pav.**).



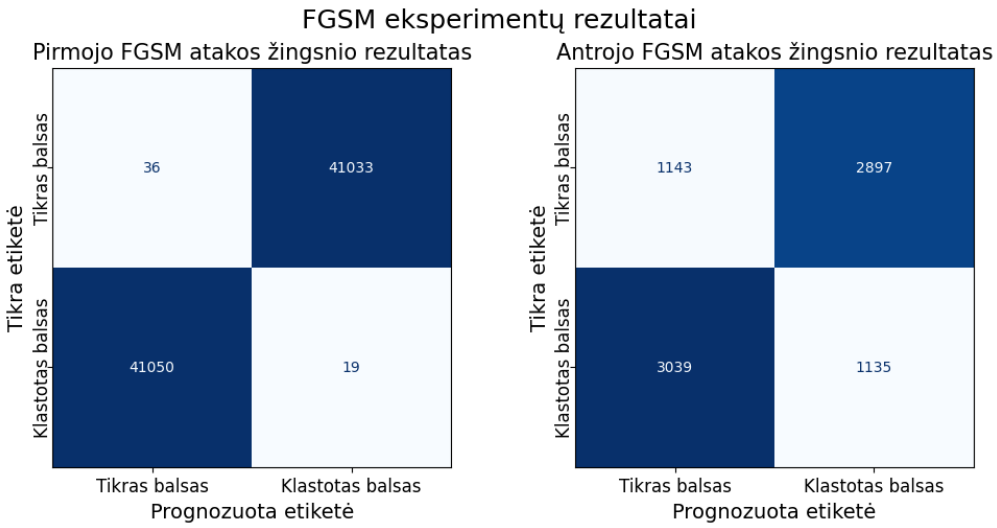
13 pav. Autoenkoderių eksperimentų rezultatai

Pirmojoje matricoje matomi bazinio modelio rezultatai. Vidurinėje matricoje matomas pirmojo triukšmo generavimo autoenkoderio rezultatas. Matoma, kad žymiai nukrito modelio tikslumas, jis nukrito iki 0,69. Akivaizdžiai matyti, kad šio modelio sugeneruotais trikdžiais paveikti garso failai yra žymiai dažniau klasifikuojami kaip klastotės. Kadangi iš eksperimentų tikimasi išgauti tokį triukšmą, kuriuo paveikti klastoti garso failai būtų klasifikuojami kaip tikri, galima teigti, kad pirmoji autoenkoderio architektūra yra netinkama tolesniems triukšmo eksperimentams. Trečiojoje matricoje matomas antrojo autoenkoderio triukšmo generavimo rezultatas. Pritaikius šio modelio generuotą

triukšmą, klasifikatoriaus tikslumas nukrito ne taip žymiai, lyginant su pirmąja architektūra. Čia tikslumas nukrito iki 0,71. Tačiau čia taip pat matoma, kad didesnė dalis klastotų balso įrašų buvo suklasifikuoti kaip tikri. Norint pasiekti geresnius rezultatus, reikėtų naudoti sudėtingesnę architektūrą, ilgiau mokyti sukurtas architektūras arba naudoti kitokias mokymo technikas. Kadangi net šių dviejų labai paprastų autoenkoderių mokymo greitaveika yra labai lėta, galiausiai buvo nuspręsta ieškoti kitokio triukšmo generavimo sprendimo.

3.4.2. Greitojo gradiento ženklo metodas

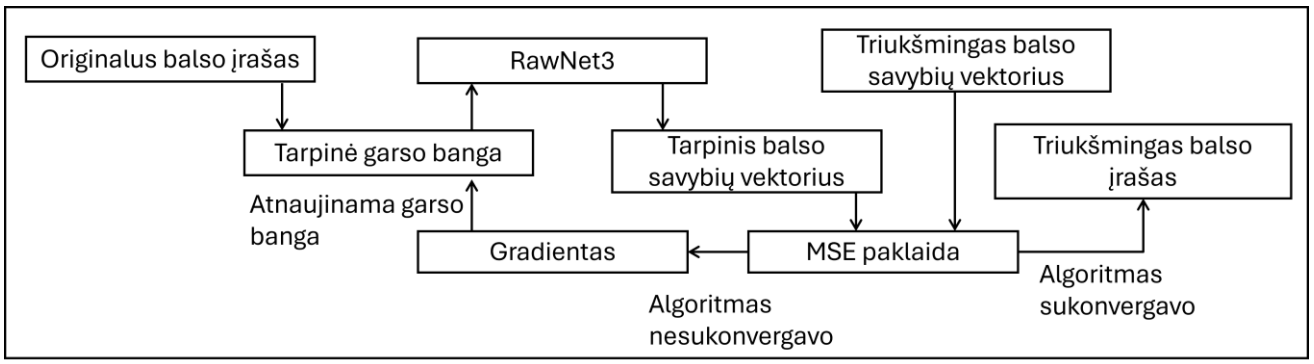
Sekantis išbandytas triukšmo generavimo metodas buvo *FGSM* ataka. Kadangi naudotas *RawNet3* klasifikatorius susideda iš dviejų dalių – paties *RawNet3*, kuris išgauna balso informacijos vektorių, ir klasifikatoriaus, kuris tą vektorių suklasifikuoja, prieš sukuriant triukšmingus garso failus pirma reikia išgauti triukšmingą informacijos vektorių. Dėl to *FGSM* ataką reikia taikyti du kartus. Triukšmingam vektoriui gauti *FGSM* ataka buvo panaudota vieną kartą su mokymosi žingsnio (žr. 1 formulė) ϵ verte 0,4. Taip buvo gauti triukšmingi *RawNet3* įterpiniai, kurių klasifikavimo tikslumas yra žymiai sumažėjęs (žr. **14 pav.**).



14 pav. FGSM eksperimentų rezultatai

Kaip matyti, po pirmo žingsnio, testuojamas *RawNet3* klasifikatorius teisingai suklasifikavo vos 55 garso įrašus iš viso pradinio duomenų rinkinio. Taip yra dėl to, nes pirmajam atakos žingsniui buvo pritaikyta labai agresyvi FGSM forma su aukštu ϵ. Papildomai, kadangi ataka atliekama specifiškai ant RawNet3 įterpinių, kurie yra tiesiog 256 ilgio vektoriai, kurie reprezentuoja atitinkamų garso įrašo požymius, lengva tuos įterpinius pamodifikuoti taip, kad jie būtų klasifikuojami neteisingai.

Turint triukšmingus balso informacijos vektorius pagal žemiau pateiktą procedūrą (žr. **15 pav.**), išgaunami patys triukšmingi garso įrašai. Atitinkami garso failai buvo optimizuojami (įterpiant minimalų triukšmą) taip, kad juos pateikus baziniam modeliui būtų išvesta priešinga klasė.



15 pav. FGSM optimizavimo schema

Procedūros pradžioje užkraunamas originalus (netriukšmingas) garso failas, jis yra apdorojamas taip, kad jį būtų galima pateikti *RawNet3* modeliui. Su šiuo modeliu išgaunamas balso požymių vektorius, kuris yra palyginamas su pirmame žingsnyje sugeneruotu triukšmingo garso informacijos vektoriumi, tarp jų apskaičiuojant MSE (angl. *mean square error*) paklaidą. Ši paklaida naudojama kaip stabdymo sąlyga optimizavimo procesui. Kai ji yra mažesnė už 0,015, arba jos pokytis tarp gretimų epochų yra mažesnis už 0,0005, optimizavimo procesas yra stabdomas. Apdorotam optimizuojamam garsui apskaičiuojamas gradientas, kuriam vėliau pritaikoma ženklo funkcija (žr. 1 formulė). Šios funkcijos rezultatas padauginamas iš mažos vertės ϵ ir yra pridedamas prie tarpinės garso bangos, kas ir yra gradientinės paieškos procedūros eilinė iteracija. Tai yra šios paieškos metu siekiama garso vektorių pakeisti minimaliai, kad būtų apeitas *RawNet* klasifikatorius. Optimizavimo pradžioje ϵ vertė yra 0,0005. Kai MSE paklaida nebekinta, ši vertė sumažinama dvigubai. Šis procesas tęsiasi iki kol procesas sukonverguoja. Tada tarpinis garsas atverčiamas į originalią formą (atkuriama garso trukmė) ir jis įrašomas kaip atskiras failas. Ši operacija vėliau buvo pritaikyta visiems pradinio testavimo duomenų rinkinio garso failams.

Kaip matyti iš antro žingsnio rezultatų (**14 pav.**), po antro FGSM atakos žingsnio tikslumas yra šiek tiek didesnis už pirmojo žingsnio, tačiau pasiektas rezultatas yra daug geresnis nei po prieš tai taikytų autoenkoderių rezultatų (žr. **13 pav.**). Dėl šios priežasties šiuo metodu modifikuoti triukšmingi garso failai naudojami sekančiuose tyrimo etapuose.

3.5. Priešiško triukšmo mažinimas

Sugeneravus triukšmingus garso failus, buvo bandyta atrasti būdą, kaip juos būtų galima atstatyti į kuo panašesnę pradinę formą. Tyrimo metu buvo išbandyta triukšmo mažinimo idėja. Šiam tikslui pasiekti buvo išbandyta principinių komponenčių analizė, triukšmo mažinimas pagal bendrąją variaciją ir galiausiai buvo apmokyti keli U-Net modeliai.

3.5.1. Principinių komponenčių analizė

Triukšmo mažinimui pagal principinių komponenčių analizę buvo nuspręsta garso bangą transformuoti į spektrogramą (žr. **4 pav.**). Transformavimui buvo naudota *librosa* bibliotekos greitosios Furjė transformacijos funkcija, taikant standartinius šios bibliotekos parametrus. Tai reiškia, kad visada naudojamas 2048 dydžio langas su 512 dydžio šuoliais. Kiekvienas garso failo laiko langas turi vertes 1025 dažnių intervalams. Garso laiko langų skaičius priklauso nuo įvestų garso failų ilgio. Pvz., 3 sekundžių ilgio garso failas turi 90 laiko langų. Principinių komponenčių analizei pritaikyti buvo panaudota *sklearn* bibliotekos *decomposition* modulio PCA funkcija. Šiuo

metodu triukšmingas garso dažnių dimensiškumas buvo mažinamas, garso spektrogramą išskirsčius į principines komponentes ir paskui šią spektrogramą atkuriant pagal nurodytą paliktų svarbiausių n komponenčių skaičių. Tyrime buvo išbandyta PCA triukšmo mažinimas su informacijos atkūrimu taikant 10, 15, 20, 30, 40, 50 komponenčių.

3.5.2. Triukšmo mažinimas pagal bendrąją variaciją

Totalios variacijos triukšmo mažinimas (angl. *Total variation denoising* arba *ROF*) taipogi buvo nuspręsta naudoti spektrogramas. Pačiam garso mažinimui pritaikytas Chambolle projekcijos metodas (angl. *Chambolle Projection Algorithm)*, kurio tikslas – suminimizuoti žemiau pateiktas funkcijas (žr. 5 ir 6 formulės):

$$\min_u \frac{1}{2\lambda} \|u - f\|^2 + TV(u) \tag{5}$$

$$TV(u) = \sqrt{u_x^2 + u_y^2} \tag{6}$$

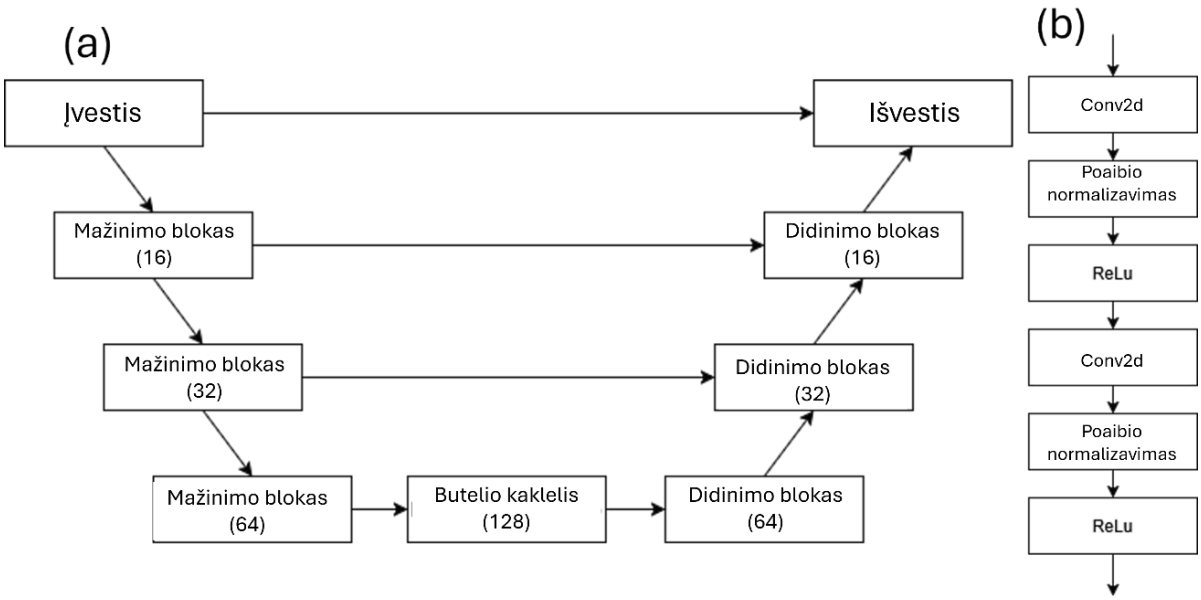
čia λ yra reguliarizacijos parametras, kuris kontroliuoja, kaip agresyviai algoritmas valo įvestą spektrogramą, u ir f yra išvalyta ir originali spektrograma atitinkamai. $TV(u)$ skatina rastis daliniam sklandumui atnaujintoje spektrogramoje u. Panašiai kaip ir su *PCA* triukšmo mažinimu, šis algoritmas siekia pašalinti mažą duomenų variaciją suteikiančias spektrogramos dalis, kuriose randamas triukšmas (maža variacija spektrogramoje lygu mažai informacijos signale).

Iš pradžių pagal pradinę spektrogramą apskaičiuojami skirtumai ($u - f$), kurie parodo, kiek duomenų variacijos yra kiekviename spektrogramos laiko ir dažnio gardelės taške. Tada pagal šiuos kitimus apskaičiuojama apvalyta spektrograma u. Jai apskaičiuojami gradientai x ir y komponenčių atžvilgiu. Gradientai yra sunormuojami ir tada apskaičiuojami gradiento kitimas x ir y komponenčių atžvilgiu. Šie kitimai apsprendžia ieškomos spektrogramos pokytį kitoje iteracijoje. Eksperimente naudota nurodyto iteracijų skaičiaus stabdymo sąlyga. Po 100 iteracijų triukšmo mažinimas stabdomas ir apvalyta spektrograma atkuriama į garso bangą.

Šis metodas turėtų sumažinti atsitiktinius triukšmus išlaikant pagrindinę garso kalbos struktūrą. Priešiškas triukšmas dažnai turi didelio dažnio nenatūralias mikroperturbacijas. *ROF* triukšmo mažinimo metodas šias perturbacijas turėtų sėkmingai pašalinti. Kaip ir su *PCA* metodas *ROF* buvo išbandytas su keliomis λ reikšmėmis.

3.5.3. Triukšmo mažinimas su U-Net

Galiausiai, triukšmo mažinimui buvo išbandyti *U-Net* tipo modeliai. Šie modeliai veikia panašiai kaip autoenkoderiai, tačiau jie sutraukia ir išskleidžia dvimačius duomenis. Dėl to vėl buvo valomos triukšmingų garso failų spektrogramos. Žemiau pateikta eksperimentuose naudota *U-Net* architektūra (žr. **16 pav.**).



16 pav. Naudota U-Net modelio (a) ir konvoliucinio bloko (b) architektūros

Tyrime naudotas *U-Net* modelis susideda iš trijų dvimačių duomenų dimensijų sutraukimo (angl. *downsampling*) ir trijų išplėtimo (angl. *upsampling*) blokų. Mažinimo blokai susideda iš vieno konvoliucinio bloko, po kurio vykdoma maksimalaus grupavimo operacija (angl. *Max pooling*) su branduolio dydžiu 2. Kiekvienas konvoliucininis blokas (žr. **16 pav.**) susideda iš dviejų konvoliucinių sluoksnių, kurių branduolių dydis yra 3, o užpildas lygus 1. Pirmasis blokas priima n vaizdo (spektrogramos) požymių žemėlapį ir juos išplečia iki $2n$ požymių. Po mažinimo blokų duomenys keliauja į butelio kaklelio bloką (angl. *bottleneck*), kuris išlaiko spektrogramos erdvines dimensijas, tačiau požymių (kanalų) skaičių vėl padidina dvigubai. Toliau seka trys didinimo blokai. Kiekvieno bloko pradžioje turimas konvoliucinio transponavimo sluoksnis, kuris įvesto vaizdo aukštį ir plotį padidina du kartus. Po duomenys perduodami į konvoliucinį bloką, kuris dvigubai sumažina išskirtų vaizdo požymių skaičių vaizde. Tarp atitinkamų mažinimo ir didinimo blokų naudojamos šuolių jungtys (angl. *skip connection*). Kiekviename didinimo bloke sujungiamos atitinkamo mažinimo bloko ir ankstesnio sluoksnio reikšmės prieš atliekant tolimesnį apdorojimą.

Šio modelio idėja yra sumažinti spektrogramoje matomus triukšmus. Dėl to prieš atliekant mokymą kiekvienas švarus ir jam atitinkamas triukšmingą garso įrašą sudarantys segmentai buvo paversti spektrogramomis. Abi spektrogramos ašys buvo praplečiamos iki 8 kartotinio. Tai yra svarbu, kadangi, jei įvedamos spektrogramos dimensijos nesidalija iš 8, *U-Net* modelis negali atkurti spektrogramos, kuri būtų tokios pat formos kaip įvesties spektrograma. Papildomai, triukšmingų garso įrašų generavimo metu kartais pasitaikydavo atvejų, kai triukšmingo garso įrašo ilgis kartais keliomis milisekundėmis skiriasi nuo švaraus ilgio. Tokiu atveju skirtingų garso failų ilgiai buvo sulyginti.

Patys *U-Net* modeliai buvo mokomi po 5-20 epochų, priklausomai nuo eksperimento. Verta paminėti, kad šiems modeliams mokyti buvo naudoti ir švarūs, ir triukšmingi duomenys. Visuose eksperimentuose buvo naudotas *ADAM* optimizavimo algoritmas su pradiniu 0,002 mokymo greičiu. Visi eksperimentai naudojo „*ReduceLROnPlateau*“ reguliarizacijos funkciją, kuri mokymo greitį sumažindavo dvigubai kas kart, kai paklaida nekito per vieną epochą. Šiuos modelius reikia apmokyti taip, kad jie neiškraipytų švarių garso failų, kai jie pateikiami modeliui. Kiekvienoje iteracijoje

pateikiant garso failą mokymui sugeneruojamas atsitiktinis skaičius nuo nulio, iki vieno. Jei jis yra mažesnis už eksperimentui pasirinktą slenkstinę vertę, modeliui paduodama švari spektrograma. Kitu atveju paduodama triukšminga spektrograma. Mokymo metu naudojama dviejų atskirų paklaidų suma (žr. formulė 7):

$$loss = 0{,}8 * l1_{loss}(output, clean) + 0{,}2 * mse_{loss}(output, clean) \quad (7)$$

Ši paklaida susideda iš 80% l1 paklaidos tarp *U-Net* apdorotos spektrogramos $output$ ir švarios, priešišku triukšmu nepaveiktos spektrogramos $clean$. Likusi 20% sudaro MSE paklaida tarp tų pačių spektrogramų. MSE apskaičiuojama iš dviejų skaitinių matricų skirtumų kvadratų sumos vidurkio. l1 paklaida padeda geriau išlaikyti apdorojamos spektrogramos struktūrą, tačiau, kadangi ši paklaida taip žymiai nebaudžia didesnių paklaidų, papildomai buvo pridėta vidurkio kvadratinės paklaidos dalis (MSE).

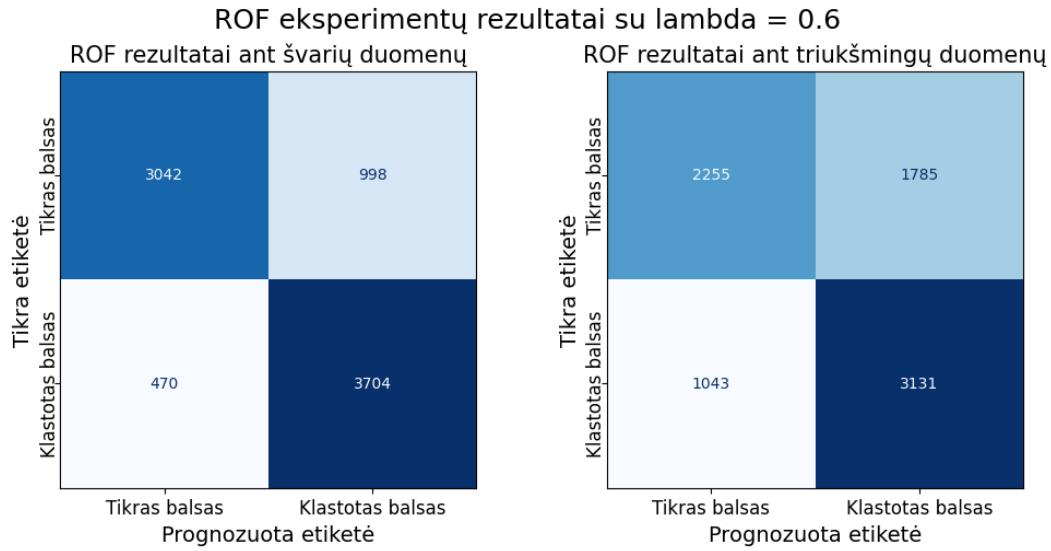
3.5.4. Priešiško triukšmo mažinimo rezultatų apibendrinimas

Žemiau pateikta triukšmo mažinimo rezultatų ištrauka (žr. **2 lentelė**). Pateikti tik geriausi kiekvieno metodo rezultatai švariems ir triukšmingiems duomenims. Pastaba ROF ir PCA modeliai nėra mokomi, dėl to jų mokymo epochos lygios 0. Prie modelio skilties parašyta kokiu metodu buvo atliktas triukšmo mažinimas prieš juos pateikiant *RawNet3* klasifikatoriui. *RawNet3* eilutėse, nebuvo taikytas joks triukšmo mažinimas.

2 lentelė. FGSM triukšmų mažinimo eksperimentų rezultatai

Modelis	**Duomenys**	**DCF**	**Tikslumas**	**Precizišku mas**	**Jautrumas**	**F1**	**Parametrų skaičius**	**Mokymo epochos**
RawNet3	Švarūs	0,211480	0,844290	0,844767	0,845446	0,844253	-	-
RawNet3	Triukšmingi	1,04009	0,277331	0,277421	0,277406	0,277331	-	-
ROF_60	Švarūs	0,288069	0.821281	0,820184	0,826962	0,820111	16280322	0
U_Net_5_ Reduced	Švarūs	0,449606	0,745047	0,743718	0,780601	0,736011	483153	5
PCA_10	Švarūs	0,400353	0,76674	0,764425	0,789074	0,761045	16280322	0
ROF_60	Triukšmingi	0,539871	0,65571	0,654144	0,660324	0,651749	16280322	0
U_Net_5_ Reduced	Triukšmingi	0,5037	0,4963	0,5	0,24815	0,331685	483153	5
PCA_10	Triukšmingi	0,833674	0,484295	0,481043	0,477479	0,46116	16280322	0

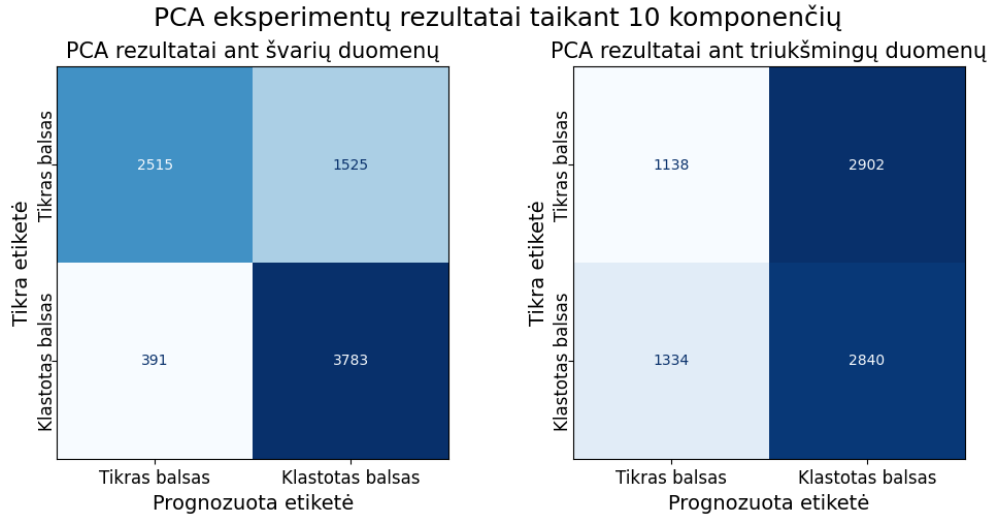
Valant triukšmingus duomenis, geriausiai pasirodė *ROF* modeliai. Specifiškai geriausiai pasirodė metodas, naudojęs $\lambda = 0{,}6$ parametrą. Pats *ROF* algoritmo išvalytų triukšmingų duomenų klasifikavimo tikslumas atrodo prastai, kadangi tikslumas pasiekia tik 65%. Tačiau pažvelgus į šio algoritmo sumaišymo matricą (žr. **17 pav.**) matomi tikslesni rezultatai.



17 pav. ROF algoritmo valytų švarių ir triukšmingų duomenų klasifikavimo rezultatai

Kaip matyti, modelis prasčiau klasifikuoja triukšmu paveiktus tikro balso duomenis. Lyginant triukšmingus suklastotus duomenis, matomas žymus klasifikavimo tikslumo pagerėjimas. Verta paminėti, kad originaliai *RawNet3* klasifikatorius klastotus netriukšmingus įrašus klasifikuodamas suklydo 769 kartus (žr. **13 pav.**). Klastotų duomenų klasifikavimo tikslumas pablogėjo, tačiau tai buvo tikėtinas rezultatas. Didžioji dalis klaidų daroma klasifikuojant tikrus duomenis. Galima teigti, kad ši triukšmo mažinimo technika veikia gerai, kadangi, jei net ir teisingi (neklastoti) duomenys yra paveikti priešiškos atakos triukšmo, jie yra manipuliuoti ir tokius garso duomenis reiktų blokuoti. Valant švarius duomenis pagerėjo klastotės klasės aptikimo tikslumas, tačiau didesnė dalis tikrų garso failų dabar klasifikuojamos kaip klastotės. Tai rodo bazinio modelio gebėjimą pastebėti, kad garso failas buvo pamodifikuotas.

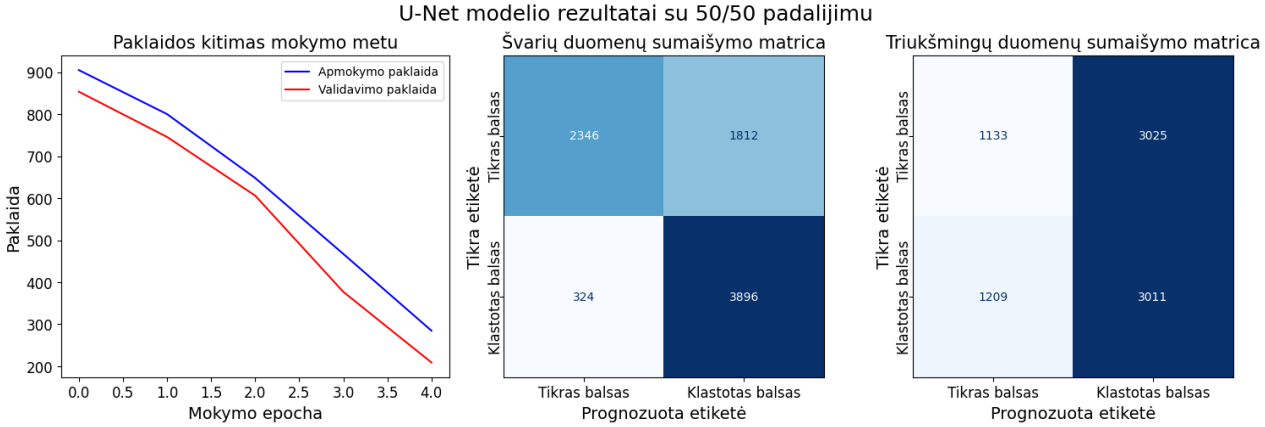
Principinių komponenčių analizės triukšmo mažinimas nepasiekė tokių gerų rezultatų. Kuo daugiau principinių komponenčių paliekama atkuriant garso įrašo spektrogramą, tuo dažniau modelis klysta. Tai rodo, kad triukšmas pasiskirstęs aukštesnio numerio komponentėse, kuriuose yra mažiau bendros variacijos. Pažvelgus į geriausius rezultatus pasiekusį *PCA* garso mažinimo su 10 principinių komponenčių, matomas žemiau pateiktas rezultatas (žr. **18 pav.**).



18 pav. PCA triukšmo šalinamo atkuriant naudojant 10 komponenčių rezultatai

Kaip matyti, šiuo metodu atkurti garso failai dažniau klasifikuojami kaip klastotės. Bandant valyti švarius duomenis, matoma panaši situacija kaip *ROF* metode: klastotės aptinkamos geriau, o tikri garso failai aptinkami prasčiau. Akivaizdžiai matoma, kad klastoti garsai aptinkami tiksliau, tačiau kadangi didelė dalis tikrų balso failų dabar aptinkami kaip klastotės, šis metodas netinka galutiniam sprendimui. Galima priminti, kad klasikinė *PCA* siekia minimizuoti variaciją ortogonaliose išvestinėse (komponenčių ašyse), kas galbūt neatitinka triukšmo kilmės, kadangi triukšmas neturi aiškios struktūros.

U-Net modelio rezultatai yra panašūs *PCA* triukšmo mažinimo rezultatams. Iš visų *U-Net* eksperimentų geriausius rezultatus pasiekė modelis, kuris naudojo 50:50% švarių-triukšmingų duomenų santykį mokymui. Žemiau pateikti šio modelio mokymo rezultatai (žr. **19 pav.**).



19 pav. U-Net 50% triukšmų mažinimo modelio mokymo rezultatai

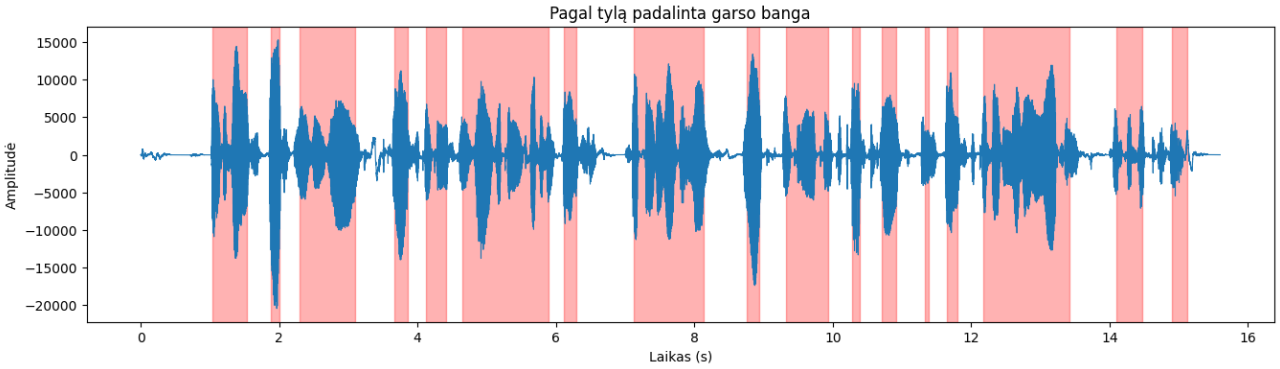
Čia matomas panašus scenarijus, kaip *PCA* 10 modelyje. Didesnė dalis visų modelio valytų įrašų visada klasifikuojami kaip suklastoti duomenys. Kaip ir su kitais dviem metodais, modelis rečiau klysta su valytais švariais duomenimis, klastotų garso failų klasifikavimo tikslumas taip pat pagerėjo. Kadangi modeliui apdorojus triukšmingus duomenis, didžioji dalis tikrų garso failų yra klasifikuojami kaip klastotės, šis sprendimas taip pat netinka galutiniam sprendimui.

3.6. Garso įvesties dalijimas

Vienas iš egzistuojančiuose metoduose pastebėtų trūkumų literatūros analizės metu yra tai, kad sprendimai dažnai priima tik iš anksto nurodyto ilgio garso įrašus. Jei šis ilgis neatitinka, metodai dažniausiai pradeda garso įrašą dalinti dalimis taip, kad kiekviena dalis būtų lygi parinktam ilgiui. Tada kiekviena dalis apdorojama atskirai, ir galutinis rezultatas yra šių dalių išvesčių (vektorių) vidurkis. Toks dalijimas dažnai neatsižvelgia į pačiame garso įraše girdimus žodžius ir dažnai padalijimas įvyksta per juos. Kadangi tikėtina, kad ir šiame darbe bus sukurtas sprendimas, kuris galės priimti tik specifinio ilgio garso įrašus, buvo nuspręsta pabandyti atrasti racionalesnę garso failų dalijimo strategiją. Tam buvo išbandytos kelios garso failo dalijimo strategijos. Visuose eksperimentuose buvo naudotas seniau įrašytas 15 sekundžių ilgio garso įrašas, kur iš viso tariami 35 atskiri žodžiai. Šie žodžiai yra įvairaus ilgio ir tarp jų yra įvairaus ilgio pauzės.

3.6.1. Garso įvesties dalijimas pagal tylos intervalus

Pirma ir pati paprasčiausia iš išbandytų dalijimo technikų yra dalinti garso failą per tas dalis, kur pakankamai ilgą laiką negirdimas pakankamai aukštas garsas. Šiam metodui buvo naudota *pydub* bibliotekos funkcija *split_on_silence*. Žemiau pateiktas tokio padalijimo rezultatas (žr. **20 pav.**).

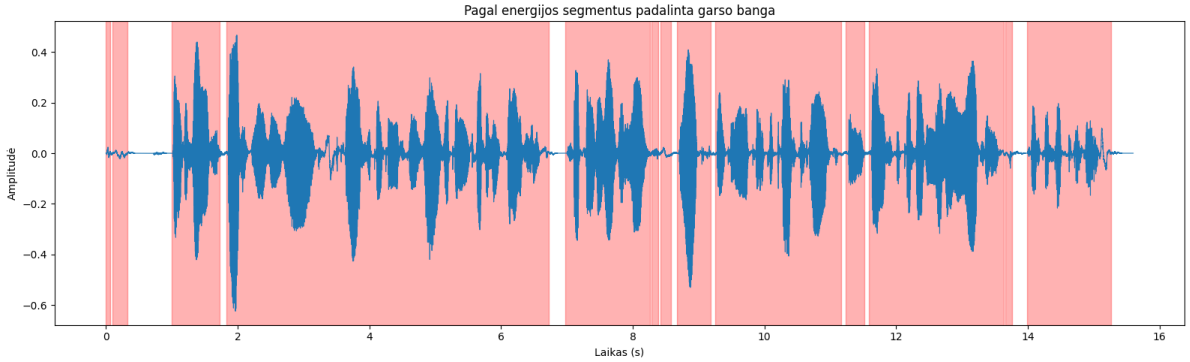


20 pav. Pagal tyla padalinta garso banga

Čia matomas vizualus eksperimente naudoto garso bangos atvaizdavimas. Rausvai pažymėtos dalys yra šio metodo atrasti atskiri girdimų žodžių segmentai. Matoma, kad dalijimas pagal tylos intervalus tikrai aptiko visus atskirus žodžius. Tačiau, pažvelgus atidžiau, galima pastebėti kelis netikslumus. Antras segmentas neaptiko viso žodžio. Kadangi žodžio pabaigoje garso amplitudinė reikšmė neviršijo slenkstinės reikšmės, algoritmas tą dalį traktavo kaip tylą ir jos nepridėjo į segmentą. Papildomai, matomi keli didesni segmentai, kur į segmentą patenka daug žodžių. Galiausiai, kartais nusikerta žodžio pradžia, kaip, pavyzdžiui, priešpaskutiniame segmente. Dėl šių priežasčių, norint šį metodą panaudoti galutiniame sprendime, reikalinga sprendimo modifikacija. Visa strategija yra ne identifikuoti žodžius o pauzes ir jas vėliau pašalinti.

3.6.2. Garso įvesties dalijimas pagal bangos energiją

Antrame dalijimo eksperimente buvo išbandyta garsą padalinti nurodyto ilgio segmentais, tada kiekvienam segmentui apskaičiuoti atitinkamame segmente randamos bangos energija ir atitinkamą segmentą filtruoti pagal parinktą slenkstinę vertę. Galiausiai, visi nufiltruoti segmentai apjungiami iki, kol gaunami visi atskiri įraše girdimi žodžiai. Atmesti segmentai parodo, kur yra pauzės tarp sakomų žodžių. Žemiau matomas šio eksperimento rezultatas (žr. **21 pav.**).

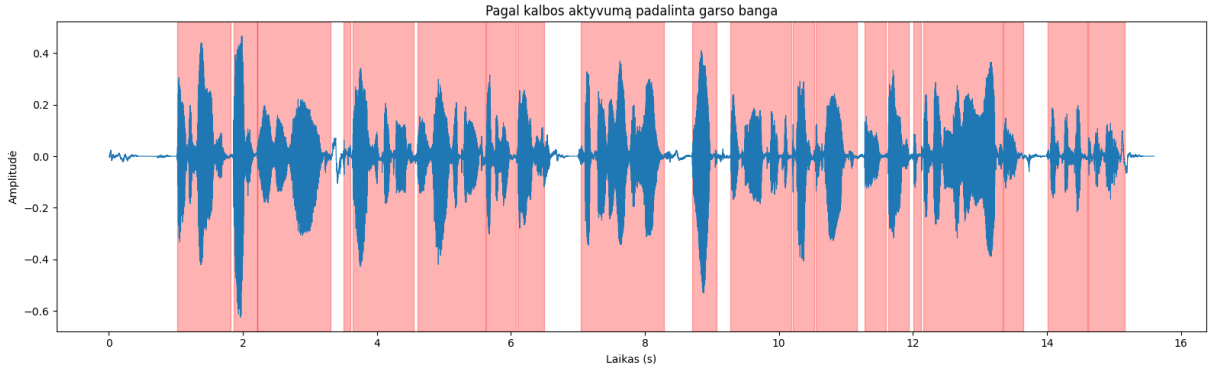


21 pav. Pagal energiją padalintas garso įrašas

Iš visų atliktų eksperimentų, šis metodas veikė prasčiausiai. Metodas sėkmingai aptiko visus žodžius su visomis tų žodžių skiemenimis, tačiau čia išskiriami labai ilgi segmentai, kur į kiekvieną segmentą patenka daug žodžių. Papildomai, matosi, kad buvo sukurti keli segmentai garso failo pradžioje. Čia girdimas pašalinis triukšmas. Dėl šių priežasčių metodas netinka galutiniam sprendimui.

3.6.3. Garso įvesties dalijimas pagal kalbos aktyvumo atpažinimą

Po to buvo išbandytas kalbos aktyvumo atpažinimo (angl. *Voice Activity Detection* arba *VAD*) algoritmas. Šiam algoritmui išbandyti buvo panaudota *webrtcvad* bibliotekos *VAD* funkcija. Žemiau pateiktas šio eksperimento rezultatas (žr. **22 pav.**).



22 pav. Pagal VAD metodą padalintas garso failas

Matoma, kad *VAD* sėkmingai aptiko visus pilnus žodžius visuose segmentuose. Taip pat matyti, kad, lyginant su garso įvesties dalijimo pagal tylos intervalus eksperimentu, nenukerpama nei vieno žodžio pradžia arba pabaiga. Tačiau kaip ir dalijimo pagal bangos energiją eksperimente, iškilo problema, kur į vieną segmentą patenka daug žodžių.

3.6.4. Garso įvesties dalijimas su kalbos transkribavimo modeliu

Galiausiai, buvo išbandytas *OpenAI* korporacijos sukurtas *Whisper* modelis. Šio modelio paskirtis yra garso transkribavimas, tačiau viena iš šio modelio išvesčių yra kiekvieno transkribuoto žodžio pradžios ir pabaigos laikai. Taigi, garso failą pateikus šiam modeliui, gaunama kiekvieno padalinto failo pradžia ir pabaiga originaliame garso įraše. Žemiau pateiktas eksperimento rezultatas (žr. **23 pav.**).

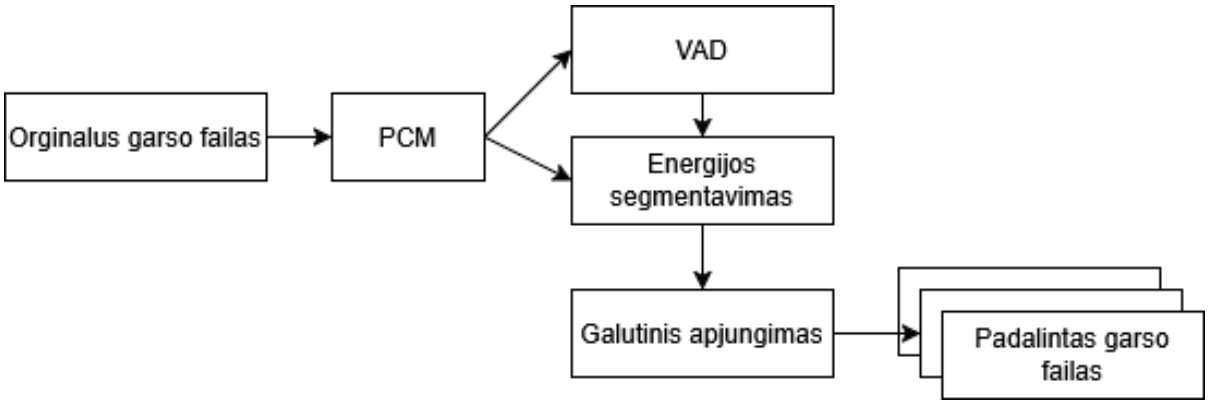
23 pav. Pagal Whisper modelį padalintas garso failas

Šis modelis surado visus 35 atskirų žodžių segmentus. Kadangi modelio paskirtis yra transkribuoti tekstą, papildomai matyti, kokie žodžiai sakomi segmentuose. Kadangi ši technika naudoja dirbtinio

intelekto modelį, ji veikia daug lėčiau nei kiti metodai. Papildomai, net ir šis metodas padarė klaidų dalijant garso signalus. Egzistuoja keli segmentai, kur nukertama žodžio pradžia arba pabaiga. Tai aiškiai matyti penktame segmente, kur girdimas žodis *world*, modelis sukūrė segmentą, į kurį nepatenka žodžio pabaiga. Dažniausiai taip nutinka dėl to, nes ta garso signalo dalis patenka į prieš tai arba po to einantį segmentą. Kadangi *Whisper* modelis yra lėčiausias, reikalauja daugiausiai kompiuterio resursų ir nėra pilnai atviro kodo iš čia tirtų technikų, jis netinka galutiniam sprendimui.

3.6.5. Garso įvesties dalijimas pagal modifikuotą kalbos aptikimą

VAD metodas pasiekė geriausius rezultatus, iš garso failo bandant išskirti atskirus žodžius (pašalinti pauzes). Tačiau dėl jame rastos problemos buvo nuspręsta šiek tiek pamodifikuoti šio metodo veikimą (žr. **24 pav.**).



24 pav. Pamodifikuota garso dalijimo schema

Prieš pradedant garso failo dalijimo procesą, garsas paverčiamas į *PCM* (wav giminės formatas) formatą. Tada šis garsas sudalijamas 10 milisekundžių ilgio segmentais, kuriems vėliau pritaikomas *VAD* algoritmas, ir kiekvienam segmentui išvedama prognozė, ar jame girdima kalba, ar ne. Kadangi ir čia naudojama *webrtcvad* biblioteka, verta paminėti, kaip ji atlieka šį prognozavimą. Kiekvienam segmentui apskaičiuota kelios metrikos:

- spektrinis signalo ir triukšmo santykis;
- spektrinis plokštumas;
- spektrinės energijos pasiskirstymas per dažnius;
- bangos energija;
- pagrindinis tonas (angl. *pitch*).

Šios metrikos vėliau naudojamos apskaičiuoti kelias vidines metrikas, kurios palyginamos su fiksuotomis slenkstinėmis vertėmis. Tada pagal apskaičiuotus rezultatus galutinis sprendimų medis nurodo, ar segmentas yra kalba, ar ne kalba.

Pritaikius *VAD* šie suklasifikuoti segmentai yra toliau dalinami, kadangi tikėtina, kad pasitaikė atvejų, kur į vieną segmentą pateko keli skirtingi pauze atskirti žodžiai. Kiekvienai kalbos sričiai apskaičiuojama išlyginta *RMS* energijos kreivė, kuri išryškina tylą tarp atskirų žodžių. Atrasti pauzės laikai konvertuojami į laiko žymes ir yra naudojami kaip ribos, kurios sudalija *VAD* išskirtus segmentus į dar mažesnes dalis. Šie du procesai garso failą į atskiras dalis sudalija tiksliau.

Šio metodo efektyvumas žemesnis situacijose, kur kokia nors žodžio raidė sakoma ilgiau arba kai garso faile egzistuoja trumpi tarpai tarp skirtingų žodžių. Tokiu atveju gauname didelį skaičių labai

trumpų segmentų, kuriuos reikia apjungti. Segmentai apjunginėjami taip, kad, jei segmento trukmė neviršija 0,15 sekundės, jis yra sudedamas su šalia esančiais taip, kad jie bendrai neviršytų 1 sekundės. Galiausiai, naudojama operacija originalų garso įrašą padalija į 0,15-1 sekundžių ilgio *n* garso įrašų, kur kiekviename įraše girdimas vienas žodis arba labai trumpų žodžių grupė. Žemiau pateikta šios modifikacijos gauti rezultatai (žr. **25 pav.**).

25 pav. Pagal patobulintą VAD padalintas garso failas

Matyti, kad rezultatas primena originalius nemodifikuotus *VAD* rezultatus (žr. **22 pav.**), tačiau dabar dalis segmentų, kurie seniau buvo sujungti, dabar yra padalinti į kelis atskirus. Garso failų padalijimo strategija iš pateikto garso signalo išskiria visus girdimus žodžius, tačiau jų neapjungia į labai ilgus segmentus. Galiausiai, buvo nuspręsta, kad toks dalijimas tyrimui tinka, kadangi susikuria pakankamai trumpi segmentai, kurie pamažina klasifikavimo modeliams reikalingų mokymo parametrų skaičių ir taip padidina jų greitaveiką. Papildomai, garso failai nėra dalijami taip, kad padalijimas įvyktų per žodžio vidurį. Taipogi pašalinama mažiau naudingos informacijos turinčios pauzės.

3.7. Garso įvesties paruošimo apibendrinimas

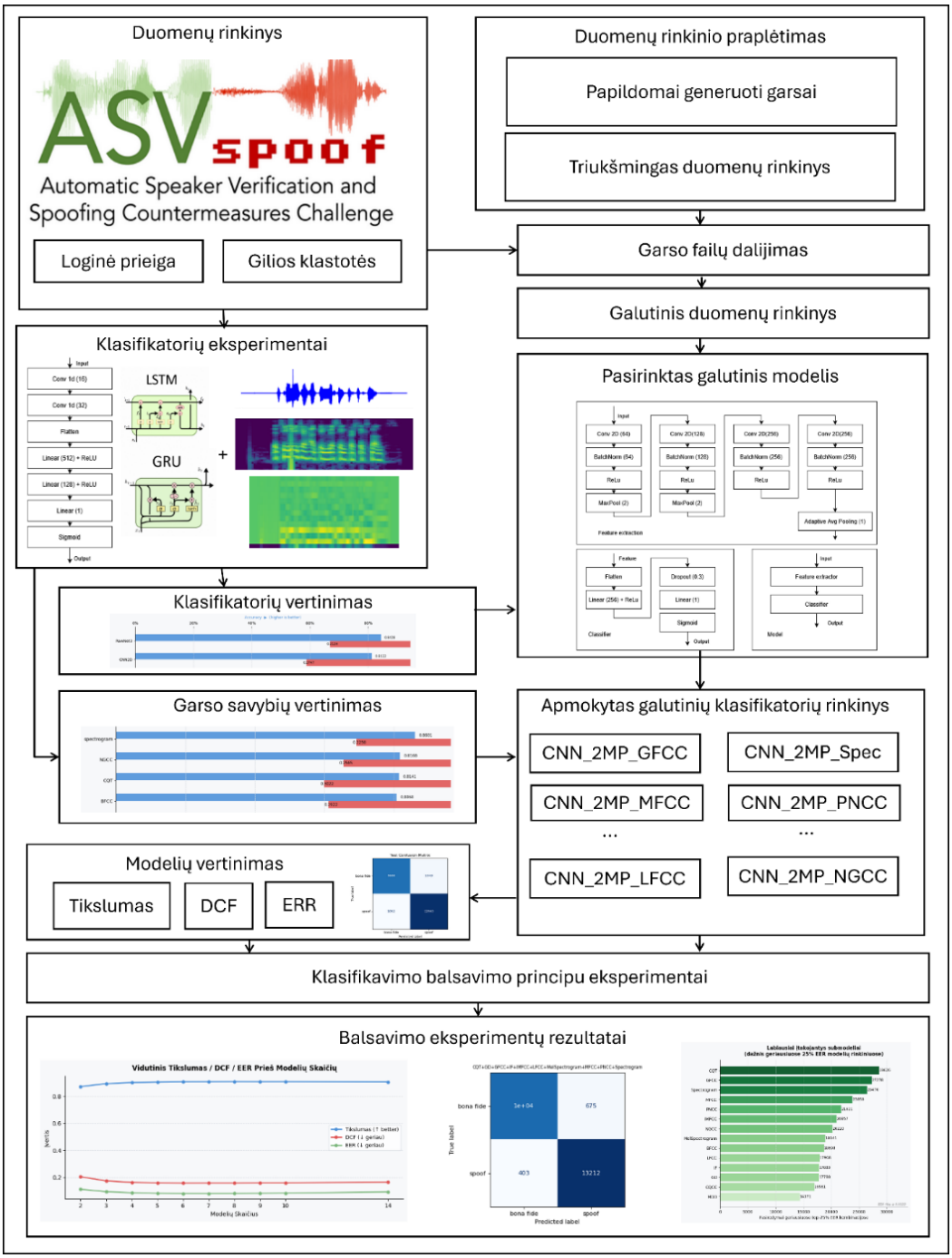
Duomenų paruošimo eksperimentų metu buvo susigeneruota daugiau tikro ir klastoto garso pavyzdžių pasitelkus naujesnius balso generavimo įrankius. Taipogi kiekvienam pasirinkto duomenų rinkinio garso įrašui buvo sugeneruota triukšminga to garso įrašo versija. Iš pačių triukšmo eksperimentų rezultatų *ROF* triukšmo mažinimo modelis pasiekė geriausius rezultatus. Kiti išbandyti triukšmo mažinimo metodai veikė daug prasčiau nei šis pasirinktas metodas.

Po garso dalijimo eksperimentų buvo sukurtas tyrimui geriausias atrastas garso failo padalijimo sprendimas. Šis metodas vėliau buvo panaudotas padalinti visus pradinio duomenų rinkinio ir visus sugeneruotus triukšmingus šio duomenų rinkinio garso failus. Taip buvo sukurtas galutinis duomenų rinkinys, su kuriuo bus mokomi galutiniai pasirinkti klasifikatoriai. Visas šis duomenų rinkinys susideda iš 492917 garso įrašų. Iš šių įrašų 272180 priklauso tikrų įrašų klasei, likę 220737 įrašai priklauso klastočių klasei.

Galiausiai, buvo apmokytas vienas bazinis klastočių klasifikavimo modelis, kurio rezultatus vėliau bus galima palyginti su sekančiame skyriuje aprašyto sprendimo rezultatais. Su šiuo modeliu buvo atrasta, kad jis sunkiau aptinka prieš tai minėtas, naujesniais balso generavimo įrankiais sugeneruotas balso klastotes.

4. Garso klasifikavimo eksperimentai

Šiame skyriuje aprašomi visi šiam darbui atlikti eksperimentai, kurie susiję su balso įrašų klasifikavimu. Žemiau pateiktas grafikas parodo, kaip tyrime vykdyti klasifikavimo eksperimentai susiję vieni su kitais (žr. **26 pav.**).



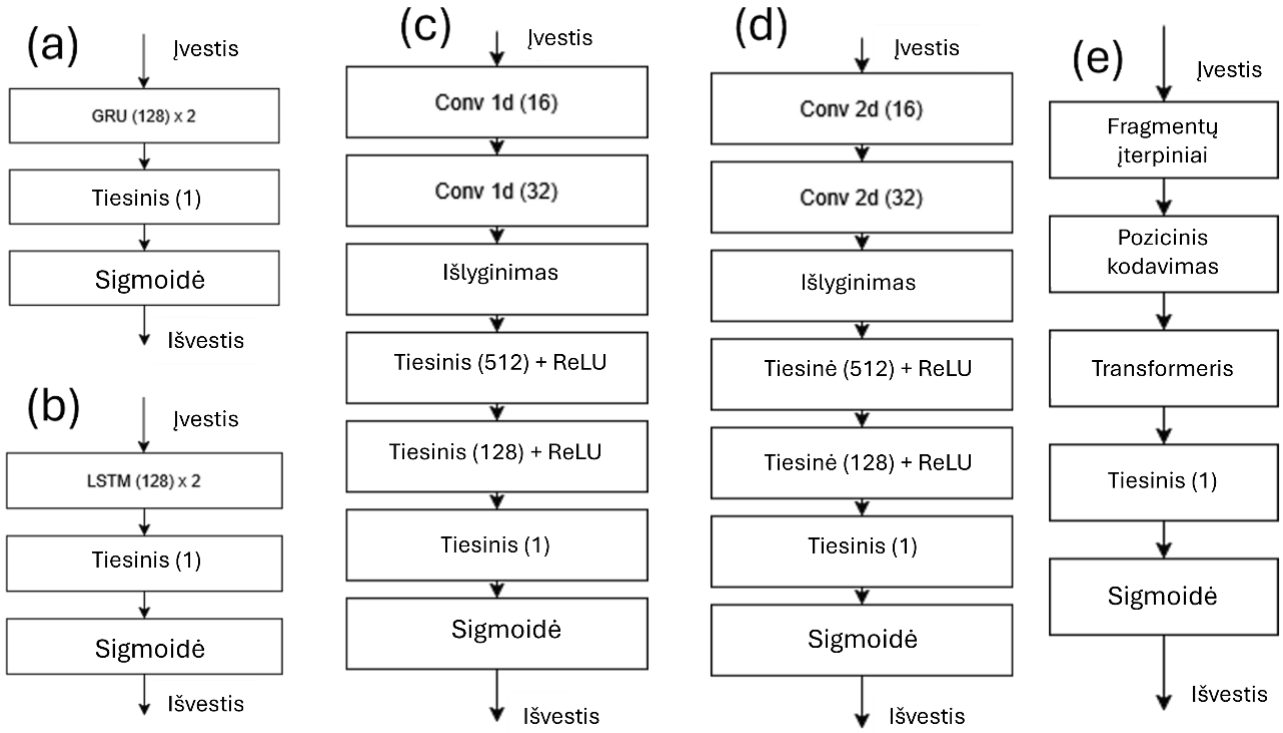
26 pav. Garso klasifikavimo eksperimentai

Visas šis skyrius susideda iš dviejų pagrindinių dalių. Pirmoji dalis naudoja tik pradinio *ASVspoof2021* duomenų rinkinio duomenis. Jų paskirtis yra atrasti, kokie klasifikavimo algoritmai ir garso požymiai geriausiai sugeba atskirti tikrą ir klastotą balsą. Tam tikslui pasiekti buvo apmokyta daug įvairių modelių. Pagal geriausius rezultatus pasiekusį modelį buvo sukurta galutinė klasifikatoriaus modelių architektūra. Šiems modeliams apmokyti naudotas, po duomenų paruošimo

eksperimentų sukurtas galutinis duomenų rinkinys. Modeliai mokomi geriausiai pasirodžiusiais garso požymiais klasifikuoti. Apmokius visus modelius, jie buvo įvertinti ir naudoti paskutiniuose šio tyrimo eksperimentuose – klasifikavimo balsavimo principu. Šių eksperimentų eigoje ieškoma geriausius klasifikavimo eksperimentus pasiekiančių modelių ir balsavimo strategijų kombinacijų.

4.1. Geriausių klasifikatorių ir savybių parinkimas

Pirmasis klasifikavimo eksperimentų žingsnis yra atrasti, kokios garso savybių ir garso klasifikavimo metodų kombinacijos geriausiai atskiria tikrus ir generuotus balsus. Šiam tikslui pasiekti buvo išbandytos visos 1.5 skyriuje atvaizduoti garso požymiai ir transformacijos (žr. **3 pav.**, **4 pav.** ir **5 pav.**). Patiems klasifikatoriams buvo pasirinkta išbandyti konvoliucinius neuroninius tinklus, *GRU*, *LSTM* rekurentinius tinklus ir transformatorius. Žemiau pateiktos tirtų klasifikatorių architektūros (žr. **27 pav.**).



27 pav. Eksperimentuose naudotų klasifikatorių architektūros: GRU (a), LSTM (b) architektūros, 1D (c) ir 2D (d) konvoliucinių neuroninių tinklų architektūros ir transformerių (e) architektūra

Abi rekurentinės architektūros (žr. **27 pav.**) naudojo po du atitinkamus rekurentinius sluoksnius, kurių paslėptas požymių dydis buvo lygus 128. Tada šių sluoksnių išvestys pateikiamos pilnai sujungtam sluoksniui, kuris išveda vieną skaičių, kuris pateikiamas sigmoidės aktyvacijos funkcijai. Modelių įvesties dydžiai priklauso nuo savybės, kurią naudojama modeliui apmokyti.

Abi konvoliucinės architektūros, (žr. **27 pav.**), pradžioje naudoja po du konvoliucinius sluoksnius. 2d architektūroje naudojami tokie parametrai: branduolio dydis 4, žingsnis 2 ir kamšalas 1. 1d architektūroje standartiškai naudojami tokie parametrai: branduolio dydis 5, žingsnis 1 ir kamšalas 1. Mokant modelius su savybėmis, kurių ilgis yra visas garso failas kaip garso banga naudoti tokie parametrai: branduolio dydis 100, žingsnis 50, kamšalas 1. Po to konvoliuciniais sluoksniais apdorotos savybės yra išlyginamos ir pateikiamos dviem pilnai sujungtiems sluoksniams su *ReLU* aktyvacijos funkcija, kurie išlygintas savybes sumažina iki 512 ir 128 ilgio vektorių. Galiausiai,

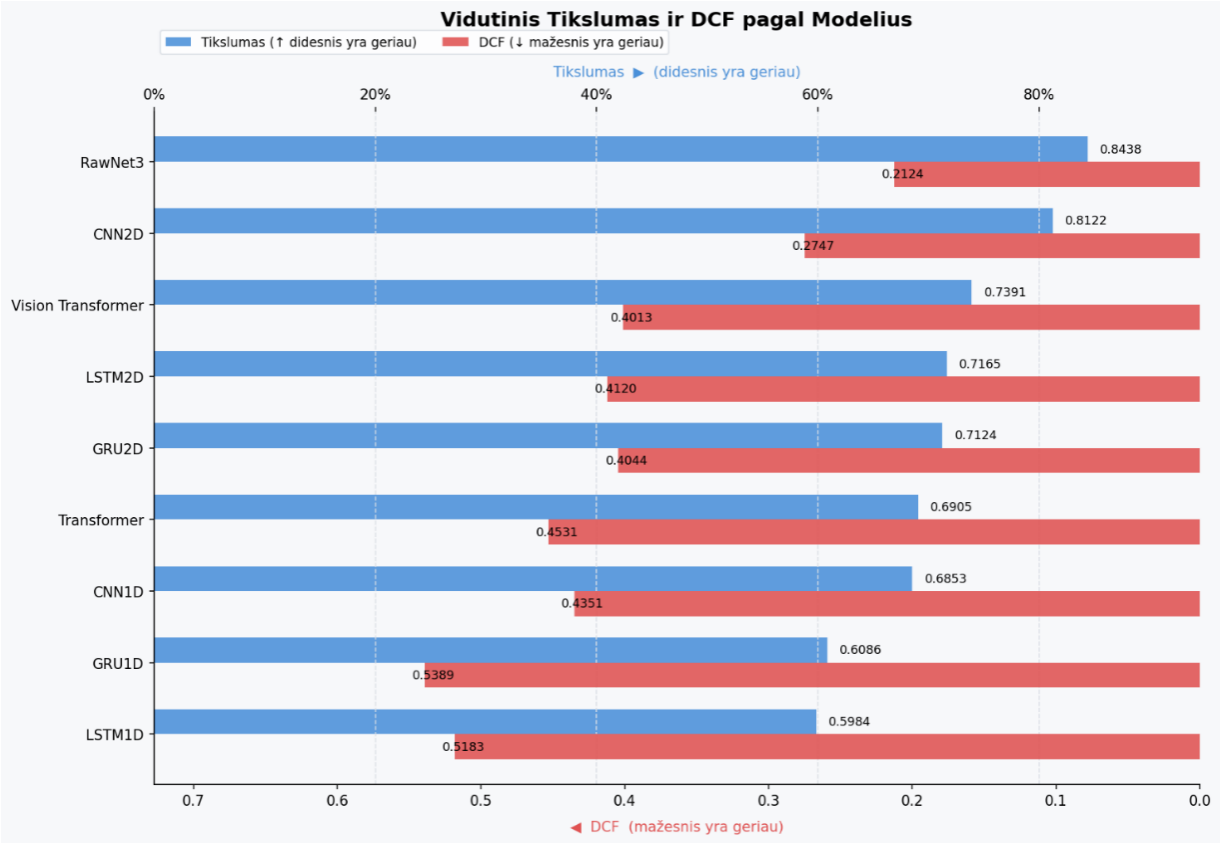
duomenys keliauja į paskutinį pilnai sujungtą sluoksnį, kur sigmoidės funkcija išveda prognozės rezultatą. Visuose darbe kurtuose klasifikatoriuose galutiniam modelio rezultatui gauti, buvo pasirinkta sigmoidės aktyvacijos funkcija. Toks sprendimas priimtas dėl to, kad nagrinėjama užduotis yra binarinio klasifikavimo pobūdžio, todėl vienos išvesties neurono pakanka klasės tikimybei įvertinti. Naudojant *Softmax* aktyvacijos funkciją dvejetainio klasifikavimo uždavinyje, reikėtų dviejų tarpusavyje priklausomų išvesčių, kurių pateikiama informacija iš esmės yra perteklinė. Be to, sigmoidės funkcija leidžia gauti tiesioginį ir lengviau interpretuojamą pasirinktos klasės pasikliautinumo įvertį, kurį galima tiesiogiai taikyti *DCF* ir *ERR%* metrikoms apskaičiuoti. Modelių mokomų parametrų skaičiai vėlgi priklauso nuo įvesties dydžio, kuris priklauso nuo tiriamos transformacijos.

Galiausiai, eksperimente išbandyti transformatoriai (žr. **27 pav.**). Prieš pradedant klasifikavimą, duomenys paduodami fragmentų įterpinių (angl. *patch embedding*) sluoksniui. Šiame sluoksnyje originali įvestis padalijama ir atskiros dalys yra tokenizuojamos. Vienmatėse savybėse vienas tokenas atitinka 400 verčių garso bangos ir autokoreliacijos savybėse. Visose kitose vienmatėse savybėse vienas tokenas atitinka 2 vienetus. Dvimatėse savybėse vienas tokenas atitinka *X* ant *X* dydžio originalios įvesties iškarpą. Čia *X* atitinka mažiausią iš šių dydžių: 25, 16, 8, 5, 4, 3, 2, priklausant nuo to, iš kurio didžiausio skaičiaus dvimatės įvesties dimensijos dalijasi. Pozicijos kodavimo sluoksnis praėjusio sluoksnio išvestiems tokenams sukuria vektorius, kurie nurodo tų tokenų eiliškumą. Tada tokenai keliauja į patį transformerį. Abiejuose modeliuose transformeriai naudoja 4 dėmesio galvas (angl. *attention heads*) ir aštuonis kodavimo-dekodavimo sluoksnius. Transformerio išvestis pateikiama vienam pilnai sujungtam sluoksniui, kuris naudojant sigmoidės funkcija atlieka klasifikavimą.

Visi šie modeliai buvo mokyti taikant ADAM optimizavimo algoritmą, kurio mokymo greitis yra 0,0001. Paklaidai skaičiuoti naudojama binarinė kryžminė entropija. Papildomai, naudojamas „*ReduceLROnPlateau*“ reguliarizacijos funkcija, kuris mokymo greitį sumažina dvigubai, kas kartą, kai validavimo imties paklaida nekinta dvi epochas. Rekurentiniai ir vienmačiai konvoliuciniai modeliai buvo mokyti po 30 epochų. Daugiamačiai konvoliuciniai modeliai mokyti po 20 epochų. Transformeriai mokyti po 5 epochas, vaizdo transformeriai mokyti po 30 epochų.

4.1.1. Geriausi klasifikatoriai

Skirtingų klasifikatorių ir savybių eksperimentams iš viso buvo apmokyti 102 modeliai. Vidutiniai skirtingų modelių rezultatai per visas savybes pateikti žemiau matomame paveikslėlyje (žr. **28 pav.**).



28 pav. Tikslumo ir DCF vidutiniai vertinimai per skirtingus modelius

Kaip matyti, geriausius rezultatus pasiekė bazinis *RawNet3* modelis. Čia verta paminėti, kad iš viso šis modelis buvo apmokytas tris atskirus kartus ir kad jis visada naudoja neapdorotas garso bangas. Iš tirtų atskirų modelių akivaizdžiai matoma, kad dvimates garso transformacijas naudojantys metodai pasiekia geresnius rezultatus, nei taikant vienmačius požymius. Iš vienmačių požymių transformeriai pasiekė geriausius rezultatus, kas buvo tikėtinas rezultatas. Tačiau iš dvimačių savybių konvoliuciniai tinklai pranoko vaizdo transformerių įverčius. Taip galėjo nutikti dėl to, nes tiriamos dvimačių savybių dimensijos yra mažos ir dėl to nesuteikia pakankamai duomenų transformeriams apmokyti. Abi rekurentinių tinklų architektūros pasirodė blogiausiai ir tarp vienmačių, ir tarp dvimačių savybių. Skirtumai tarp skirtingų rekurentinių modelių yra minimalūs, tačiau verta paminėti, kad *GRU* modeliai pasirodė geriau su vienmačiais požymiais, o *LSTM* su dvimatėmis transformacijomis. Kadangi tikėtina, kad visų tiriamų požymių sąraše egzistuoja bent keli, kurie pasiekia labai prastus rezultatus ir taip nusveria šio grafiko įverčius, žemiau pateikti keli aukščiausius tikslumus pasiekę modeliai (žr. **3 lentelė**).

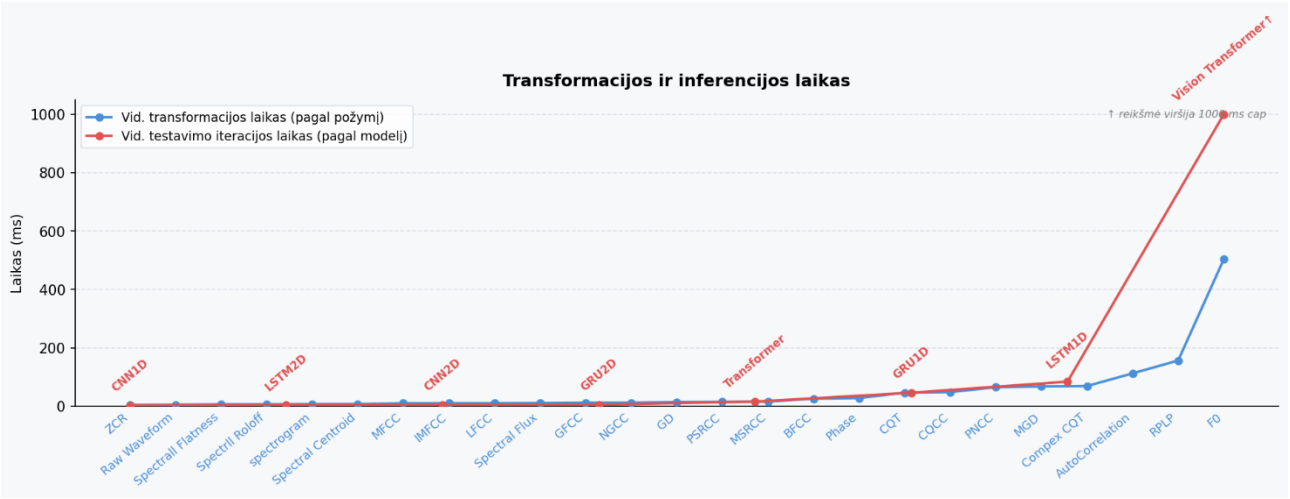
3 lentelė. Klasifikatorių eksperimentų rezultatų iškarpa

Garso požymis	**Modelis**	**DCF**	**Tikslumas**	**Preciziškumas**	**Jautrumas**	**F1**	**Mokomi parametrai**	**Mokymo epochos**
Spektrograma	CNN2D	0,065108	0,951546	0,95172	0,951619	0,951545	96543793	20
NGCC	CNN2D	0,11422	0,921719	0,921672	0,92172	0,921694	3712049	20
PNCC	CNN2D	0,118907	0,916484	0,916541	0,916448	0,916473	3712049	20
GFCC	CNN2D	0,136182	0,898223	0,898609	0,898973	0,898214	3712049	20
BFCC	CNN2D	0,180485	0,880326	0,880042	0,880791	0,880205	3712049	20
Spektrograma	GRU2D	0,19541	0,869126	0,868886	0,869421	0,869017	542721	30

Spektrograma	LSTM2D	0,254431	0,85963	0,857747	0,880001	0,857239	723585	30
CQT	CNN2D	0,228537	0,852325	0,851764	0,853955	0,85198	7988273	20
Garso banga	RawNet3	0,21148	0,84429	0,844767	0,845446	0,844253	16280834	30
NGCC	GRU2D	0,231495	0,84356	0,843349	0,843729	0,843448	154113	30
GD	CNN2D	0,239786	0,830777	0,830934	0,83087	0,830774	96543793	20
MFCC	CNN2D	0,278074	0,813854	0,813509	0,814295	0,813633	1205297	20
MGD	CNN2D	0,26663	0,81215	0,812309	0,812252	0,812148	96543793	20

Šioje lentelėje paryškinti geriausi bazinio modelio rezultatai. Kaip matoma, egzistuoja kelios modelių ir garso savybių kombinacijos, kurios net šio eksperimento metu pasiekė geresnius rezultatus nei pasirinktas bazinis modelis. Aukščiausią tikslumą pasiekė spektrogramą naudojęs konvoliucinis modelis. Tačiau šį įvertį galima vadinti išimtimi, kadangi pats modelis yra didesnis. Nemaža dalis geriausius rezultatus pasiekusių sprendimų su žymiai mažiau mokomų parametrų pasiekė panašius rezultatus. Pilna rezultatų lentelė pateikta prieduose (žr. **1 priedas**).

Modelių greitaveiką galima numatyti pagal modelio parametrų dydį. Daugiau parametrų turintis modelis užtruks ilgiau apmokyti ir prognozes atliks lėčiau, lyginant su mažiau parametrų naudojančiais modeliais. Tiksliai greitaveikai įvertinti buvo apskaičiuoti eksperimentuose mokytų modelių vidutiniai inferencijos (angl. *inference*) laikai ir garso požymių išgavimo laikai (žr. **29 pav.**).

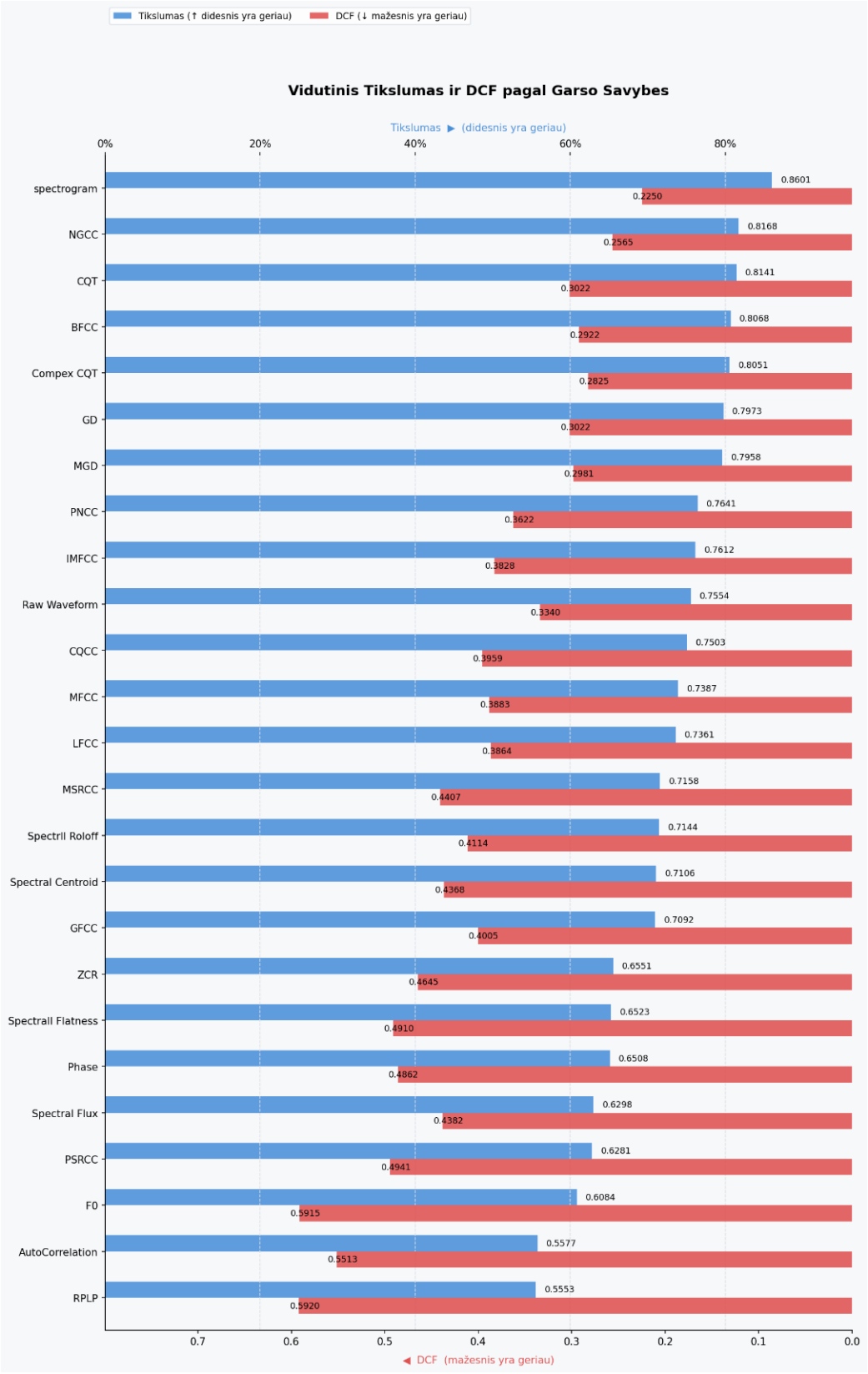


29 pav. Vidutinis garso savybių išgavimo laikas ir vidutinis modelių prognozės išgavimo laikas

Kaip matyti, beveik visi modeliai prognozes atlieka per labai trumpą laiką. Vienintelė išimtis yra vaizdo transformeriai, kurie prognozę atlieka per ~6 sekundes. Iš garso savybių matyti, kad visų savybių išgavimo laikas irgi yra labai trumpas ir vienintelė išimtis yra pagrindinis tonas, kurį išgauti vidutiniškai užtrunka pusę sekundės.

4.1.2. Geriausi garso požymiai

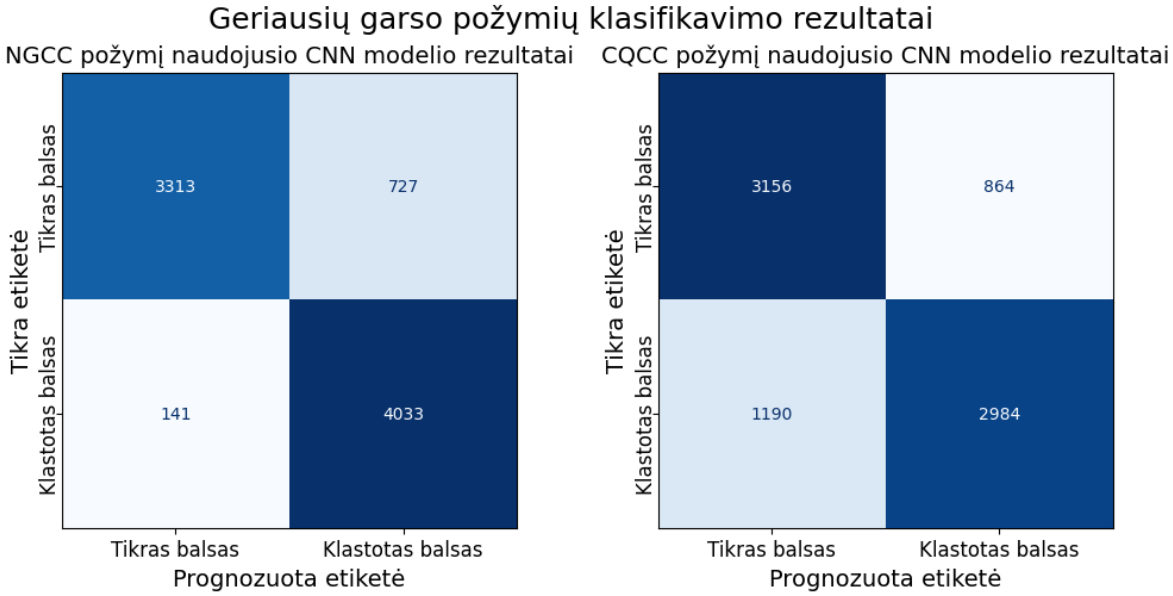
Pažvelgus į visus rezultatus (žr. **3 lentelė** ir **1 priedas**), iš vienmačių požymių pasireiškia spektrinis srautas, nuokrypis ir spektriniai centroidai. Taipogi pasireiškia neapdorota garso banga. Tą galima pamatyti ir žemiau pateiktame bendrame visų savybių vertinime (žr. **30 pav.**). Tačiau čia verta paminėti, kad ši savybė pasireiškia tik su *RawNet3* modeliu. Tas modelis yra daug sudėtingesnis nei kiti čia tirti modeliai ir dėl to turi žymiai didesnį skaičių mokymo parametrų.



30 pav. Tikslumo ir DCF vidutiniai vertinimai pagal skirtingas garso požymius

Iš dvimačių požymių akivaizdžiai pasireiškia spektrogramos (žr. **30 pav.** ir priedas 1). Ypač pasižymi šį požymį naudojęs konvoliucinis neuroninis tinklas. Tas pačias spektrogramas naudoję rekurentiniai modeliai taipogi pasiekė aukštą tikslumą. Čia verta paminėti, kad rekurentiniai modeliai turi daug

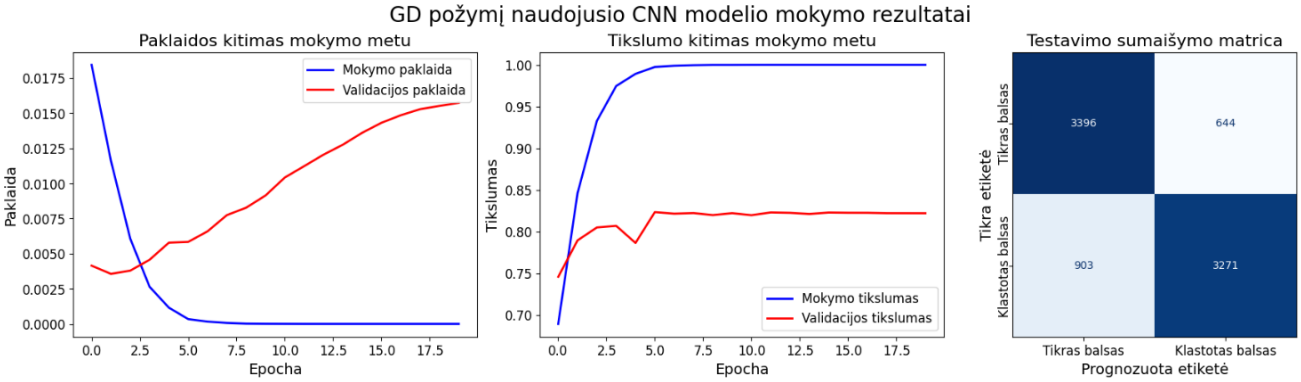
mažiau parametrų, dėl to jie yra apmokomi ir veikia greičiau nei konvoliucinius tinklus naudoję modeliai. Iš kepstrinių koeficientų *NGCC* pasiekė geriausius rezultatus (žr. **31 pav.**).



31 pav. Geriausių kepstrinių garso požymių klasifikavimo tikslumas

Atidžiau pažvelgus į šią transformaciją naudojusio modelio sumaišymo matricą, matyti, kad modelis tikrai aukštu tikslumu išskiria klastotus garso failus. Kaip matyti, modelis vos 141 kartą klastotą balsą suklasifikavo kaip tikrą, tačiau tikras balsas dažniau buvo klasifikuotas kaip tikras. Kadangi ieškoma tokių sprendimų, kurie kuo rečiau praleistų suklastotus balsus, toks modelis tiktų galutiniam sprendimui. Kaip matyti, *CQCC* transformaciją naudojusio modelio rezultatai rodo atvejį, kur atitinkamas modelis dažniau neteisingai suklasifikuoja klastotus balso įrašus. Šis modelis nebūtų tinkamas galutiniam sprendimui. Papildomai, gerus rezultatus pasiekė *PNCC*, *GFCC*, *BFCC* ir *CQT* savybės (žr. **30 pav.** ir **1 priedas**).

Fazę naudojusios savybės pasiekė prastesnius rezultatus. Geriausias fazę naudojęs modelis buvo *CNN2D*, modelis naudojęs *GD* transformaciją, tačiau tas modelis taipogi yra labai didelis savo mokymo parametrų skaičiumi (žr. 1 priedas). Pažvelgus į šio modelio mokymo rezultatus, akivaizdu, kodėl modelis veikia prasčiau (žr. **32 pav.**).

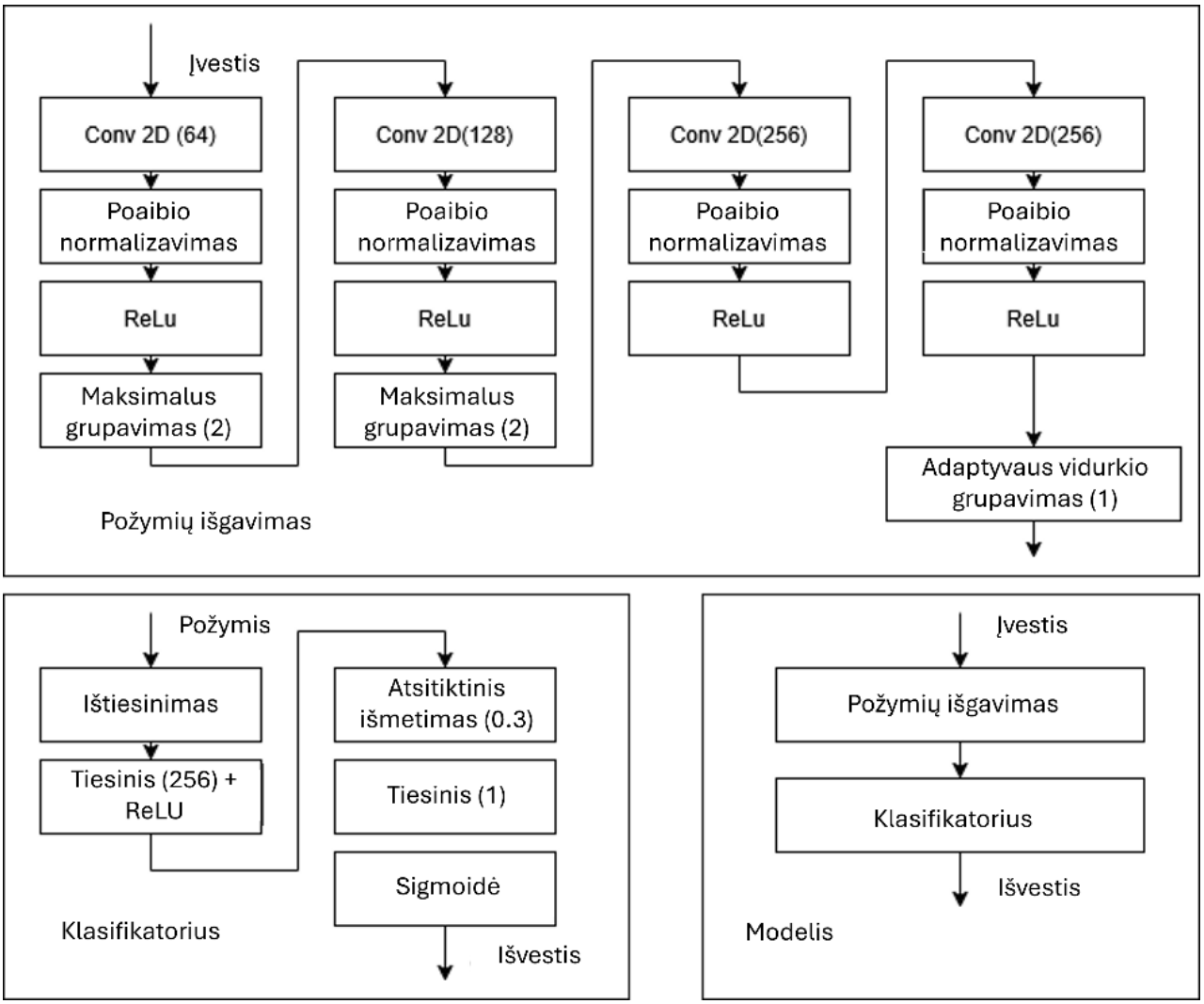


32 pav. GD požymį naudojusio CNN modelio mokymo rezultatai

Pažvelgus į tikslumo ir paklaidos kitimo kreives matoma, kad modelis labai persimokė (angl. *overfitting*). Identiška situacija matoma ir kituose faze paremtuose savybėse: *MGD*, *IF* ir kompleksinėje *CQT*. Geresniems rezultatams pasiekti, reiktų taikyti sudėtingesnes architektūras.

4.2. Galutinis konvoliucinis klasifikatorius

Kadangi po geriausių klasifikatorių paieškos atrasta, kad geriausius rezultatus pasiekia konvoliuciniai neuroniniai tinklai, galutiniam sprendimui buvo nuspręsta naudoti būtent šį klasifikavimo metodą. Praeituose eksperimentuose naudota CNN architektūra (žr. **27 pav.**) yra labai paprasta, dėl to klasifikavimo rezultatai tiems modeliams buvo prastesni. Šiam eksperimentų etapui buvo nuspręsta sukurti šiek tiek sudėtingesnę *CNN* modelių architektūrą, kuri turėtų pasiekti geresnius rezultatus. Žemiau pateikta šio modelio architektūra (žr. **33 pav.**).



33 pav. Galutinio klasifikatoriaus CNN_2MP struktūra

Šis modelis susideda iš dviejų pagrindinių komponentų: savybių išgavimo dalies ir paties klasifikatoriaus. Savybių išgavimo dalis susideda iš keturių konvoliucinių blokų. Pirmų dviejų blokų pirmas komponentas yra konvoliucinis sluoksnis, kuris naudoja tokius parametrus: branduolio dydis 3, žingsnis 1 ir kamšalas 1. Trečias ir ketvirtas blokai taipogi naudoja konvoliucinius sluoksnius, tačiau jų parametrai kitokie: branduolio dydis 4, žingsnis 2 ir kamšalas 1. Po konvoliucinių sluoksnių visi blokai naudoja poaibio normalizaciją ir *ReLu* aktyvacijos funkciją. Galiausiai, pirmi du blokai

turi maksimalaus grupavimo sluoksnius. Pirmas blokas n kanalų įvestį praplečia iki 64 savybių, tada kiekvienas konvoliucinis blokas kas kartą padvigubina savybių skaičių. Visi blokai dvigubai sumažina įvesto vaizdo dydį. Pirmi du blokai tai padaro pergrupavimo operaciją, o sekantys du per konvoliucinius sluoksnius. Galiausiai, įvykdoma adaptyvaus vidurkio grupavimo operacija (angl. *Adaptive average pooling*), kuri visas išgautas savybes sumažina iki 512x1x1 dydžio matricos, kuri pateikiama klasifikavimo daliai.

Klasifikatorius prasideda su išlyginimo operacija, kuri išgautas savybes paverčia vektoriumi. Tada tas vektorius pateikiamas pirmam pilnai sujungtam sluoksniui su *ReLu* aktyvacijos funkcija, kuris šį vektorių sumažina dvigubai. Po šio sluoksnio mokyme naudojamas atsitiktinio išmetimo sluoksnis, kuris pašalina 30% išvesto vektoriaus verčių. Sumažintas vektorius galiausiai pateikiamas paskutiniam pilnai sujungtam sluoksniui, kuris pritaikęs sigmoidės funkciją išveda galutinę prognozę.

Su šia architektūra buvo apmokyti modeliai, kurie naudoja skirtingas pirminiame testavime aukščiausius vidutinius tikslumus pasiekusias savybes. Visi modeliai mokyme naudojo galutinį duomenų rinkinį (492917 failai). Kadangi šis duomenų rinkinys susideda iš daug garso failų, mokymo, validavimo ir testavimo imčių santykiai buvo pakeisti į 80:15:5%. Kadangi visi garso failai šiame duomenų rinkinyje yra įvairaus ilgio, visi garso failai buvo praplėsti iki 1 sekundės ilgio. Failas praplečiamas taip, kad pats garsas būtų pačiame garso sekos centre, tada garso signalas iš abiejų pusių praplečiamas nuliais.

Geriausių garso požymių paieškos eksperimentų metu buvo atrasta, su kokiomis savybėmis klasifikatoriai geriausiai sugeba atskirti klastotą balsą nuo tikro. Galutinių modelių mokymui buvo naudojami visi požymiai, su kuriais klasifikatoriai sugebėjo pasiekti didesnį vidutinį tikslumą už 70%. Papildomai, dalis modelių buvo apmokyti du kartus. Vieną kartą buvo naudojamas visas pilnas galutinis duomenų rinkinys, o antrą kartą buvo naudota tik švarių garso įrašų dalis.

4.3. Galutinio konvoliucinio klasifikatoriaus rezultatai

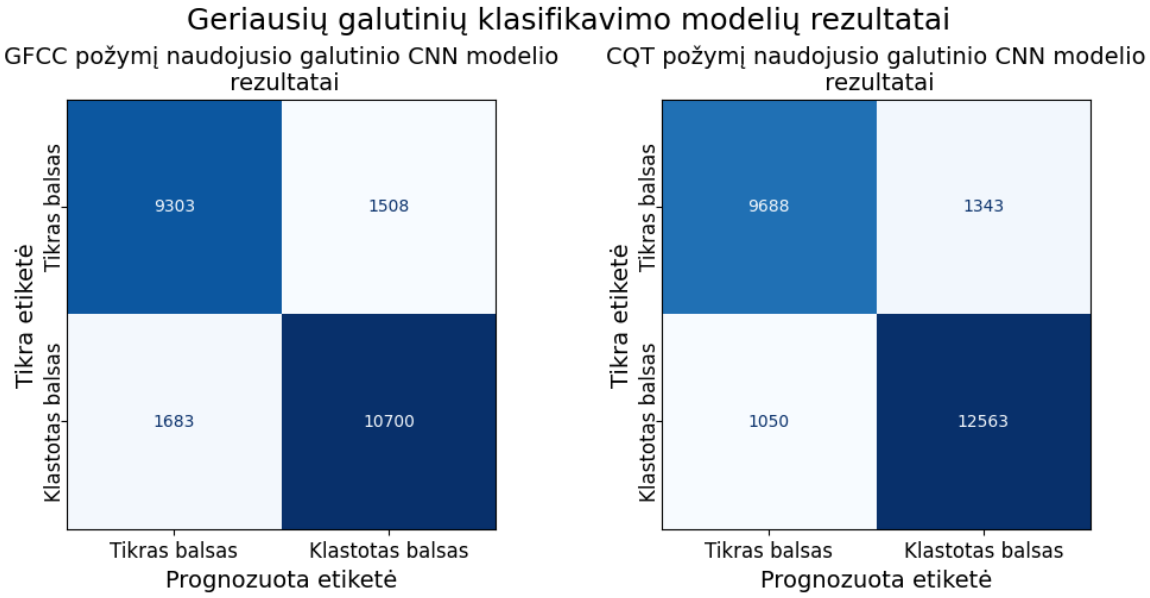
Žemiau pateikti galutinio modelio (žr. **33 pav.**) mokymo rezultatai su visomis pasirinktomis garso transformacijomis (žr. **4 lentelė**). Pastaba: siekiant taikyti vienodas sąlygas visiems modeliams buvo taikomas toks pat fiksuotas mokymo epochų skaičius.

4 lentelė. Galutinių konvoliucinių klasifikatorių testavimo rezultatai

Garso požymis	DCF	Tikslumas	Preciziškumas	Jautrumas	F1	Mokymo epochos
GFCC_adv_30	0,129514	0,909559	0,908937	0,908333	0,908623	100
Spectrogram_adv_512	0,138043	0,903514	0,90288	0,902202	0,902525	100
CQT_adv_84	0,146218	0,902824	0,900492	0,902736	0,901483	100
PNCC_adv_30	0,148466	0,897874	0,896648	0,896793	0,89672	100
MFCC_adv_30	0,152467	0,893289	0,892636	0,89184	0,892215	100
NGCC_adv_30	0,154548	0,891098	0,890679	0,889534	0,890055	100
IMFCC_adv_30	0,161158	0,888177	0,887166	0,886814	0,886986	100
Spectrogram_512	0,165267	0,880443	0,881116	0,879749	0,880146	200
BFCC_adv_30	0,173395	0,879737	0,878633	0,878289	0,878456	100
MFCC_adv_30	0,181944	0,876045	0,874147	0,874964	0,874533	100

Mel_Spectrogram_adv_512	0,187061	0,869796	0,868758	0,868178	0,868455	100
GFCC_30	0,196094	0,862421	0,8623	0,86164	0,86192	200
NGCC_30	0,204109	0,852505	0,853301	0,851888	0,852191	200
NGCC_20	0,213085	0,836854	0,839654	0,838546	0,836815	200
LFCC_adv_30	0,227286	0,844762	0,84252	0,843262	0,84287	100
PNCC_20	0,229615	0,837242	0,837457	0,836429	0,836774	200
PNCC_30	0,237807	0,843408	0,841059	0,844086	0,842081	200
Mel_Spectrogram_512	0,242015	0,825386	0,826271	0,824879	0,825063	200
CQCC_adv_30	0,254041	0,822243	0,82114	0,82011	0,820568	100
CQCC_adv_84	0,287759	0,802767	0,800158	0,800678	0,800405	100
GD_adv_512	0,29832	0,797939	0,794437	0,796093	0,795149	100
IF_adv_512	0,328682	0,782805	0,777152	0,781867	0,778748	100
MGD_adv_512	0,344307	0,762761	0,760051	0,760127	0,760089	100

Šioje lentelėje matomi visi su galutiniu klasifikatoriumi atliktų eksperimentų rezultatai. Kaip matyti, gama tono dažnio kepstriniai koeficientai (*GFCC*), spektrogramos ir *CQT* spektrogramos pasiekė geriausius rezultatus. Jų tikslumas aukštesnis už 0,9. Papildomai, galima pamatyti, kad su pilnu galutiniu duomenų rinkiniu, kur prie požymių yra prierašas *adv*, visada pasiekti geresni rezultatai, nei taikant tik švarius garso įrašus. Tai rodo, kad priešišku triukšmu paveiktų garso failų naudojimas modelio mokyme tą modelį padaro labiau atspariu priešiškai atakai. Papildomai, verta paminėti tam tikrų modelių rezultatus (žr. **34 pav.**).



34 pav. Galutinių konvoliucinių modelių testavimo rezultatai

Tikslinga priminti tai yra iki vienos sekundės ilgio garso įrašų klasifikavimo rezultatai. Nors pagal bendrus rezultatus (žr. **4 lentelė**) *GFCC* modelis pasiekė didesnį tikslumą, pagal sumaišymo matricas matyti, kad šią savybę naudojęs modelis dažniau klysta bandant klasifikuoti klastotus garso įrašus. Pažvelgus į *CQT* požymio testavimo imties sumaišymo matricą (žr. **34 pav.**), matomas priešingas atvejis. Tikri garso įrašai dažniau klasifikuojami, kaip suklastoti. Šioje situacijoje galima teigti, kad

CQT požymį naudojantis modelis yra geresnis, kadangi jis rečiau klysta klasifikuojant klastotus duomenis. Praktikoje šis modelis būtų saugesnis.

4.4. Komandinis balsavimas

Kadangi modeliai, kurie mokymo imtyje turėjo priešiška ataka paveiktus garso failus, pasiekė geresnius rezultatus, komandinio balsavimo eksperimentams naudojami tik šie modeliai. Taigi, šiame etape iš viso naudojama pirmų 14 galutinio klasifikatoriaus eksperimentuose geriausiai pasirodžiusių modelių (žr. **4 lentelė**).

Komandinio balsavimo eksperimentams įgyvendinti, pirma buvo išsaugotos visų modelių išvestys visai testavimo imčiai. Šios išvestys visur yra intervale nuo 0 iki 1 ir jos nurodo modelio tikėtinumą (angl. *confidence*), kad pateiktas garso įrašas yra klastotas. Būtent šie tikėtinumai ir yra naudojami balsavimo eksperimentuose. Susidarius paritetui balsavime, galutinį sprendimą sprendžia suminis tikėtinumas prognozės klasės viduje. Priklausant nuo eksperimente taikomo balsavimo būdo, įvertinimai sumuojami ir taip gaunama galutinė balsavimo išvestis. Šiame tyrime buvo išbandyti tokie balsavimo būdai:

- balsų dauguma;
- didžiausias užtikrintumas;
- svertinis vidurkis.

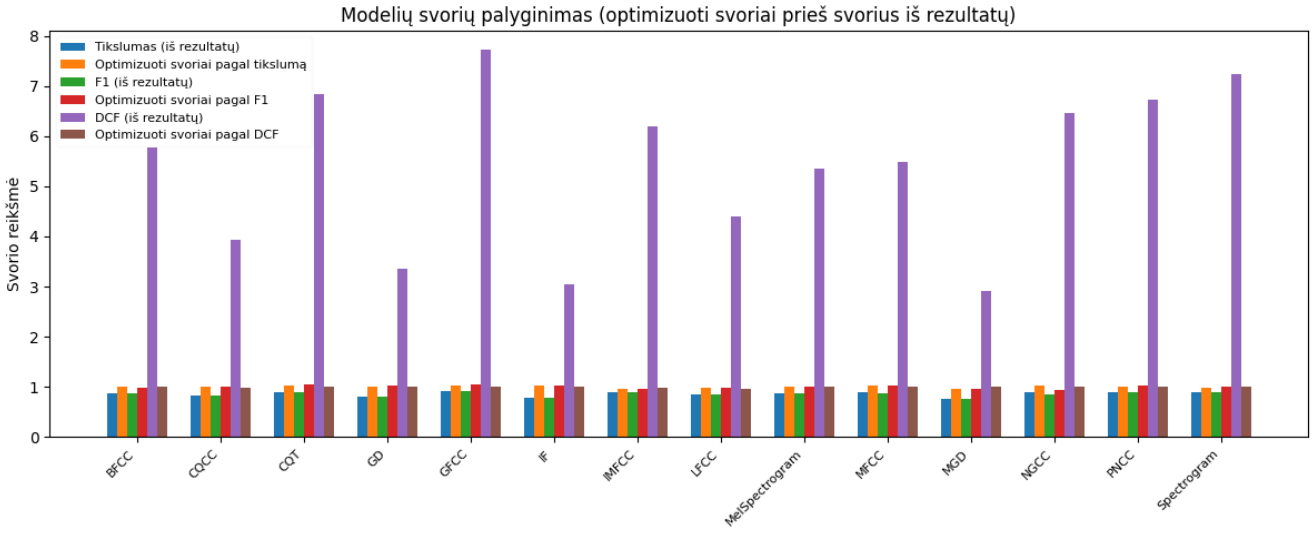
Iš šių technikų paprasčiausia yra didžiausio užtikrintumo balsavimas. Šiame darbe šis balsavimo būdas kaip galutinį rezultatą išveda didžiausią užtikrintumą turėjusio modelio išvestį. Šiek tiek sudėtingesnis balsavimo būdas yra balsų dauguma. Šis balsavimo būdas standartiškai patikrina visų atskirų modelių išvestis ir išveda tokį rezultatą, kuris visų modelių buvo išvestas dažniausiai. Šiame tyrime šis balsavimo būdas grąžina visų atskirų modelių išvesčių vidurkį. Galiausiai, buvo išbandytas svertinio vidurkio balsavimas. Tyrime šis balsavimas veikia identiškai kaip išvestis balsų dauguma, tačiau dabar kiekvienas balsas turi tam tikrą svorį, kuris yra sudauginamas su išvestu tikėtinumu prieš atliekant balsavimą. Svoriai buvo parinkti trimis būdais:

- pagal modelių tikslumą – modelio tikslumas naudojamas, kaip jo balso svoris galutinei prognozei;
- pagal modelių *DCF* įvertį – atvirkščiai proporcingas modelio *DCF* įvertis (1 / *DCF*) naudojamas, kaip jo svoris;
- pagal modelio F1 įvertį – modelio F1 įvertis naudojamas kaip jo svoris.

Papildomai, buvo pabandyta surasti optimalius svorius pagal šias tris metrikas. Svoriams surasti buvo panaudota *scipy* bibliotekos *optimize* paketo metodas *minimize*. Specifiškai buvo pritaikytas Nelderio-Meado optimizavimo algoritmas, jam buvo naudoti žemiau pateikti parametrai:

- absoliuti tarp iteracijų galima paklaida, kuri yra priimtina algoritmo sukonvergavimui (parametrai *xatol* ir *fatol*) – 0.000001;
- maksimalus iteracijų skaičius (parametras *maxiter*) – 50000.

Žemiau pateiktas visų skirtingų svorių palyginimas (žr. **35 pav.**).



35 pav. Svertinio vidurkio balsavimo eksperimentų svoriai

Kaip matyti, beveik visų svorių parinkimo rezultatai buvo labai panašūs ir svyruoja aplink vieneto reikšmę. Vienintelė išimtis tam yra atvirkščiai proporcingi *DCF* svoriai, paimti iš modelių vertinimo (žr. **4 lentelė**). Pagal šiuos svorius labai aiškiai matosi, kurie modeliai yra svarbesni, o kurie yra mažiau svarbūs. Trys svarbiausi požymiai yra *GFCC*, spektrograma ir *CQT*. Trys mažiausiai svarbūs svoriai naudojo fazinius požymius *MGD*, *IF* ir *GD*.

Taigi iš viso išbandytos aštuonios balsavimo strategijos ir keturiolika modelių, su kuriais galima atlikti balsavimą. Geriausiai kombinacijai surasti buvo išbandytos visos galimos 130952 balsavimo derinių kombinacijos.

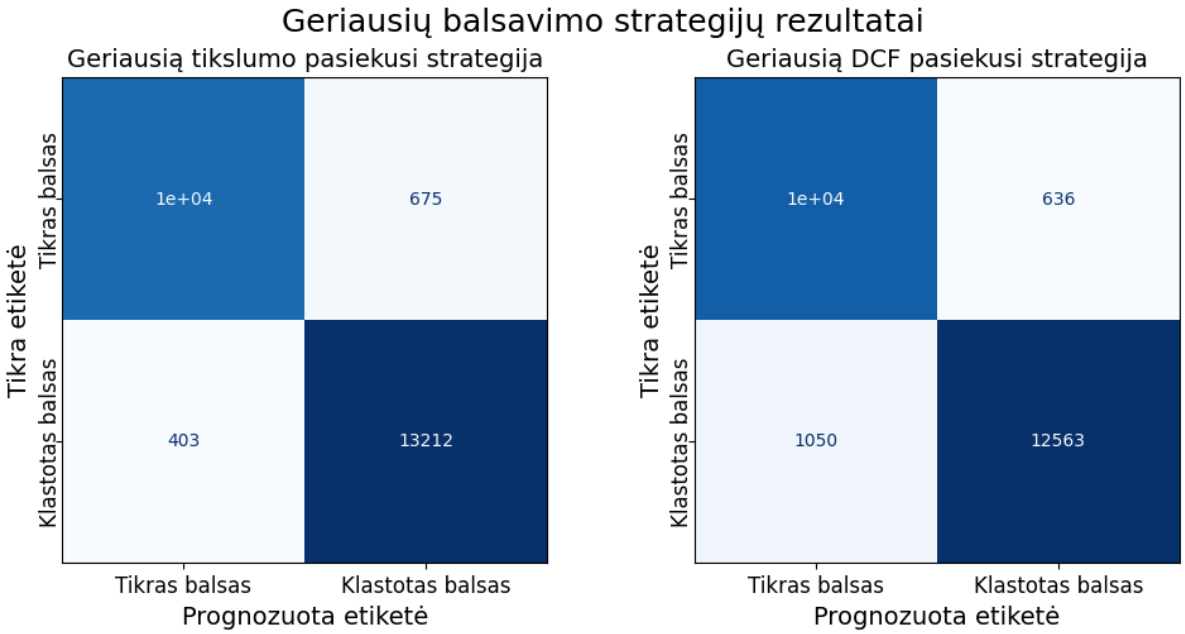
4.5. Balsavimo eksperimentų rezultatai

Žemiau pateikta skaitinių balsavimo eksperimentų rezultatų iškarpa (žr. **5 lentelė**).

5 lentelė. Geriausi, kiekvieno modelio skaičiaus, balsavimo rezultatai pagal tikslumą

Modelių skaičius	**Balsavimo režimas**	**DCF**	**EER**	**Tikslumas**
10	Svertinis vidurkis pagal DCF	0,068388	0,044148	0,956261
8	Svertinis vidurkis pagal F1	0,067812	0,046272	0,95618
9	Svertinis vidurkis pagal DCF	0,068340	0,044877	0,956017
11	Svertinis vidurkis pagal DCF	0,068380	0,044239	0,955977
7	Svertinis vidurkis pagal F1	0,068279	0,045145	0,955895
12	Svertinis vidurkis pagal DCF	0,068303	0,043967	0,955652
6	Svertinis vidurkis pagal optimizuotą F1	0,069411	0,046324	0,955530
13	Svertinis vidurkis pagal DCF	0,069293	0,044436	0,954881
5	Svertinis vidurkis pagal F1	0,070064	0,046505	0,954475
4	Svertinis vidurkis pagal tikslumą	0,070876	0,048917	0,953664
14	Svertinis vidurkis pagal DCF	0,075878	0,044436	0,952934
3	Svertinis vidurkis pagal optimizuotą tikslumą	0,081632	0,054573	0,945427
2	Svertinis vidurkis pagal optimizuotą DCF	0,090684	0,065543	0,939625

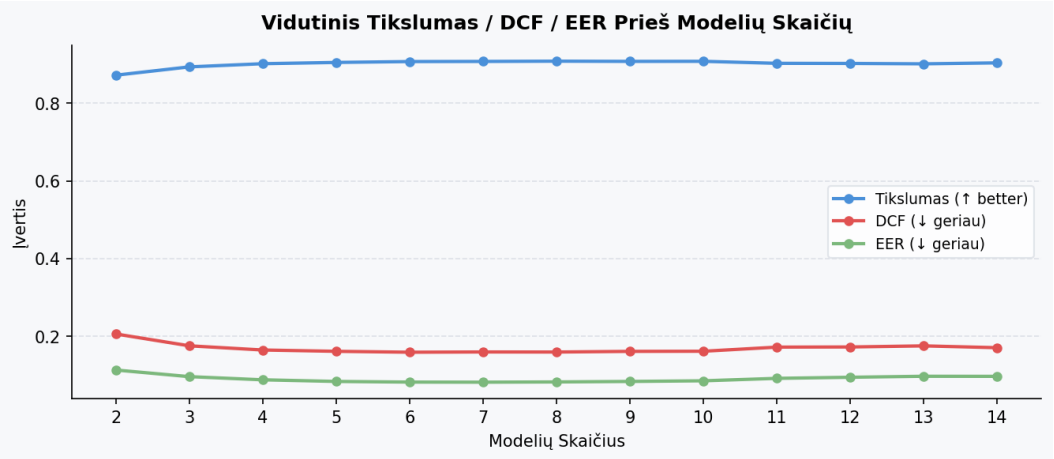
Tarp eilučių naudojami skirtingo naudotų modelių skaičių geriausi pasiekti rezultatai. Kaip matyti, pati geriausia kombinacija pasiekė 95,6% tikslumą. Jo *EER%* įvertis yra 4,4%. Didžioji dalis čia pavaizduotų kombinacijų taipogi pasiekia už 95% aukštesnį tikslumą. Žemiau pavaizduotos geriausią tikslumą ir geriausią DCF pasiekusių kombinacijų sumaišymo matricos (žr. **36 pav.**).



36 pav. Geriausių balsavimo kombinacijų rezultatai

Kaip matyti, geriausią tikslumą pasiekusi kombinacija yra šiek tiek geresnė už geriausią DCF įvertį pasiekusią. Taip yra dėl to, nes šis modelis rečiau klastoto balso įrašus suklasifikavo kaip tikrus. Geriausią DCF pasiekęs sprendimas rečiau neteisingai suklasifikavo tikro balso failus. Tai yra tikėtinas rezultatas, kadangi DCF įvertis labiau atsižvelgia į neteisingai suklasifikuotus tikrus garso įrašus.

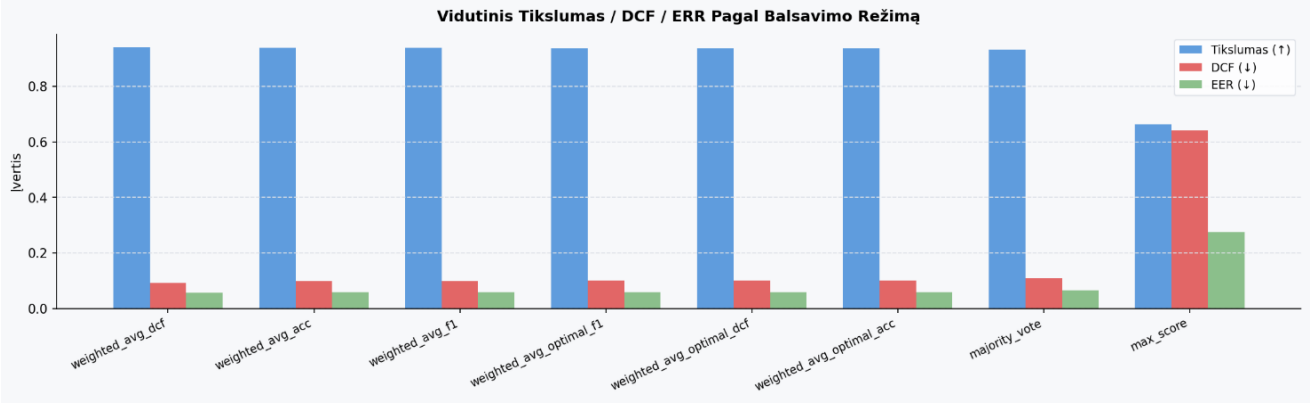
Rezultatų lentelėje (žr. **5 lentelė**) pateikti aukščiausius tikslumus turėjusios kombinacijos kiekvienam modelių skaičiui. Kaip matyti, geriausias rezultatas buvo pasiektas pasitelkus dešimt skirtingų klasifikatorių. Jeigu rezultatai būtų rikiuoti pagal *DCF* įvertį, geriausią rezultatą pasiektų aštuonis modelius naudojusi kombinacija. Žemiau pateikti vidutinių tikslumų, *DCF* ir *EER%* įverčių kitimo vertės, didėjant sprendime naudojamų modelių skaičiui (žr. **37 pav.**).



37 pav. Vidutinis tikslumo, DCF ir EER kitimas didėjant modelių skaičiui

Kaip matyti, metrikos keičiasi labai nežymiai, tačiau darbe pasiliekami aukščiausią tikslumą duodančios modelių kombinacijos. Balsavime taikant su dviem modeliais visos metrikos visada yra prasčiausios, kas yra tikėtinas rezultatas. Didinant modelių skaičių visos metrikos neženkliai gerėja iki, kol pasiekiama dešimt skirtingų modelių. Po šios ribos visi įverčiai pradeda prastėti.

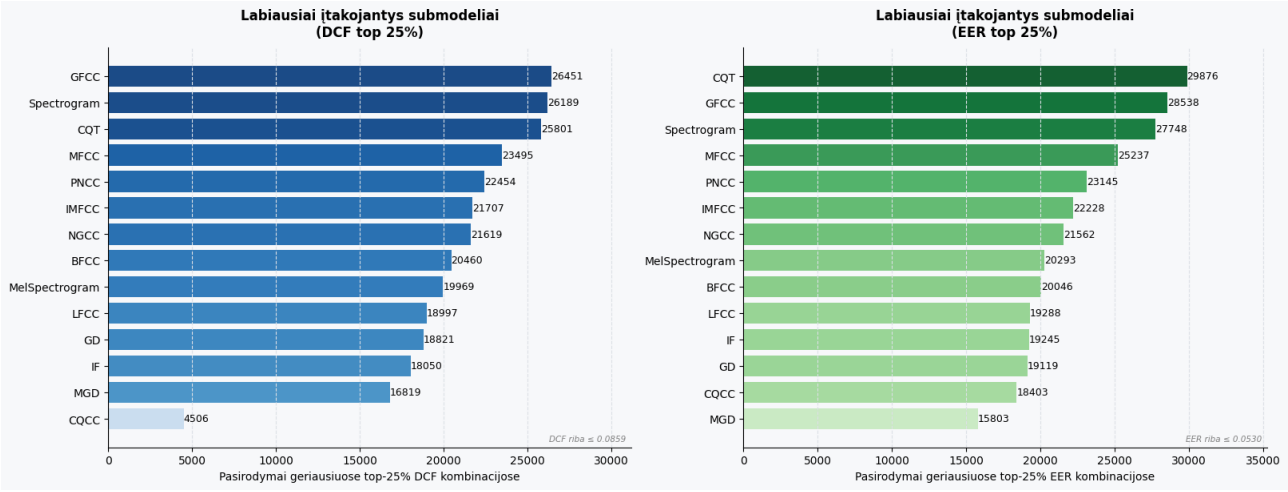
Iš balsavimo strategijų, geriausiai pasirodo svertinio vidurkio pagal *DCF* skaičiavimas. Iš geriausių pateiktų rezultatų (žr. **5 lentelė**), ši strategija taikyta dažniausiai. Skirtingų balsavimo strategijų vertinimai pateikti žemiau (žr. **38 pav.**).



38 pav. Vidutinis tikslumas, DCF ir EER per balsavimo metodus

Kaip matyti, beveik visos vertinimo metrikos visiems balsavimo metodams yra labai panašios. Žymiai išsiskyrė didžiausio tikėtinumo balsavimas. Čia paskutinei grupei akivaizdžiai matyti prastesni rezultatai. Tai buvo tikėtinas rezultatas, kadangi taikant šį balsavimą visada išvedamas aukščiausią tikėtinumą turinčio modelio rezultatas. Visų šiame tyrime apmokytų modelių išvestis yra vienas skaičius, kuris rodo užtikrintumą, kad jam pateiktas balso įrašas yra klastotė. Taigi, jei bent vienas modelis suklysta ir neteisingai išveda klastotės prognozę, šioje balsavimo strategijoje šio modelio balsas nusveria visus likusius modelius, net jei jie visi pateikė priešingą prognozę. Dar kartą įsitikinta, kad galutinė prognozė jautri pasirinktam balsavimo tipui.

Galiausiai, buvo išanalizuota, kokius požymius naudojantys modeliai geriausiuose balsavimo kombinacijose pasirodo dažniausiai. Žemiau pateikti didžiausią įtaką turėję požymiai pagal DCF ir EER įverčius (**39 pav.**).



39 pav. Didžiausią įtaką, balsavimo eksperimentams, turėjusios savybės

Kaip matyti, abiem įverčiams svarbios panašios savybės, tačiau yra keli skirtumai. Geresniam *DCF* įverčiui svarbesni gamatono dažnių kepstriniai koeficientai (*GFCC*), o *EER%* CQT išgauta spektrograma. Abiem atvejais mažiausiai svarbūs požymiai yra Q konstantų kepstriniai koeficientai (*CQCC*) ir modifikuotas grupės vėlinimas (*MGD*).

4.6. Garso klasifikavimo eksperimentų apibendrinimas

Šio skyriaus eksperimentų metu buvo bandyta atrasti optimalų (geriausią iš bandytų) klastoto balso klasifikavimo sprendimą. Pirmų eksperimentų pradžioje buvo apmokyti 102 įvairūs modeliai, kurių metu buvo patikrinta, kokios klasifikavimo modelių ir garso transformacijų kombinacijos geriausiai sugeba atskirti tikrą ir klastotą balsą. Konvoliuciniai neuroniniai tinklai pasiekė geriausius rezultatus, dėl to sekančiuose žingsniuose buvo nuspręsta naudoti šiek tiek sudėtingesnį *CNN* neuroninio tinklo modelį. Ši modelio architektūra buvo apmokyta pasitelkus pirmų eksperimentų metu atrastomis geriausius rezultatus lėmusių garso požymių. Papildomai, modeliams apmokyti buvo panaudotas garso paruošimo eksperimentų metu sukurtas ir praplėstas duomenų rinkinys. Apmokius modelius buvo pastebėta, kad modeliai, kurie apmokymo duomenų imtyje turėjo papildomų priešišku triukšmu paveiktų garso įrašų, pasiekė geresnius rezultatus, nei tie, kurie nenaudojo šių failų. Galiausiai, buvo išsirinkti 14 geriausiai pasirodę modeliai ir jų išvestys buvo panaudotos parinkti, kokia modelių kombinacija pasiekia geriausius klastoto balso klasifikavimo rezultatus. Iš visų balsavimo strategijų geriausiai pasirodė svertinį vidurkį pagal *DCF* naudojęs balsavimo sprendimas, naudojantis dešimt modelių.

5. Apjungtas klastoto balso aptikimo sprendimas

Šiame skyriuje aprašomas įgyvendintas prieš tai vykdytų eksperimentų rezultatais paremtas sprendimas ir jo testavimo rezultatai. Apjungtas sprendimas veikia pagal antrame skyriuje aprašytą logiką (žr. **10 pav.**). Sprendimas įgyvendintas *python* programavimo kalba ir susideda iš trijų pagrindinių komponentų. Visi tarpiniai rezultatai po kiekvieno komponento atlikto veiksmo yra išsaugomi.

Pirmajam komponentui buvo pasirinkta taikyti *ROF* triukšmo mažinimo algoritmą. Šiame sprendime jis veikia identiškai, kaip ir per eksperimentus, aprašytus 3.5.2 skyriuje. Verta paminėti, kad dėl greitaveikos buvo sumažintas naudojamų triukšmo mažinimo iteracijų skaičius. Dabar triukšmo mažinimas vykdomas per 50 iteracijų, po to procesas stabdomas, ir valoma spektrograma atverčiama garso signalu. Triukšmo mažinimui taikomas $\lambda = 0{,}6$ reguliarizacijos parametras, kadangi su šia verte algoritmas empiriškai pasiekė geriausius rezultatus eksperimentų metu. Galiausiai, atlikus triukšmo mažinimą, originalaus ir valyto garso failo kopijos išsaugomos *flac* formatu.

Antrajam komponentui naudojama eksperimentų metu sukurta VAD metodo modifikacija (žr. **24 pav.**). Šio metodo veikimas aprašytas 3.6.5 skyriuje ir jis apjungtame sprendime veikia identiškai. Čia verta paminėti, kad padalinti segmentai gali būti įvairaus ilgio, nuo 150 milisekundžių iki vienos sekundės. Dėl to prieš pateikiant šiuos segmentus sekančiam komponentui, visi segmentai praplečiami iki vienos sekundės, iš abiejų pusių pridedant pauzę (pridedant mažas, nuliui artimas vertes). Garso nepildome nuliais, nes tam tikros garso transformacijos negali būti atliktos, jei garso bangoje egzistuoja nuliai. Šiame žingsnyje *flac* formatu išsaugomi atskiri garso segmentai. Taip pat išsaugomas padalintos garso bangos grafikas (žr. **25 pav.**).

Trečiajam komponentui galiausiai buvo nuspręsta naudoti geriausią tikslumą pasiekusią modelių kombinaciją, kuri taikė svertinio vidurkio pagal *DCF* balsavimo strategiją. Ši kombinacija naudoja dešimt galutinių *CNN* modelių (žr. **33 pav.**), kurie kiekvienas naudoja skirtingą garso požymį. Kombinacijoje naudojami šie požymiai: *CQT*, *GD*, *GFCC*, *IF*, *IMFCC*, *LFCC*, melų skalės spektrograma, *MFCC*, *PNCC* ir paprasta spektrograma. Visi modeliai turėjo priešišku triukšmu paveiktų failų savo mokymo imtyje. Šis komponentas išveda galutinę įvesto garso įrašo klasifikavimo prognozę. Balsavimas vyksta kiekvieno atskiro segmento prognozių dauguma. Tai, jei didžioji dalis modelių išvedė, kad didesnė dalis išskirtų žodžių yra suklastoti, galutinai išvedama klastotės klasė. Papildomai, išsaugomas grafikas, kuriame atvaizduojama kiekvieno modelio išvestis kiekvienam padalintam segmentui, kiek segmentų modelis prognozavo kaip tikrus ir klastotus ir modelių balsų pasiskirstymas, naudotas galutinei prognozei gauti (pastaba: tam kad klastotė viename segmente nepakeistų viso konteksto faile klastojamam garso įraše, laikoma, kad priešiškas triukšmas tolygiai taikomas visame faile).

Toliau šiame skyriuje aprašomi šio sprendimo testavimo rezultatai. Pirma, patikrinta sukurto sprendimo greitaveika, tada patikrintas atskirų garso įrašų klasifikavimo rezultatas. Kad būtų išsiaiškinta, kas lemia vieną arba kitą prognozę, prieš tai minėti atskiri atvejai buvo ištirti su aiškinamojo dirbtinio intelekto metodu *gradCAM*. Galiausiai, sprendimo skaitiniai įverčiai palyginti su kitais jau egzistuojančiais sprendimais.

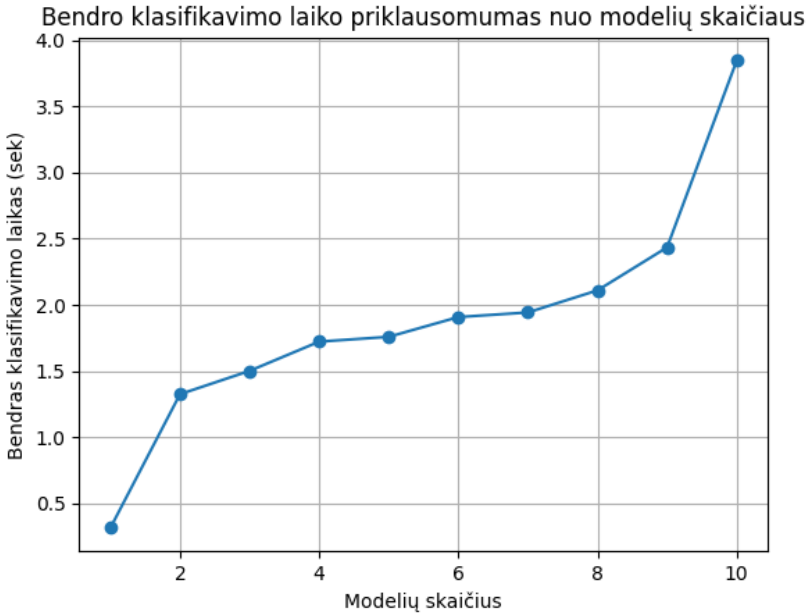
5.1. Greitaveikos testavimas

Žemiau pateikti vidutiniai kiekvieno apjungto sprendimo atskirų operacijų vidutiniai vykdymo laikai (žr. **6 lentelė**). Greitaveika tikrinta įvykdžius keturis šimtus, penkiolikos sekundžių ilgio garso failų, klasifikavimų.

6 lentelė. Apjungto sprendimo greitaveikos įvertinimai

Operacija	**Vidutinis operacijos atlikimo laikas (sekundėmis)**
Triukšmo mažinimas	1,790855
Garso failo padalijimas išskiriant žodžius	0,461177
Savybės išgavimo laikas vienam modeliui	0,010954
Vieno segmento klasifikavimas vienu modeliu	0,008691
Galutinės prognozės išgavimas balsavimo būdu	0,000093

Kaip matyti, daugiausiai viso klasifikavimo laiko užima triukšmo mažinimo operacija. Antra ilgiausia operacija yra garso failo dalijimas. Čia daugiausiai laiko užima atskirų garso failų įrašymas. Verta paminėti, kad savybių išgavimo ir vieno modelio klasifikavimo laikas yra palyginus trumpas, tačiau čia jis nurodo tik vieno modelio, vieno garso segmento apdorojimo vidutinį laiką. Realiai visas klasifikavimo žingsnis užima apie dešimt kartų ilgiau, kadangi naudojama dešimt modelių (praktikoje pakaktų naudoti 3-4 modelius neprarandant daug tikslumo). Papildomai, klasifikavimo laiką padidina garso įraše girdimų žodžių skaičius, kadangi kiekvienam žodžiui arba mažų žodžių grupei daroma atskira klasifikavimo operacija. Galiausiai, galutinės prognozės išgavimas balsavimo būdu užima labai trumpą laiką, kadangi tame žingsnyje apskaičiuojamas visų išvesčių vidurkis. Žemiau pateiktas bendro klasifikavimo laiko priklausomumas nuo modelių skaičiaus (žr. **40 pav.**). Laiko grafikas tik dar kartą patvirtina, kad jungtinis klasifikatorius yra adityvus modelis, tai yra, modelių trukmės tiesiog tiesiškai susideda.



40 pav. Apjungto sprendimo klasifikavimo laiko priklausomumas nuo modelių skaičiaus

Kaip matyti, labai aiškiai matosi klasifikavimo laiko augimo tendencija, didėjant klasifikavimui naudojamų modelių skaičiui. Tai yra tikėtinas rezultatas, tačiau čia verta paminėti, kad atskiri

modeliai turi skirtingas greitaveikas, priklausant nuo jų pateikiamų garso savybių dydžio. Tas pats galioja ir tų savybių išgavimui, kuris jau buvo ištirtas klasifikavimo eksperimentų metu (žr. **29 pav.**).

5.2. Atskiri testai

Šiame skyriuje patikrinta, kokius rezultatus išveda apjungtas sprendimas, jam pateikus įvairaus tipo garso įrašus. Žemiau pateikti visi vykdyti individualių testų scenarijai (žr. **7 lentelė**), buvo taikoma atsitiktinė atranka tarp tikrų ir klastotų balso įrašų.

7 lentelė. Apjungto sprendimo individualių testų rezultatas

Scenarijus	Testo scenarijus	Tikro įrašo balsai	Klastoto įrašo balsai	Galutinė prognozė
1	Tikras balsas su triukšmo mažinimu	9	7	Tikras balas
2	Tikras balsas be triukšmo mažinimo	15	1	Tikras balsas
3	Klastotas balsas su triukšmo mažinimu	1	2	Klastotas balsas
4	Klastotas balsas be triukšmo mažinimo	0	3	Klastotas balsas
5	Triukšmingas tikras balsas su triukšmo mažinimu	2	0	Tikras balsas
6	Triukšmingas tikras balsas be triukšmo mažinimo	2	0	Tikras balsas
7	Triukšmingas klastotas balsas su triukšmo mažinimu	1	3	Klastotas balsas
8	Triukšmingas klastotas balsas be triukšmo mažinimo	3	1	Tikras balsas

Kaip matyti, buvo išbandytos visos apibendrintam sprendimui įmanomos tipai. Iš pirmo scenarijaus testo greitai pasimato, kad pritaikius triukšmo mažinimo operaciją, didesnė dalis garso segmentų buvo suklasifikuoti, kaip klastotas balsas (ketvirtas stulpelis). Galutinė šio testo prognozė vis tiek yra teisinga, tačiau rezultatas nėra toks užtikrintas, kokio norėtųsi. Jei klasifikuojant išjungiama triukšmo mažinimo operacija, kaip matyti antrame scenarijuje, galutinis rezultatas iš karto pagerėja. Penkiolika iš šešiolikos segmentų šiame teste buvo suklasifikuoti, kaip tikras balsas, dėl to čia daug užtikrinčiau galima pasakyti, kad teste pateiktas garsas tikrai sudarytas iš tikro balso. Abiejų šių testų detalūs rezultatai gali būti rasti prieduose (**žr. priedas 2 ir 3**). Į juos pažvelgus, labai aiškiai matyti tai, kad vykdant triukšmo mažinimo operaciją, tam tikri modeliai pradeda žymiai klysti. Šie modeliai naudoja šias garso savybes: GD, IF, LFCC, MFCC ir spektrogramas. Akivaizdu, kad šie modeliai greičiausiai negali susidoroti su scenarijumi, kai apjungtam sprendimui pateikiamas tikras netriukšmingas garso įrašas. Netriukšmingam garsui atliekant triukšmo mažinimo operaciją, garso signalas yra pakankamai pamodifikuojamas, kad klasifikavimo modeliai juos pradeda identifikuoti kaip klastotus. Tai nėra visiškai blogas rezultatas, kadangi jis parodo, kad modeliai išmoko aptikti manipuliuotus failus, kas ištiktųjų yra svarbiau nei gerus failus supainioti su klastotėmis.

Panaši situacija matoma ir klastoto balso klasifikavimo testų scenarijuose (3 ir 4 scenarijai). Taikant triukšmo mažinimą, vienas segmentas (dažniausiai labai trumpų žodžių rinkinys) bendrai aptinkamas kaip tikro balso sakomas žodis. Netaikant triukšmo mažinimo, tas žodis (žodžių rinkinys) aptinkamas

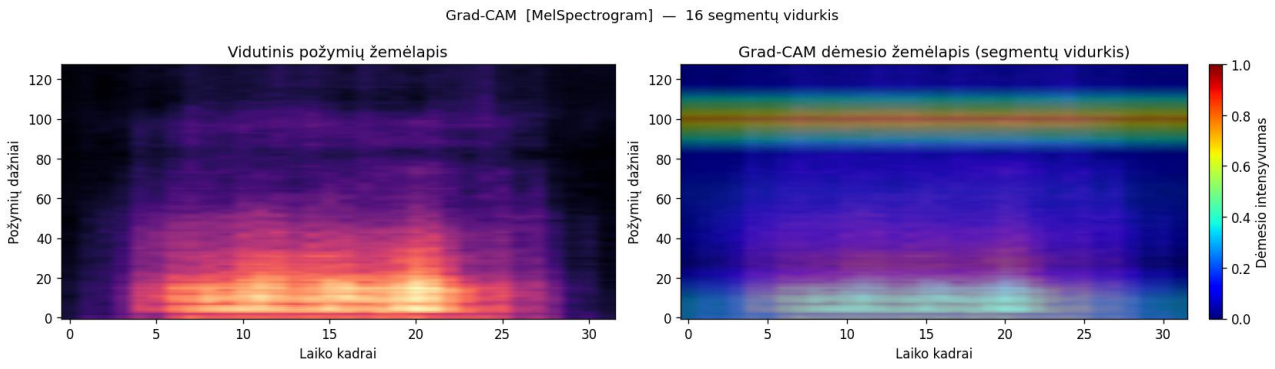
kaip klastotė. Detalesnis šių testų apibendrinimas pateiktas prieduose (žr. **4 ir 5 priedas**). Ten matyti, kad vykdant triukšmo mažinimą, trečiasis išskirtas garso segmentas didesnės dalies klasifikatorių yra suklasifikuojamas kaip tikras balsas. Scenarijuje, kur nebuvo taikytas triukšmo mažinimas, beveik visi segmentai daugelyje modelių labai užtikrintai suklasifikuoti kaip klastotės. Verta paminėti, kad ši neteisingos prognozės dėl švaraus failo valymo (triukšmo mažinimo) problemos pasireiškia tik tam tikruose modeliuose. Tikėtiniausia to priežastis yra tai, kad šie modeliai geriau išmoko atskirti, kada garso failas buvo pamodifikuotas. Kadangi šių modelių mokymo imtyje nebuvo valytų jau švaraus balso pavyzdžių, modeliai nesugeba atskirti, kad failas iš tiesų yra geras.

Sekantys keturi testavimo scenarijai (5-6 scenarijai) yra taikomi triukšmingiems garso failams. Kaip matyti, abiejose triukšmingo tikro balso testuose apibendrintas sprendimas susidorojo su jam pateiktu iššūkiu. Ir taikant, ir netaikant triukšmo mažinimo, apibendrintas sprendimas prognozavo, kad visi garso įraše girdimi žodžiai yra tikri. Tačiau verta paminėti, kad netaikant triukšmo mažinimo operacijos pasitaikė dvi klaidos (du segmentai dviejų modelių buvo suklasifikuoti, kaip klastoti). Scenarijuje, kur triukšmo mažinimas buvo įjungtas, pasitaikė tik viena klaida, kuri turėjo mažesnį užtikrintumą nei abi prieš tai minėtos klaidos. Tai rodo, kad parinkta triukšmo mažinimo strategija iš tiesų yra veiksminga, bandant šalinti triukšmą. Tai taip pat parodo, kad triukšmingų duomenų pridėjimas į modelių mokymo imtį padėjo tiems modeliams išmokti atskirti bent tikrus triukšmingus duomenis.

Triukšmo mažinimo komponento vertė dar geriau išryškėja paskutiniuose dviejuose testavimo scenarijuose (7-8 scenarijai), kur bandoma suklasifikuoti triukšmingą klastotą garso failą. Kaip matyti rezultatų lentelėje (žr. **7 lentelė**), netaikant triukšmo mažinimo, trys atskiri garso segmentai buvo suklasifikuoti kaip tikri. Tame testo scenarijuje galutinai išvesta neteisinga prognozė, kas parodo, kad modeliai prasčiau susidoroja su triukšmingais klastotais duomenimis. Įjungus triukšmo mažinimo komponentą, šis garso įrašas atstatomas pakankamai, kad modeliai galiausiai išvestų teisingą prognozę. Abiejų šių testavimo scenarijų detalesni rezultatai pateikti prieduose (žr. **6 ir 7 priedas**).

5.3. Klasifikatoriaus garso požymiai išskiriami taikant aiškinamąjį dirbtinį intelektą

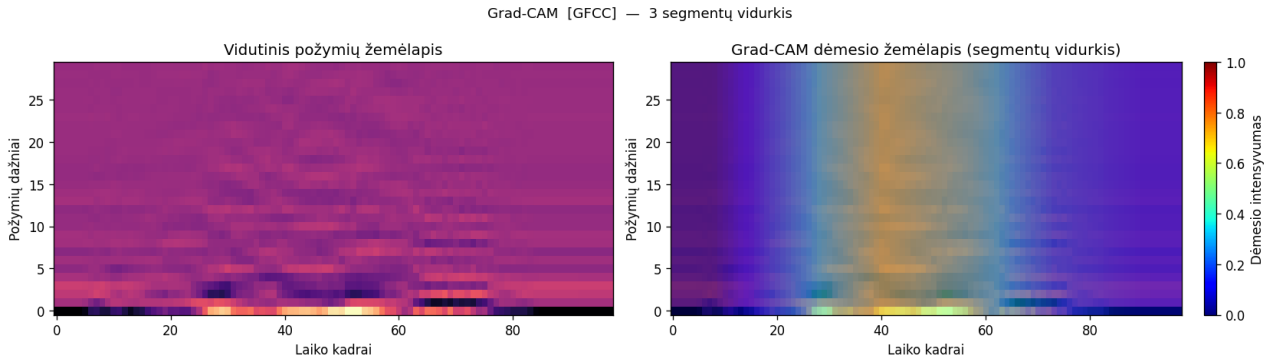
Kad dar geriau pasimatytų, kodėl modeliai išveda vienokias ar kitokias prognozes, panaudotas aiškinamojo dirbtinio intelekto (angl. *explainable artificial intelligence* arba *XAI*) metodas *gradCAM*. Šis metodas parodo, kokie pateikti pradinio garso požymiai turėjo didžiausią įtaką jungtinio klasifikatoriaus prognozei. Žemiau pateiktas melų spektrogramos dėmesio žemėlapis tikrame garso įraše (žr. **41 pav.**).



41 pav. Melų skalės spektrogramos XAI paaiškinimas tikrame garso įraše

Šiuose paveikslėliuose matomi visų klasifikuotų segmentų reprezentacijų vidurkiai. Dešinėje matomas paveikslėlis rodo, kuri spektrogramos dalis buvo svarbiausia norint išgauti tikro garso įrašo prognozę. Kaip matyti, melų spektrogramos modelis fokusavosi į artimo šimtui melų skalės intervalą. Šis intervalas atspindi aukštą dažnį, dėl to matyti, kad modelis fokusavosi į aukštesnį dažnį bandydamas atrasti, ar garso įrašas yra klastotas, ar ne. Tai yra tikėtinas rezultatas, kadangi dirbtiniu intelektu generuoti garso įrašai dažniausiai turi generatorių paliktus artefaktus aukštesniuose garso dažniuose, kurie standartiškai leidžia atskirti, ar garso įrašas yra generuotas. Kadangi šis modelis tokių artefaktų nerado, modelis galiausiai išvedė tikro balso prognozę. Daugiau melų spektrogramos dėmesių žemėlapių galima rasti prieduose (žr. **8 priedas**). Kaip matyti, ir kituose garso failuose modelis susikoncentruoja į vieną specifinį melų skalės intervalo dažnį. Dažniausiai šie dažniai yra aukšti.

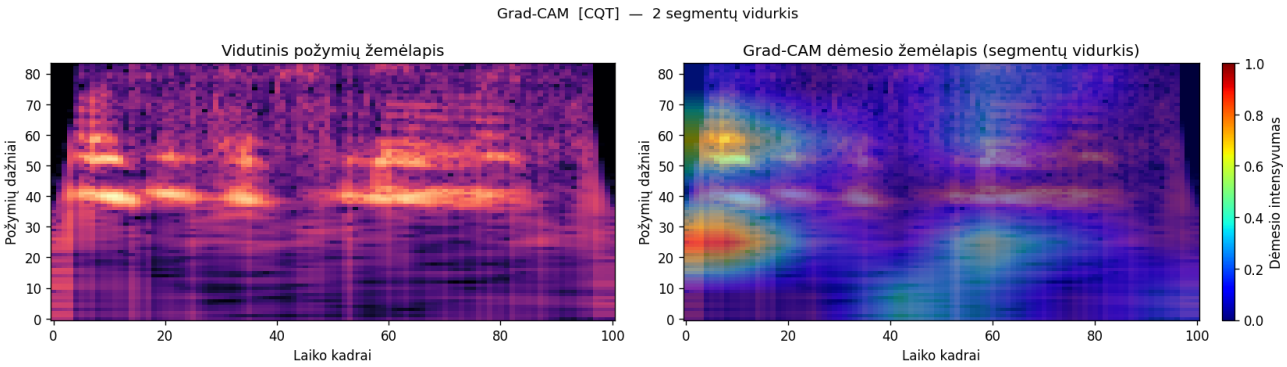
Žemiau pavaizduotas gamatono dažnio kepstrinių koeficientų atidumo žemėlapis (žr. **42 pav.**)



42 pav. GFCC XAI paaiškinimas klastotame garso įraše

Čia matyti, kad modelis fokusuojasi į visą dažnių spektrą specifiniame laiko intervale. Iš paties požymių žemėlapio kairėje matyti, kad tame laiko intervale girdimas įraše tariamas žodis (intervale ~25-75 laiko kadruose). Verta paminėti, kad dėmesio žemėlapis yra ryškiausias aplink keturiasdešimtą laiko kadrą, taigi tuo laiko momentu modelis aptinka kažkokį artefaktą, kuris nulemia galutinę modelio prognozę – klastotė. Daugiau *GFCC* dėmesio žemėlapių pavyzdžių galima rasti prieduose (žr. **9 priedas**). Kaip matyti ir kituose garso įrašuose, modelis yra susifokusavo į specifinį laiko intervalą, pagal kurį modelis išveda galutinę savo prognozę garso įrašo segmentams.

Paskutinis specifinis scenarijus, kuris analizuojamas šiame poskyryje, yra Q konstantų transformacijos sukurta spektrograma, jos dėmesio žemėlapiai matomi žemiau (žr. **43 pav.**).



43 pav. CQT XAI paaiškinimas išvalytame tikrame garso įraše

Čia matyti, kad modelis koncentravosi į vieno specifinio dažnio arba laiko momentą, galutinei prognozei išvesti. Čia modelis susifokusavo į dvi specifines vietas spektrogramos pradžioje, laiko atžvilgiu ties dviem specifiniais dažnių intervalais – apie trisdešimt ir šešiasdešimt. Modelis specifinėse spektrogramos vietose bandė atrasti, kokius nors dirbtinio intelekto metodų sukurtus artefaktus, tačiau, kadangi jie nebuvo rasti, modelis galiausiai išvedė tikro garso įrašo prognozę. Panaši situacija matoma ir kituose šios savybės dėmesio žemėlapių pavyzdžiuose, kuriuos galima rasti prieduose (žr. **10 priedas**). Modelis prognozę išveda pagal specifiniame regione esančią informaciją. Ši informacija dažniausiai randama spektrogramos viduriniuose dažniuose.

Galiausiai, prieduose (žr. **11 priedas**) galima rasti visų modelių dėmesio žemėlapius vienam garso įrašui. Iš čia matyti, kad *LFCC* ir *PNCC* modeliai, kaip ir *GFCC*, koncentruojasi į specifinius laiko intervalus. Matyti, kad *LFCC* dėmesys išskiriamas panašus į *GFCC*, tačiau šiek tiek platesniame laiko intervale, o *PNCC* „stebi“ visiškai skirtingą laiko intervalą. *GD*, *IF* ir paprasta spektrograma panašiai kaip melų skalės spektrograma, labiau fokusuojasi į specifinius dažnius. Tačiau čia matyti, kad šios trys savybės atidžiau „stebi“ po tris skirtingus dažnius. Galiausiai, iš priede pateiktų *MFCC* ir *IMFCC* vaizdų matyti, kad abu modeliai koncentruojasi į beveik visą matomą vaizdą, išskyrus pačius žemiausius dažnius.

5.4. Palyginimas su egzistuojančiais sprendimais

Žemiau pateiktas apjungto sprendimo (jungtinio klasifikatoriaus) vertinimas lyginant su jau egzistuojančiais sprendimais (žr. **8 lentelė**). Verta paminėti, kad modeliai vertinami atviros būklės (angl. *open condition*) *ASVspoof* scenarijuje, kuri geriausiai atspindi, kaip buvo sukurtas šiame tyrime pasiūlytas sprendimas, kadangi modeliams mokyti buvo naudotas praplėstas duomenų rinkinys.

8 lentelė. Apjungto sprendimo įvertinimas lyginant su kitais egzistuojančiais sprendimais

Eil. Nr.	Sprendimas	DCF	EER%
1	Apjungtas sprendimas be triukšmo mažinimo komponento	**0,0684**	4,42%
2	Apjungtas sprendimas su triukšmo mažinimo komponentu	0,1451	6,92%
3	Bazinis RawNet3	0,2116	15,891%
4	Bazinis RawNet3 su triukšmingais duomenimis	1,0409	72,129%
5	Bazinis RawNet3 su valytais duomenimis	0,5399	32,92%
6	ASVspoof5 bazinis RawNet2 [5]	0,8266	36,04%
7	ASVspoof5 bazinis AASIST [5]	0,7106	29,12%
8	T45 [5, 57] MFA-Res2Net + WavLM	**0,0750**	2,59%
9	T43 [5, 89] AASIST	0,1149	4,04%
10	T13 [5, 53] AASIST	0,1301	4,50%
11	T06 [5, 71] WavLM transformeris	0,1348	5,02%
12	T31 [5, 70], WavLM transformeris	0,1499	5,56%
13	AASIST3 [5, 76]	0,1414	4,89%
14	T33 [5, 90], WavLM-ResNET18	0,2021	7,01%

Pirmose dviejose eilutėse matomi šiame darbe pasiūlyto sprendimo vertinimai su išjungtu ir įjungtu triukšmo mažinimo komponentu. Kaip matyti, taikant triukšmo mažinimo komponentą ir *DCF*, ir *EER%* padidėja. Šį pokytį paaiškina tai, kad klasifikatoriai aptinka triukšmo valymo metodu

pamodifikuotus garso signalus ir juos pradeda traktuoti kaip klastotus. Tai detaliau buvo paaiškinta atskirų testų poskyryje. Kadangi triukšmo mažinimo metodas galiausiai sumažino sprendimo galutinius įverčius, šis komponentas galiausiai buvo išjungtas. Be jo galutinis apjungtas sprendimas pasiekia geresnius rezultatus. Kaip matyti, buvo pasiektas 0,0683 *DCF* ir 4,4148% *EER%* klaidos įvertis.

Sekančios trys eilutės skirtos šiame tyrime įvairiuose eksperimentuose naudoto bazinio *RawNet3* įverčiams atvaizduoti įvairiose situacijose. Iš šių vertinimų verta atkreipti dėmesį į įverčius, gautus naudojant priešiška ataka paveiktus duomenis. Kaip matyti ketvirtoje eilutėje, modelio DCF padidėjo apie keturis kartus, o daromų klaidų procentas išaugo apie penkis kartus. Tai parodo priešiškos atakos efektyvumą, automatinių generuotų kalbėtojų sistemų apėjimo srityje. Modelio rezultatai žymiai pagerėja triukšmingiems duomenims atliekant triukšmo mažinimą su *ROF* algoritmu. Tačiau vis tiek matyti, kad abi metrikos yra apie du kartus prastesnės, lyginant su originaliais šio modelio rezultatais (3 eilutė).

Paskutinės devynios eilutės priklauso egzistuojantiems sprendimams, kuriuos galima rasti *ASVspoof5* [5] iššūkio rezultatuose. Šeštoje ir Septintoje eilutėse pavaizduoti to iššūkio naudotų bazinių modelių rezultatai. Iššūkyje naudoti senesni *RawNet2* ir *AASIST* modeliai. Čia verta paminėti, kad šiam tyrimui naudotas bazinis *RawNet3* modelis pasiekė apie du kartus geresnius įverčius už abu iššūkio bazinius modelius. Paskutinės septynios eilutės rodo, kelių iššūkio dalyvių pateiktų sprendimų pasiektus rezultatus. Aštuntoje eilutėje matomi geriausius rezultatus iššūkyje pasiekusio sprendimo įverčiai. Lyginant su šiame tyrime pasiūlytu sprendimu, jis turi apie du kartus mažesnį *EER%* ir šiek tiek didesnį *DCF* įvertį. Tai rodo, kad *T45* komandos pasiūlytas *Res2NET* konvoliucinių neuroninių tinklų ir *WavLM* transformerių junginys pasiekė prastesnius rezultatus, nei šiame tyrime pasiūlytas apjungtas sprendimas. Norint apskaičiuoti *EER%* įvertį apjungtai dešimties modelių grupei, reikia žymiai pakeisti slenkstinę vertę *EER%* metrikoje sigmoido klasifikavimo prognozei išgauti, kas žymiai padidina bendro sprendimo daromų klaidų procentą, lyginant su *T45* sprendimu. Jeigu pagal šį įvertį būtų vertintas pasiūlytas sprendimas, jis *ASVspoof5* iššūkio rezultatų lentelėje užimtų šeštą vietą. Modelius vertinant pagal *DCF*, kaip tai buvo daryta iššūkyje, šiame tyrime pasiūlytas sprendimas būtų pats geriausias. Aišku, čia verta paminėti, kad *DCF* vertė smarkiau auga, kai klasifikavimo sprendimas dažniau neteisingai suklasifikuoja tikrus balsus. Jeigu modelis beveik niekada neklysta klasifikuojant klastotus garso įrašus, tačiau daro daugiau klaidų klasifikuojant tikrus balsus, to modelio *DCF* bus didesnis nei modelio, kuris daro po lygiai klaidų abiejose klasėse. Tokia situacija matoma ir apjungto sprendimo rezultatuose (žr. **36 pav.**). Balsavimo eksperimentų rezultatų apžvalgos metu buvo atrasta, kad galiausiai pasirinktas sprendimas apie pusę karto dažniau daro klaidas būtent klasifikuojant tikro balso failus. Tikėtina, kad panaši situacija egzistuoja ir visuose *ASVspoof* iššūkyje pateiktuose sprendimuose. To tiksliai negalima patikrinti, nes nei iššūkio organizatoriai, nei pasiūlytų sprendimų autoriai nepateikė egzistuojančių sprendimų klaidų pasiskirstymo.

Galiausiai, šiame poskyryje verta paminėti, kad visi rezultatų lentelėje matomi (žr. **8 lentelė**) sprendimai naudoja daug sudėtingesnius klasifikavimo modelius nei šiame tyrime pasiūlytas sprendimas. Aštuntoje ir keturioliktoje eilutėse matomi sprendimai naudojo daug sudėtingesnius konvoliucinius neuroninius tinklus. Devintoje, dešimtoje ir tryliktoje eilutėse pasiūlyti sprendimai naudoja įvairias AASIST modelio modifikacijas. Galiausiai, vienuoliktoje ir dvyliktoje eilutėse matomi sprendimai naudojo transformerių tipo modelius. Visi šie modeliai naudoja daug sudėtingesnes architektūras ir jiems apmokyti buvo naudoti didesni duomenų rinkiniai nei šiame

tyrime pasiūlyto sprendimo. Papildomai, dalis šių sprendimų taipogi naudoja komandinį balsavimą, tačiau ten vėlgi taikomos sudėtingesnės strategijos su daugiau modelių, nei buvo taikyta šiame tyrime. Blogiausiu atveju pasiūlytas sprendimas yra šeštas geriausias, o geriausiu atveju – pats geriausias, iš 34 *ASVspoof5* konkurse pasiūlytų sprendimų. Iš to galima išvesti išvadą, kad pritaikius daug paprastų modelių galima pasiekti geresnius rezultatus, nei taikant pavienius sudėtingus modelius.

5.5. Apjungto sprendimo apibendrinimas

Šiame skyriuje buvo aprašytas pasirinktas apjungtas klastoto balso klasifikavimo sprendimas. Jis atitinka darbo pradžioje aprašytą logiką (žr. **10 pav.**). Sprendimas susideda iš trijų komponentų – garso valymo, žodžių išskyrimo ir klasifikavimo (komandiškai balsavus) taikant ne vieną modelį. Patikrinus greitaveiką, buvo pastebėta, kad lėčiausias komponentas yra triukšmo mažinimas, kuris papildomai pakankamai pamodifikuoja netriukšmingus failus, kad sekančiame žingsnyje veikiantys klasifikatoriai tuos failus pradeda klasifikuoti kaip manipuliuotus. Šiai problemai išspręsti reiktų apmokyti naujus klasifikavimo modelius su dar daugiau praplėstu duomenų rinkiniu, kuriame būtų galima rasti valytų netriukšmingų garso įrašų. Atlikus bandymus su triukšmingais duomenimis nustatyta, kad triukšmo mažinimo komponentas gali būti svarbus, kadangi be triukšmo mažinimo priešiškos atakos paveikti garso failai kai kuriais atvejais sugebėdavo apgauti apjungtą sprendimą. Tačiau verta paminėti, kad tam tikrose situacijose modeliai gali teisingai atskirti triukšmo paveiktus garso failus. Empiriškai įvertinus triukšmo komponento mažinimo komponento įtaką galutiniam sprendimui, buvo atrasta, kad modelio tikslumas sumažėjo, lyginant su triukšmo mažinimo komponento nenaudojančiu sprendimu. Dėl to galiausiai buvo nuspręsta naudoti sprendimą su išjungtu triukšmo mažinimu. Tikėtina, praktikoje vertėtų triukšmo mažinimo stadijos atsisakyti.

Pritaikius aiškinamojo dirbtinio intelekto metodą *gradCAM*, atrasta, kad skirtingi modeliai fokusuojasi į skirtingas jiems pateiktų garso įrašo dalis prognozei pasiekti. Galiausiai, įgyvendintas sprendimas palygintas su jau egzistuojančiais sprendimais. Čia atrasta, kad šiame darbe pasiūlytas sprendimas yra geresnis už visus *ASVspoof5* iššūkyje pasiūlytus sprendimus, atsižvelgiant į tame iššūkyje modeliams vertinti naudojamą *DCF* metriką.

Išvados

1. Atlikus klastoto balso aptikimo literatūrinę apžvalgą paaiškėjo, kad pagrindinė generuoto balso naudojimo sritis yra kenkėjiška ataka. Garso generavimo metodai yra lengvai prieinami, kas dar labiau pabrėžia klastotos kalbos aptikimo uždavinio svarbą. Šiais metodais sugeneruoti garso failai dažnai pasižymi nenatūralumais aukštų dažnių srityse. Dėl to garso transformacijos, kurios išryškina šią sritį, leidžia efektyviau išskirti generuotus garsus. Vienas didžiausių iškeliamų iššūkių, su kuriuo susiduria dabartinės klastotos kalbos aptikimo sistemos, yra priešiška ataka. Iš egzistuojančių klastotos kalbos aptikimo sprendimų, dažniausiai naudojami konvoliucinių neuroniniai tinklai ir transformeriai.
2. Atsižvelgiant į literatūrinės apžvalgos rezultatus, suformuotas balso klasifikavimo algoritmas, kuris susideda iš trijų principinių dalių – įvesto garso valymo, padalijimo išskiriant žodžius ir klasifikavimo daugelio modelių komandinio balsavimo būdu.
3. Atlikus garso paruošimo eksperimentus, buvo įsitikinta, kad egzistuojantys sprendimai žymiai prasčiau aptinka, naujais garso generavimo metodais sintezuotus ir priešišku triukšmu paveiktus garsus. Duomenų rinkiniui pritaikius FGSM metodą, eksperimentuose naudoto modelio tikslumas nukrito per ~57%. Iš tirtų triukšmo mažinimo metodų geriausiai pasirodė triukšmo mažinimo pagal bendrąją variaciją algoritmas ROF. Galiausiai, iš tirtų žodžių išskyrimo metodų garse geriausiai pasirodė balso aktyvumo aptikimo algoritmas VAD.
4. Po garso klasifikavimo eksperimentų buvo pastebėta, kad klastotam balsui aptikti geriausiai tinka konvoliuciniai neuroniniai tinklai, transformeriai ir garso signalų transformacijos, kaip CQT ir NGCC, kurios garse išryškina aukštus dažnius. Apjungus keletą modelių išvesčių, galima gauti tikslesnę prognozę, nei taikant pavienius modelius. Dar vienas efektyvus priešiškos atakos įveikimo būdas yra modelių mokymo metu naudoti priešišku triukšmu paveiktus garso įrašus.
5. Atlikus visus skaitinius eksperimentus, geriausius rezultatus pasiekę kiekvienos principinės dalies metodai buvo apjungti į prieš tai suformuotą sprendimą. Triukšmo mažinimui panaudotas ROF algoritmas, žodžių išskyrimui pamodifikuotas VAD, klasifikavimui apjungta dešimt skirtingų modelių, kurie balsavimo būdu išveda bendrą prognozę.
6. Atlikus apjungto sprendimo testavimą, paaiškėjo, kad garso mažinimo strategija nepasiteisino. Algoritmas ROF garso failus pamodifikuoja taip, kad jie dažniau aptinkami kaip klastoti. Tai padidiną pačių klastotų balsų teisingo aptikimo procentą, tačiau žymiai sumažina teisingo tikro balso aptikimo procentą. Atskirų žodžių išskyrimas ir bendros prognozės išvedimas pagal juos pasiteisino. Fokusuojantis į pačius garse girdimus žodžius, modeliai tiksliau identifikuoja klastotus balsus. Įgyvendintą algoritmą palyginus su jau egzistuojančiais sprendimais atrasta, kad darbe pasiūlytas galutinis sprendimas turi patį geriausią DCF įvertį iš pačiame naujausiame *ASVspoof2024* iššūkyje patiektų klastoto balso aptikimo technologijų.

Literatūros sąrašas

1. KHAN, A. ir kt. Battling voice spoofing: a review, comparative analysis, and generalizability evaluation of state-of-the-art voice spoofing counter measures. In *Artificial Intelligence Review*. 2023. Vol. 56, no. 1, p. 513–566, ISSN 1573-7462. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1007/s10462-023-10539-8.
2. AMEZAGA, N. - HAJEK, J. Availability of Voice Deepfake Technology and its Impact for Good and Evil. In *SIGITE 2022 - Proceedings of the 23rd Annual Conference on Information Technology Education* [interaktyvus]. Chicago, IL, USA: Association for Computing Machinery, Inc, 2022. p. 23–28. [žiūrėta 2026-04-16]. Prieiga per internetą: https://dl.acm.org/doi/10.1145/3537674.3554742.
3. WANG, X. - YAMAGISHI, J. A Practical Guide to Logical Access Voice Presentation Attack Detection. In *arXiv preprint arXiv:2201.03321* [interaktyvus]. 2022. no. 1, p. 169–214. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2201.03321v1.
4. HUYNH, N.D. ir kt. Adversarial Attacks on Speech Recognition Systems for Mission-Critical Applications: A Survey. In *arXiv preprint arXiv:2202.10594* [interaktyvus]. 2022. no. 1. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2202.10594v1.
5. WANG, X. ir kt. ASVspoof 5: crowdsourced speech data, deepfakes, and adversarial attacks at scale. In *arXiv preprint arXiv:2408.08739* [interaktyvus]. 2024. p. 1–8. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.48550/arXiv.2408.08739.
6. BHAGTANI, K. ir kt. Are Recent Deepfake Speech Generators Detectable? In *IH MMSec 2024 - Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security* [interaktyvus]. 2024. p. 277–282, ISBN 9798400706370. [žiūrėta 2026-04-16]. Prieiga per internetą: https://dl.acm.org/doi/10.1145/3658664.3659658.
7. CASANOVA, E. ir kt. XTTS: a Massively Multilingual Zero-Shot Text-to-Speech Model. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. [s.l.]: International Speech Communication Association, 2024. p. 4978–4982. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2406.04904v1.
8. LI, Y.A. ir kt. StyleTTS: A Style-Based Generative Model for Natural and Diverse Text-to-Speech Synthesis. In *IEEE Journal on Selected Topics in Signal Processing* [interaktyvus]. 2022. Vol. 19, no. 1, p. 283–296, ISSN 1941-0484. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2205.15439v2.
9. CHEN, S. ir kt. Takin: A Cohort of Superior Quality Zero-shot Speech Generation Models. In *arXiv preprint arXiv:2409.12139* [interaktyvus]. 2024. [žiūrėta 2026-04-16]. Prieiga per internetą: http://arxiv.org/abs/2409.12139.
10. YENDURI, G. ir kt. Generative Pre-trained Transformer: A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions. In *IEEE Access* [interaktyvus]. 2023. Vol. 12, p. 54608–54649, ISSN 2169-3536. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2305.10435v2.
11. RADFORD, A. ir kt. Language Models are Unsupervised Multitask Learners. In [interaktyvus]. [žiūrėta 2026-05-19]. Prieiga per internetą: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.
12. LUO, T. ir kt. WaveFM: A High-Fidelity and Efficient Vocoder Based on Flow Matching. In *Proceedings of the 2025 Annual Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies: Long Papers,*

NAACL-HLT 2025 [interaktyvus]. Albuquerque, New Mexico: Association for Computational Linguistics (ACL), 2025. p. 2187–2198. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2503.16689v1.

13. HUANG, R. ir kt. ProDiff: Progressive Fast Diffusion Model For High-Quality Text-to-Speech. In *MM 2022 - Proceedings of the 30th ACM International Conference on Multimedia* [interaktyvus]. Lisbon, Portugal: Association for Computing Machinery, Inc, 2022. p. 2595–2605. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2207.06389v1.
14. REN, Z. Selection of Optimal Solution for Example and Model of Retrieval Based Voice Conversion. In *Proceedings of the 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)* [interaktyvus]. [s.l.]: Atlantis Press, 2024. p. 468–475. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.2991/978-94-6463-370-2_48.
15. BIRD, J.J. - LOTFI, A. Real-time Detection of AI-Generated Speech for DeepFake Voice Conversion. In *arXiv preprint arXiv:2308.12734* [interaktyvus]. 2023. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2308.12734v1.
16. Voice Models: Over 27,900+ Unique AI RVC Models. In [interaktyvus]. [žiūrėta 2026-04-16]. Prieiga per internetą: https://voice-models.com/.
17. BALL, J. Voice Activity Detection (VAD) in Noisy Environments. In *arXiv preprint arXiv:2312.05815* [interaktyvus]. 2023. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/2312.05815v1.
18. SOFER, A. - CHAZAN, S.E. CNN self-attention voice activity detector. In *arXiv preprint arXiv:2203.02944* [interaktyvus]. 2022. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/2203.02944v1.
19. RADFORD, A. ir kt. Robust Speech Recognition via Large-Scale Weak Supervision. In *Proceedings of Machine Learning Research* [interaktyvus]. 2022. Vol. 202, p. 28492–28518, ISSN 2640-3498. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/2212.04356v1.
20. GOMEZ-ALANIS, A. ir kt. Adversarial Transformation of Spoofing Attacks for Voice Biometrics. In *arXiv preprint arXiv:2201.01226* [interaktyvus]. 2022. p. 255–259. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2201.01226v1.
21. ARUN GEORGE ZACHARIAH A Deep Dive into the Fast Gradient Sign Method | by Arun George Zachariah | Medium. In [interaktyvus]. 2023. [žiūrėta 2026-04-16]. Prieiga per internetą: https://medium.com/@zachariaharungeorge/a-deep-dive-into-the-fast-gradient-sign-method-611826e34865.
22. YUAN, X. ir kt. Adversarial Examples: Attacks and Defenses for Deep Learning. In *IEEE transactions on neural networks and learning systems* [interaktyvus]. 2017. Vol. 30, no. 9, p. 2805–2824, ISSN 2162-2388. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1712.07107v3.
23. CHAKRABORTY, A. ir kt. A survey on adversarial attacks and defences. In *CAAI Transactions on Intelligence Technology* [interaktyvus]. 2021. Vol. 6, no. 1, p. 25–45, ISSN 2468-2322. [žiūrėta 2026-04-16]. Prieiga per internetą: https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/cit2.12028.
24. ABOMAKHELB, A. ir kt. A Comprehensive Review of Adversarial Attacks and Defense Strategies in Deep Neural Networks. In *Technologies 2025, Vol. 13,* [interaktyvus]. 2025. Vol. 13,

no. 5, ISSN 2227-7080. [žiūrėta 2026-04-16]. Prieiga per internetą: https://www.mdpi.com/2227-7080/13/5/202.

25. PIZZI, K. ir kt. Comparative Study on Noise-Augmented Training and its Effect on Adversarial Robustness in ASR Systems. In *Computer Speech & Language* [interaktyvus]. 2026. Vol. 96, p. 123–140, ISSN 0885-2308. [žiūrėta 2026-04-16]. Prieiga per internetą: http://arxiv.org/abs/2409.01813.
26. LI, B. A principal component analysis approach to noise removal for speech denoising. In *Proceedings - 2018 International Conference on Virtual Reality and Intelligent Systems, ICVRIS 2018*. Hunan, China: Institute of Electrical and Electronics Engineers Inc., 2018. p. 429–432. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/ICVRIS.2018.00111.
27. COLOM, M. - BUADES, A. Analysis and Extension of the PCA Method, Estimating a Noise Curve from a Single Image. In *Image Processing On Line* [interaktyvus]. 2016. Vol. 6, p. 365–390, ISSN 2105-1232. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.5201/ipol.2016.124.
28. PETROV, O. V. The use of self-adaptive principal components in PCA-based denoising. In *Journal of Magnetic Resonance*. 2025. Vol. 371, p. 107824, ISSN 1090-7807. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1016/J.JMR.2024.107824.
29. GETREUER, P. Rudin-Osher-Fatemi Total Variation Denoising using Split Bregman. In *Image Processing On Line* [interaktyvus]. 2012. Vol. 2, p. 74–95, ISSN 2105-1232. [žiūrėta 2026-04-16]. Prieiga per internetą: https://www.ipol.im/pub/art/2012/g-tvd/.
30. LEE, S. ir kt. Defensive denoising methods against adversarial attack. In *24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* [interaktyvus]. London, United Kingdom: The Association for Computing Machinery (ACM), 2018. [žiūrėta 2026-05-19]. Prieiga per internetą: https://www.kdd.org/kdd2018/files/deep-learning-day/DLDay18_paper_33.pdf.
31. BANK, D. ir kt. Autoencoders. In *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook, Third Edition* [interaktyvus]. 2020. p. 353–374, ISBN 9783031246289. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2003.05991v2.
32. OKTAY, O. ir kt. Attention U-Net: Learning Where to Look for the Pancreas. In *arXiv preprint arXiv:1804.03999* [interaktyvus]. 2018. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1804.03999v3.
33. WENG, W. - ZHU, X. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *IEEE Access* [interaktyvus]. 2015. Vol. 9, p. 16591–16603, ISSN 2169-3536. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1505.04597v1.
34. LI, C. ir kt. Defense Against Adversarial Attacks via Adversarial Noise Denoising Networks in Image Recognition. In *Proceedings - 2023 International Conference on Networking and Network Applications, NaNA 2023*. Qingdao, China: Institute of Electrical and Electronics Engineers Inc., 2023. p. 520–526. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/NANA60121.2023.00092.
35. GUO, Q. ir kt. INOR—An Intelligent noise reduction method to defend against adversarial audio examples. In *Neurocomputing*. 2020. Vol. 401, p. 160–172, ISSN 0925-2312. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1016/j.neucom.2020.02.110.
36. JUNG, S. ir kt. Adversarial example denoising and detection based on the consistency between Fourier-transformed layers. In *Neurocomputing*. 2024. Vol. 606, p. 128351, ISSN 0925-2312. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1016/J.NEUCOM.2024.128351.

37. BOASHASH, B. ir kt. Time-frequency features for pattern recognition using high-resolution TFDs: A tutorial review. In *Digital Signal Processing: A Review Journal*. 2015. Vol. 40, no. 1, p. 1–30, ISSN 1051-2004. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1016/j.dsp.2014.12.015.
38. JUNG, J.W. ir kt. RawNet: Advanced end-to-end deep neural network using raw waveforms for text-independent speaker verification. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. Graz, Austria: International Speech Communication Association, 2019. p. 1268–1272. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1904.08104v2.
39. MITTAL, G. ir kt. PITCH: AI-assisted Tagging of Deepfake Audio Calls using Challenge-Response. In *Proceedings of ACM Conference (Conference'17)* [interaktyvus]. New York, United States: Association for Computing Machinery, 2024. p. 559–575. [žiūrėta 2026-05-08]. Prieiga per internetą: http://arxiv.org/abs/2402.18085.
40. GIANNAKOPOULOS, T. - PIKRAKIS, A. Audio Features. In *Introduction to Audio Analysis*. Oxford: Elsevier, 2014. p. 59–103. [žiūrėta 2026-04-16]. ISBN 9780080993881. Prieiga per internetą: https://doi.org/10.1016/B978-0-08-099388-1.00004-2.
41. YU, Y. ir kt. Spectral Roll-off Points Variations: Exploring Useful Information in Feature Maps by Its Variations. In *arXiv preprint arXiv:2102.00369* [interaktyvus]. 2021. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2102.00369v2.
42. DRONGELEN, W. VAN Continuous, Discrete, and Fast Fourier Transform. In *Signal Processing for Neuroscientists*. [s.l.]: Elsevier, 2007. p. 91–105. [žiūrėta 2026-05-08]. ISBN 9780123708670. Prieiga per internetą: https://doi.org/10.1016/B978-012370867-0/50006-1.
43. KHODZHAEV, Z. A Practical Guide to Spectrogram Analysis for Audio Signal Processing. In *arXiv preprint arXiv:2403.09321* [interaktyvus]. 2024. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2403.09321v1.
44. YANG, Y. ir kt. Mel-McNet: A Mel-Scale Framework for Online Multichannel Speech Enhancement. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. [s.l.]: International Speech Communication Association, 2025. p. 1173–1177. [žiūrėta 2026-05-08]. Prieiga per internetą: https://arxiv.org/abs/2505.19576v1.
45. HOLIGHAUS ir kt. A framework for invertible, real-time constant-Q transforms. In *IEEE Transactions on Audio, Speech and Language Processing* [interaktyvus]. 2012. Vol. 21, no. 4, p. 775–785. [žiūrėta 2026-05-19]. Prieiga per internetą: http://arxiv.org/abs/1210.0084.
46. KAMBLE, M.R. ir kt. Advances in anti-spoofing: from the perspective of ASVspoof challenges. In *APSIPA Transactions on Signal and Information Processing* [interaktyvus]. 2020. Vol. 9, no. 1, p. 1–18, ISSN 2048-7703. [žiūrėta 2026-04-16]. Prieiga per internetą: https://dx.doi.org/10.1017/ATSIP.2019.21.
47. XIAO, X. ir kt. Spoofing speech detection using high dimensional magnitude and phase features: the NTU approach for ASVspoof 2015 challenge. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. Dresden, Germany: International Speech and Communication Association, 2015. p. 2052–2056. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.21437/INTERSPEECH.2015-465.
48. MOHAMMADI, M. - SADEGH MOHAMMADI, H.R. Robust features fusion for text independent speaker verification enhancement in noisy environments. In *2017 25th Iranian*

Conference on Electrical Engineering, ICEE 2017 [interaktyvus]. 2017. p. 1863–1868, ISBN 9781509059638. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/IranianCEE.2017.7985357.

49. MON, K.Z. ir kt. Spoof Detection using Voice Contribution on LFCC features and ResNet-34. In *18th International Conference on Artificial Intelligence and Natural Language Processing and International Conference on Artificial Intelligence and Internet of Things, iSAI-NLP 2023* [interaktyvus]. Bangkok, Thailand: Institute of Electrical and Electronics Engineers Inc., 2023. p. 1–6. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/ISAI-NLP60301.2023.10354625.
50. ABDUL, Z.K. - AL-TALABANI, A.K. Mel Frequency Cepstral Coefficient and its Applications: A Review. In *IEEE Access* [interaktyvus]. 2022. Vol. 10, p. 122136–122158, ISSN 2169-3536. [žiūrėta 2026-04-16]. Prieiga per internetą: https://ieeexplore.ieee.org/document/9955539.
51. TODISCO, M. ir kt. Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. In *Computer Speech & Language* [interaktyvus]. 2017. Vol. 45, p. 516–535, ISSN 0885-2308. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1016/J.CSL.2017.01.001.
52. ROHDIN, J. ir kt. BUT Systems and Analyses for the ASVspoof 5 Challenge. In *arXiv preprint arXiv:2408.11152* [interaktyvus]. 2024. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2408.11152v1.
53. WANG, X. ir kt. ASVspoof 5: Evaluation of Spoofing, Deepfake, and Adversarial Attack Detection Using Crowdsourced Speech. In *arXiv preprint arXiv:2601.03944* [interaktyvus]. 2026. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/html/2601.03944v1#S4.
54. ALBAWI, S. ir kt. Understanding of a convolutional neural network. In *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*. Antalya, Turkey: Institute of Electrical and Electronics Engineers Inc., 2017. p. 1–6. [žiūrėta 2026-05-07]. Prieiga per internetą: https://doi.org/10.1109/ICENGTECHNOL.2017.8308186.
55. HERSHEY, S. ir kt. CNN architectures for large-scale audio classification. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. New Orleans, LA, USA: Institute of Electrical and Electronics Engineers Inc., 2017. p. 131–135. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/ICASSP.2017.7952132.
56. HE, K. ir kt. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* [interaktyvus]. Las Vegas, NV, USA: IEEE Computer Society, 2015. p. 770–778. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1512.03385v1.
57. CHEN, Y. ir kt. USTC-KXDIGIT system description for ASVspoof5 Challenge. In *The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)* [interaktyvus]. Kos, Greece: International Speech Communication Association, 2024. p. 109–115. [žiūrėta 2026-05-06]. Prieiga per internetą: https://doi.org/10.21437/ASVspoof.2024-16.
58. SCHMIDT, R.M. Recurrent Neural Networks (RNNs): A gentle Introduction and Overview. In *arXiv preprint arXiv:1912.05911* [interaktyvus]. 2019. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/1912.05911v1.
59. PHAN, H. ir kt. Audio Scene Classification with Deep Recurrent Neural Networks. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. Stockholm, Sweden: International Speech Communication

Association, 2017. p. 3043–3047. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1703.04770v2.

60. TOHARUDIN, T. ir kt. Employing long short-term memory and Facebook prophet model in air temperature forecasting. In *Communications in Statistics: Simulation and Computation* [interaktyvus]. 2023. Vol. 52, no. 2, p. 279–290, ISSN 1532-4141. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1080/03610918.2020.1854302.
61. CHEN, Z. ir kt. Recurrent Neural Networks for Automatic Replay Spoofing Attack Detection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* [interaktyvus]. Calgary, AB, Canada, 2018. p. 2052–2056. [žiūrėta 2026-05-07]. Prieiga per internetą: https://ieeexplore.ieee.org/document/8462644.
62. ANWAR, Z. ir kt. AEXANet: An End-to-End Deep Learning based Voice Anti-spoofing System. In *Workshop on Artificial Intelligence for Multimedia Forensics and Disinformation Detection (AI4MFDD)* [interaktyvus]. 2022. [žiūrėta 2026-05-19]. Prieiga per internetą: https://par.nsf.gov/servlets/purl/10356298.
63. HUANG, L. - ZHAO, J. On the Use of LSTM-RNN for Detecting Audio Spoofing Attacks. In *Proceedings - 2022 International Conference on 3D Immersion, Interaction and Multi-Sensory Experiences, ICDIIME 2022* [interaktyvus]. Madrid, Spain: Institute of Electrical and Electronics Engineers Inc., 2022. p. 147–150. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/ICDIIME56946.2022.00040.
64. VASWANI, A. ir kt. Attention Is All You Need. In *arXiv preprint arXiv:1706.03762* [interaktyvus]. 2017. p. 1–15. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/1706.03762v7.
65. CHEN, S. ir kt. WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing. In *IEEE Journal on Selected Topics in Signal Processing* [interaktyvus]. 2022. Vol. 16, no. 6, p. 1505–1518, ISSN 1941-0484. [žiūrėta 2026-05-07]. Prieiga per internetą: http://arxiv.org/abs/2110.13900.
66. VIROLI, C. - MCLACHLAN, G.J. Deep Gaussian Mixture Models. In *Statistics and Computing* [interaktyvus]. 2017. Vol. 29, no. 1, p. 43–51, ISSN 1573-1375. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/1711.06929v1.
67. MAHON, L. - LAPATA, M. K*-Means: A Parameter-free Clustering Algorithm. In *arXiv preprint arXiv:2505.11904* [interaktyvus]. 2025. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/2505.11904v1.
68. LEI, Z. ir kt. GMM-ResNet2: Ensemble of Group ResNet Networks for Synthetic Speech Detection. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* [interaktyvus]. Seoul, Republic of Korea: Institute of Electrical and Electronics Engineers, 2024. p. 12101–12105. [žiūrėta 2026-04-16]. Prieiga per internetą: http://dx.doi.org/10.1109/ICASSP48485.2024.10447628.
69. CHETTRI, B. - STURM, B.L. A Deeper Look at Gaussian Mixture Model Based Anti-Spoofing Systems. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* [interaktyvus]. Calgary, AB, Canada: Institute of Electrical and Electronics Engineers Inc., 2018. p. 5159–5163. [žiūrėta 2026-04-16]. Prieiga per internetą: https://doi.org/10.1109/ICASSP.2018.8461467.
70. ZHU, Y. ir kt. Learn from real: reality defender's submission to ASVspoof5 Challenge. In *The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)*

[interaktyvus]. Kos, Greece: International Speech Communication Association, 2024. p. 116–123. [žiūrėta 2026-05-19]. Prieiga per internetą: https://doi.org/10.48550/arXiv.2410.07379.

71. MARTÍN-DOÑAS, J.M. ir kt. ASASVIcomtech: the Vicomtech-UGR speech deepfake detection and SASV systems for the ASVspoof5 Challenge. In *The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)*. Kos, Greece: International Speech Communication Association, 2024. p. 144–151. [žiūrėta 2026-05-07]. Prieiga per internetą: https://doi.org/10.21437/ASVSPOOF.2024-21.
72. JUNG, J.W. ir kt. Pushing the limits of raw waveform speaker recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. Incheon, South Korea: International Speech Communication Association, 2022. p. 2228–2232. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2203.08488v2.
73. TAK, H. ir kt. End-to-end anti-spoofing with RawNet2. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* [interaktyvus]. [s.l.]: Institute of Electrical and Electronics Engineers Inc., 2020. p. 6369–6373. [žiūrėta 2026-05-19]. Prieiga per internetą: https://doi.org/10.48550/arXiv.2011.01108.
74. WANG, F. ir kt. Additive Margin Softmax for Face Verification. In *IEEE Signal Processing Letters* [interaktyvus]. 2018. Vol. 25, no. 7, p. 926–930, ISSN 1070-9908. [žiūrėta 2026-04-16]. Prieiga per internetą: http://arxiv.org/abs/1801.05599.
75. JUNG, J.W. ir kt. AASIST: Audio Anti-Spoofing using Integrated Spectro-Temporal Graph Attention Networks. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* [interaktyvus]. Singapore, Singapore: Institute of Electrical and Electronics Engineers Inc., 2021. p. 2405–2409. [žiūrėta 2026-05-07]. Prieiga per internetą: https://arxiv.org/abs/2110.01200v1.
76. BORODIN, K. ir kt. AASIST3: KAN-enhanced AASIST speech deepfake detection using SSL features and additional regularization for the ASVspoof 2024 Challenge. In *The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)* [interaktyvus]. Kos, Greece: International Speech Communication Association, 2024. p. 48–55. [žiūrėta 2026-05-06]. Prieiga per internetą: https://doi.org/10.21437/ASVSPOOF.2024-8.
77. ASVspoof 2021 Duomenų rinkinys. In [interaktyvus]. [žiūrėta 2026-05-04]. Prieiga per internetą: https://www.asvspoof.org/index2021.html.
78. Jungjee/RawNet: Official repository for RawNet, RawNet2, and RawNet3. In [interaktyvus]. [žiūrėta 2026-05-01]. Prieiga per internetą: https://github.com/jungjee/RawNet.
79. jungjee/RawNet3 · Hugging Face. In [interaktyvus]. [žiūrėta 2026-05-01]. Prieiga per internetą: https://huggingface.co/jungjee/RawNet3.
80. SHEN, J. ir kt. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* [interaktyvus]. [s.l.]: Institute of Electrical and Electronics Engineers Inc., 2017. p. 4779–4783. [žiūrėta 2026-05-03]. Prieiga per internetą: https://arxiv.org/abs/1712.05884v2.
81. CASANOVA, E. ir kt. YourTTS: Towards Zero-Shot Multi-Speaker TTS and Zero-Shot Voice Conversion for everyone. In *Proceedings of Machine Learning Research* [interaktyvus]. 2021. Vol. 162, p. 2709–2720, ISSN 2640-3498. [žiūrėta 2026-05-03]. Prieiga per internetą: https://arxiv.org/abs/2112.02418v4.

82. AO, J. ir kt. SpeechT5: Unified-Modal Encoder-Decoder Pre-Training for Spoken Language Processing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics* [interaktyvus]. [s.l.]: Association for Computational Linguistics (ACL), 2021. p. 5723–5738. [žiūrėta 2026-05-03]. Prieiga per internetą: https://arxiv.org/abs/2110.07205v3.
83. RVC-Project/Retrieval-based-Voice-Conversion-WebUI: Easily train a good VC model with voice data <= 10 mins! In [interaktyvus]. [žiūrėta 2026-05-03]. Prieiga per internetą: https://github.com/RVC-Project/Retrieval-based-Voice-Conversion-WebUI.
84. UnitSpeech: Speaker-adaptive Speech Synthesis with Untranscribed Data (INTERSPEECH 2023) | Guided-TTS 2: A Diffusion Model for High-quality Adaptive Text-to-Speech with Untranscribed Data. In [interaktyvus]. [žiūrėta 2026-05-03]. Prieiga per internetą: https://unitspeech.github.io/.
85. KIM, H. ir kt. UnitSpeech: Speaker-adaptive Speech Synthesis with Untranscribed Data. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* [interaktyvus]. [s.l.]: International Speech Communication Association, 2023. p. 3038–3042. [žiūrėta 2026-05-03]. Prieiga per internetą: https://arxiv.org/abs/2306.16083v1.
86. ProDiff: Progressive Fast Diffusion Model for High-Quality Text-to-Speech. In [interaktyvus]. [žiūrėta 2026-05-03]. Prieiga per internetą: https://prodiff.github.io/.
87. LIU, S. ir kt. DiffGAN-TTS: High-Fidelity and Efficient Text-to-Speech with Denoising Diffusion GANs. In *arXiv preprint arXiv:2201.11972* [interaktyvus]. 2022. [žiūrėta 2026-04-16]. Prieiga per internetą: https://arxiv.org/abs/2201.11972v1.
88. keonlee9420/DiffGAN-TTS: PyTorch Implementation of DiffGAN-TTS: High-Fidelity and Efficient Text-to-Speech with Denoising Diffusion GANs. In [interaktyvus]. [žiūrėta 2026-05-03]. Prieiga per internetą: https://github.com/keonlee9420/DiffGAN-TTS.
89. XU, Y. ir kt. SZU-AFS antispoofing system for the ASVspoof 5 Challenge. In *The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)* [interaktyvus]. Kos, Greece: International Speech Communication Association, 2024. p. 64–71. [žiūrėta 2026-05-19]. Prieiga per internetą: https://doi.org/10.48550/arXiv.2408.09933.
90. CHAN, P.-C. ir kt. Enhancing spoofing detection in ASVspoof 5 Workshop 2024: fusion of WavLM-ResNet18-SA for optimal performance against speech deepfakes. In *The Automatic Speaker Verification Spoofing Countermeasures Workshop (ASVspoof 2024)* [interaktyvus]. Kos, Greece: International Speech Communication Association, 2024. p. 158–162. [žiūrėta 2026-05-07]. Prieiga per internetą: https://doi.org/10.21437/ASVSPOOF.2024-23.

Priedai

1 priedas. Pilni klasifikatorių eksperimentų rezultatai

Požymis	Modelis	DCF	Tikslumas	Preciziškumas	Jautrumas	F1	Modelio parametrai	Mokymo epochos
Spektrograma	CNN2D	0.065108	0.951546	0.95172	0.951619	0.951545	96543793	20
NGCC	CNN2D	0.11422	0.921719	0.921672	0.92172	0.921694	3712049	20
PNCC	CNN2D	0.118907	0.916484	0.916541	0.916448	0.916473	3712049	20
GFCC	CNN2D	0.136182	0.898223	0.898609	0.898973	0.898214	3712049	20
BFCC	CNN2D	0.180485	0.880326	0.880042	0.880791	0.880205	3712049	20
Spektrograma	GRU2D	0.19541	0.869126	0.868886	0.869421	0.869017	542721	30
Spektrograma	LSTM2D	0.254431	0.85963	0.857747	0.880001	0.857239	723585	30
CQT	CNN2D	0.228537	0.852325	0.851764	0.853955	0.85198	7988273	20
Garso banga	RawNet3	0.21148	0.84429	0.844767	0.845446	0.844253	16280834	30
Garso banga	RawNet3	0.212625	0.843803	0.844264	0.84488	0.84377	16280834	30
NGCC	GRU2D	0.231495	0.84356	0.843349	0.843729	0.843448	154113	30
Garso banga	RawNet3	0.213209	0.843438	0.843897	0.844504	0.843406	16280834	40
GD	CNN2D	0.239786	0.830777	0.830934	0.83087	0.830774	96543793	20
MFCC	CNN2D	0.278074	0.813854	0.813509	0.814295	0.813633	1205297	20
MGD	CNN2D	0.26663	0.81215	0.812309	0.812252	0.812148	96543793	20
NGCC	LSTM2D	0.307572	0.808132	0.807017	0.813697	0.806847	205441	30
Kompleksinė CQT	CNN2D	0.258096	0.807645	0.808349	0.810068	0.807473	7988529	20
MGD	LSTM2D	0.280083	0.806915	0.80686	0.806865	0.806862	723073	30
IMFCC	CNN2D	0.304931	0.804967	0.804113	0.808084	0.804127	1205297	20
CQT	Vizijos transformeris	0.31131	0.803628	0.802612	0.808097	0.802506	2280369	30
CQCC	CNN2D	0.289299	0.803506	0.803295	0.803603	0.803373	3712049	20
Kompleksinė CQT	Vizijos transformeris	0.306915	0.802654	0.801849	0.805369	0.801881	2280945	30
CQT	LSTM2D	0.340102	0.80168	0.799722	0.819587	0.798105	241793	30
BFCC	Vizijos transformeris	0.298831	0.799123	0.798796	0.799462	0.798903	2518449	30
CQT	GRU2D	0.32878	0.798758	0.797364	0.807316	0.796787	181377	30
PNCC	Vizijos transformeris	0.334831	0.798515	0.796914	0.809988	0.795994	2518449	30
GD	LSTM2D	0.298417	0.797784	0.797542	0.797926	0.797627	723073	30
LFCC	CNN2D	0.309106	0.794984	0.7945	0.795827	0.794611	1205297	20
MGD	Vizijos transformeris	0.305345	0.789871	0.789791	0.789821	0.789805	2358129	30
GD	GRU2D	0.304772	0.786827	0.786928	0.786854	0.786818	542337	30
MFCC	Vizijos transformeris	0.345471	0.78634	0.78499	0.793897	0.784371	2009265	30

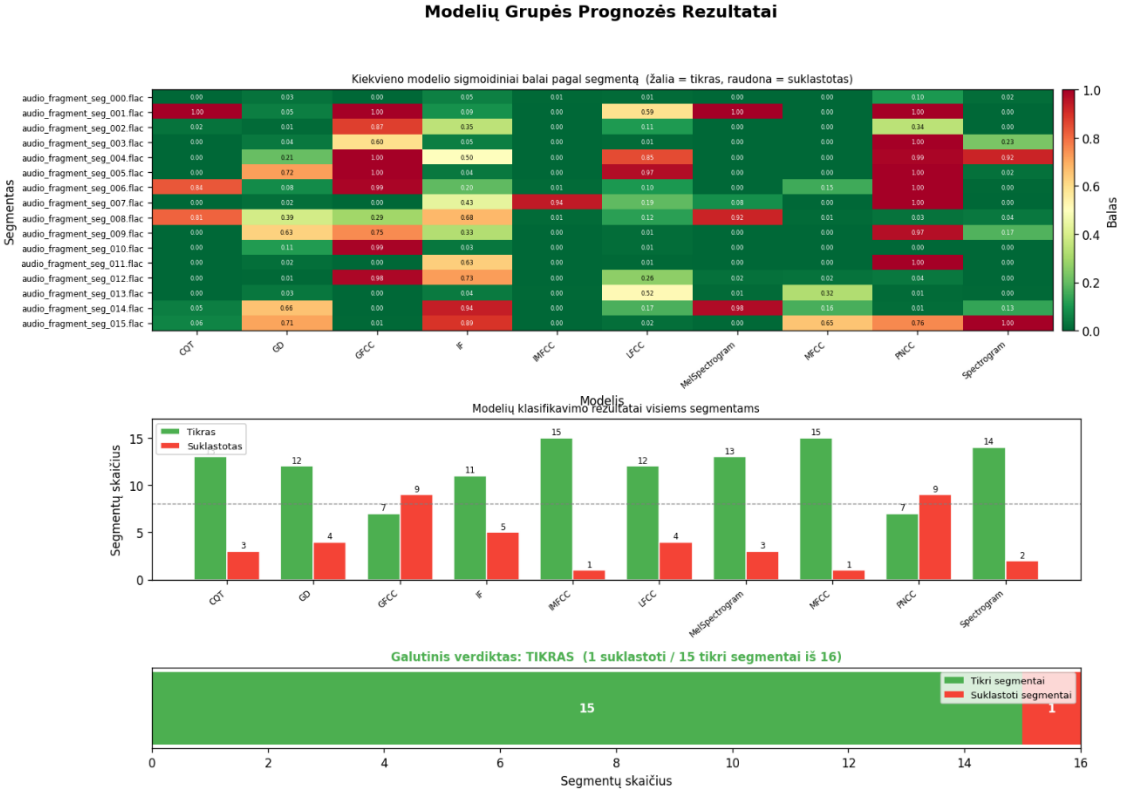
BFCC	GRU2D	0.327161	0.784271	0.783693	0.785449	0.783768	154113	30
LFCC	Vizijos transformeris	0.344035	0.783175	0.782043	0.788286	0.781701	2009265	30
Spektrinis srautas	CNN1D	0.337424	0.776966	0.776398	0.778044	0.776462	1543553	30
IMFCC	Vizijos transformeris	0.355685	0.77514	0.774005	0.78007	0.773611	2009265	30
MGD	GRU2D	0.340528	0.77441	0.773863	0.775379	0.773928	542337	30
GD	Vizijos transformeris	0.365888	0.773923	0.772482	0.782121	0.771596	2358129	30
CQCC	Vizijos transformeris	0.363477	0.770636	0.769454	0.775895	0.768967	2518449	30
BFCC	LSTM2D	0.362345	0.763331	0.762573	0.765237	0.762506	205441	30
Spektriniai centroidai	CNN1D	0.348101	0.76114	0.761012	0.761095	0.761042	1543553	30
IMFCC	LSTM2D	0.405783	0.760653	0.758459	0.77992	0.755379	205441	30
Spektrograma	Vizijos transformeris	0.385123	0.760166	0.758747	0.767577	0.757775	2519217	5
Spektrinis nuokrypis	CNN1D	0.332067	0.759313	0.759863	0.760699	0.759204	1543553	30
Spektrinis srautas	Transformeris	0.388934	0.758218	0.756763	0.765971	0.755699	1592961	20
IMFCC	GRU2D	0.399525	0.755296	0.75361	0.76583	0.751978	154113	30
GFCC	Vizijos transformeris	0.419284	0.749452	0.747355	0.765991	0.744389	2518449	30
MFCC	LSTM2D	0.415705	0.746896	0.745062	0.759032	0.742908	205441	30
MSRCC	CNN2D	0.421196	0.742391	0.740594	0.753737	0.738484	3712049	20
Spektriniai centroidai	Transformeris	0.395909	0.741052	0.740242	0.742967	0.740062	1592961	20
CQCC	GRU2D	0.392683	0.734965	0.73459	0.735203	0.734641	154113	30
Spektrinis nuokrypis	Transformeris	0.358023	0.729851	0.731001	0.735126	0.728922	1592961	20
MSRCC	Vizijos transformeris	0.41786	0.718773	0.718332	0.719093	0.718351	2518449	30
Spektrinis nuokrypis	Transformeris	0.403725	0.715486	0.71573	0.715785	0.715483	1592961	10
IMFCC	Vizijos transformeris	0.448101	0.710007	0.708929	0.71299	0.708247	2009265	5
Spektriniai centroidai	Transformeris	0.462126	0.70672	0.705305	0.712138	0.703862	1592961	10
Fazė	Vizijos transformeris	0.440346	0.706477	0.705864	0.707196	0.705768	2519217	30
MSRCC	GRU2D	0.454419	0.704894	0.703854	0.707535	0.703214	154113	30
LFCC	LSTM2D	0.414889	0.702459	0.70298	0.703593	0.702335	205441	30
Spektrinis nuokrypis	GRU1D	0.505296	0.701242	0.698548	0.723882	0.691632	149505	30

MSRCC	LSTM2D	0.469223	0.697103	0.695933	0.700374	0.695	205441	30
ZCR	CNN1D	0.434539	0.69552	0.695691	0.695681	0.69552	1543553	30
Spektriniai centroidai	GRU1D	0.482895	0.694059	0.692553	0.699782	0.690739	149505	30
NGCC	Vizijos transformeris	0.372681	0.693937	0.696434	0.715766	0.687665	2518449	30
Spektrinė plokštuma	Transformeris	0.459046	0.693694	0.693072	0.694345	0.692944	1592961	20
ZCR	Transformeris	0.445751	0.692963	0.692862	0.692878	0.692869	1592961	20
CQCC	LSTM2D	0.537972	0.692233	0.688825	0.730129	0.676562	205441	30
F0	CNN1D	0.45711	0.688946	0.688642	0.688947	0.688669	1543553	30
RPLP	CNN2D	0.446944	0.687607	0.687742	0.687713	0.687604	3712049	20
PNCC	LSTM2D	0.481556	0.686633	0.685564	0.689067	0.684778	205441	30
F0	Transformeris	0.45571	0.675554	0.676002	0.676382	0.675466	1592961	20
MFCC	Vizijos transformeris	0.445605	0.674702	0.675561	0.677272	0.674137	2009265	5
MFCC	GRU2D	0.456635	0.67178	0.672392	0.67318	0.671549	154113	30
Spektrinis nuokrypis	LSTM1D	0.458096	0.665936	0.666801	0.66846	0.665342	199297	30
LFCC	GRU2D	0.47738	0.663745	0.664025	0.664119	0.663731	154113	30
Spektrinė plokštuma	CNN1D	0.460226	0.663258	0.664185	0.666092	0.662543	1543553	30
Fazė	CNN2D	0.481665	0.662527	0.662716	0.662724	0.662527	96543793	20
Spektrinė plokštuma	Transformeris	0.59104	0.658997	0.655404	0.69385	0.639784	1592961	10
PNCC	GRU2D	0.513465	0.654614	0.654063	0.654837	0.653931	154113	30
Spektriniai centroidai	LSTM1D	0.494875	0.649866	0.650238	0.650447	0.649814	199297	30
Garso banga	Transformeris	0.512637	0.636596	0.637018	0.637288	0.636509	1653249	20
PSRCC	Vizijos transformeris	0.466959	0.628804	0.63129	0.644281	0.621157	2518449	30
PSRCC	GRU2D	0.48858	0.628439	0.63016	0.635934	0.625041	154113	30
PSRCC	LSTM2D	0.489603	0.628074	0.629777	0.635407	0.624751	205441	30
PSRCC	CNN2D	0.531446	0.6271	0.627337	0.627381	0.627093	3712049	20
Spektrinė plokštuma	LSTM1D	0.47262	0.623691	0.626239	0.639457	0.615499	199297	30
Spektrinė plokštuma	GRU1D	0.472255	0.621865	0.624522	0.638854	0.612842	149505	30
Fazė	GRU2D	0.531678	0.619308	0.619944	0.620573	0.618991	542721	30
ZCR	GRU1D	0.492647	0.617361	0.619514	0.628147	0.611592	149505	30
Fazė	LSTM2D	0.491125	0.615047	0.617377	0.627461	0.608147	723585	30
ZCR	LSTM1D	0.485257	0.614561	0.617128	0.629629	0.606011	199297	30
Garso banga	CNN1D	0.519917	0.608717	0.610333	0.614643	0.605585	75723	30
GFCC	LSTM2D	0.535573	0.605551	0.606766	0.609041	0.603902	205441	30

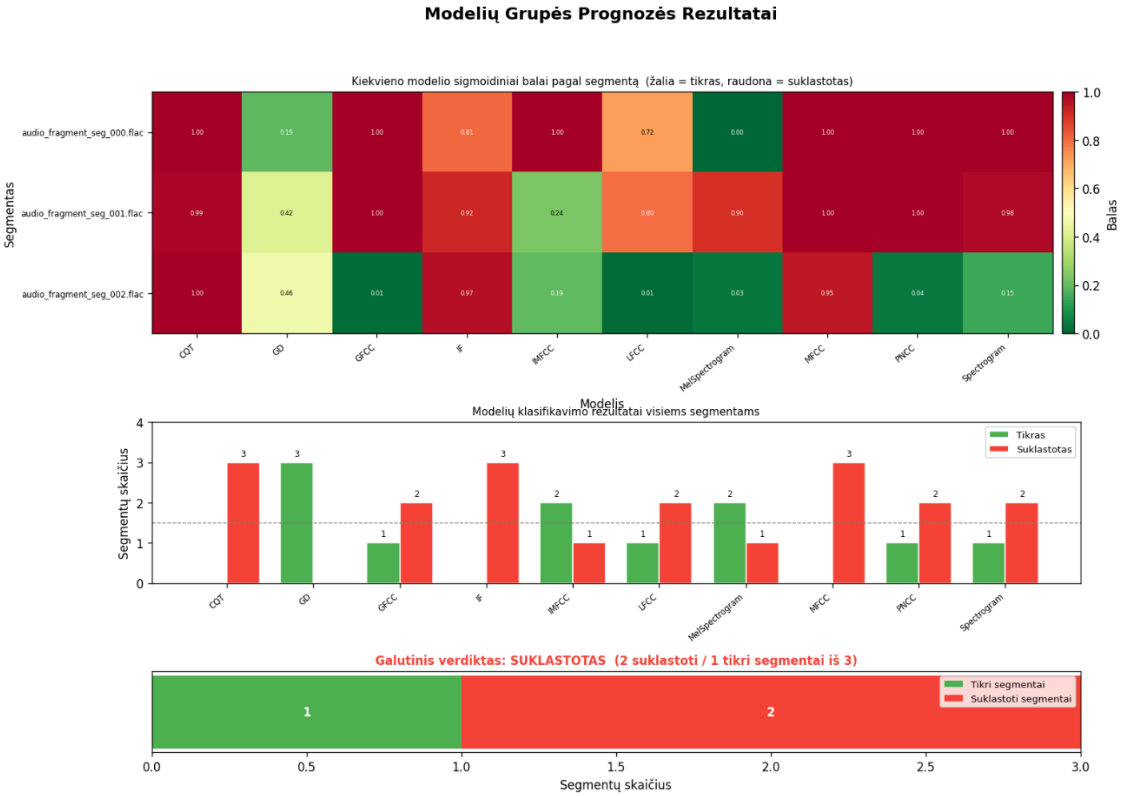
Autokoreliacija	Transformeris	0.511054	0.586681	0.589772	0.60452	0.572871	1653249	20
GFCC	GRU2D	0.511152	0.583516	0.586768	0.60287	0.567952	154113	30
F0	LSTM1D	0.690808	0.544436	0.543218	0.544211	0.541236	199297	30
Autokoreliacija	CNN1D	0.591466	0.528731	0.531938	0.537727	0.511493	75723	30
RPLP	GRU2D	0.516922	0.524714	0.530833	0.570267	0.450991	154113	30
F0	GRU1D	0.76226	0.524592	0.521821	0.524675	0.508972	149505	30
RPLP	Vizijos transformeris	0.896129	0.517166	0.509929	0.546751	0.393254	2518449	30
Spektrinis srautas	GRU1D	0.518091	0.492208	0.499986	0.499845	0.349338	149505	30
Spektrinis srautas	LSTM1D	0.508157	0.491843	0.5	0.245922	0.329688	199297	30
RPLP	LSTM2D	0.508157	0.491843	0.5	0.245922	0.329688	205441	30

2 priedas. Tikro garso įrašo klasifikavimo su triukšmo mažinimu rezultatas

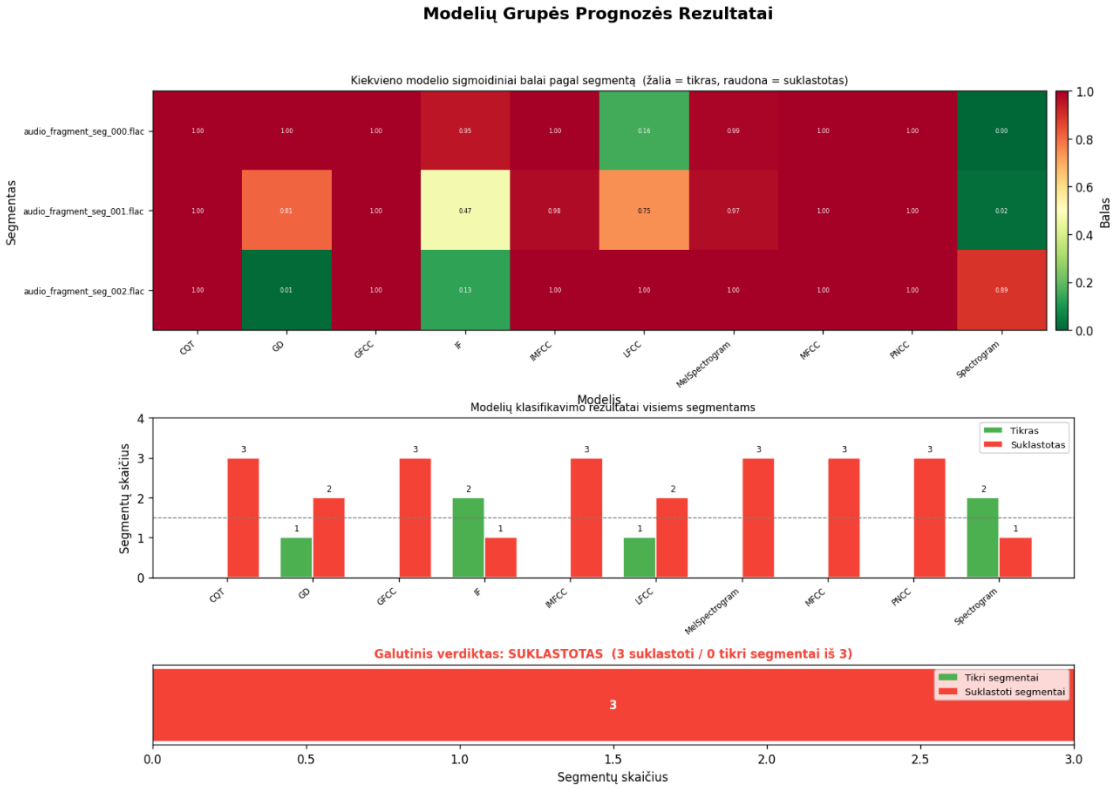
3 priedas. Tikro garso įrašo klasifikavimo be triukšmo mažinimo rezultatas



4 priedas. Klastoto garso įrašo klasifikavimo su triukšmo mažinimu rezultatas



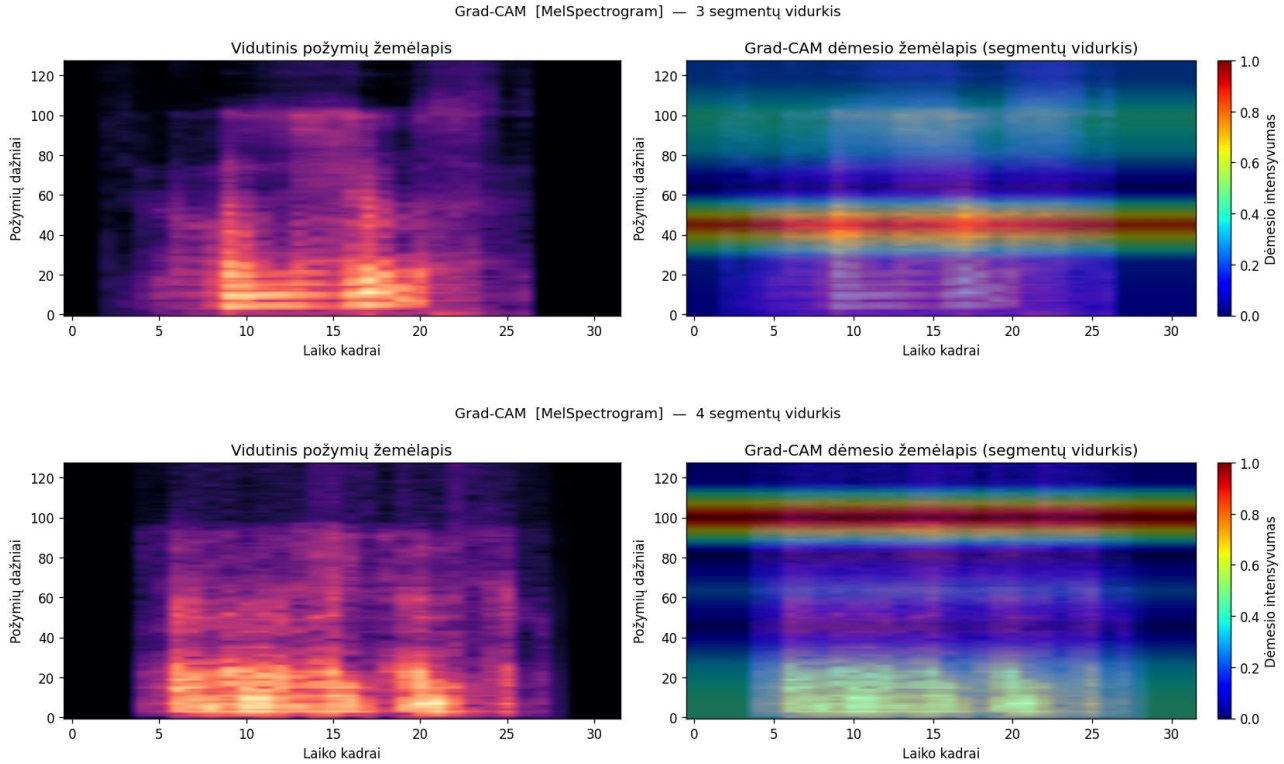
5 priedas. Klastoto garso įrašo klasifikavimo be triukšmo mažinimo rezultatas

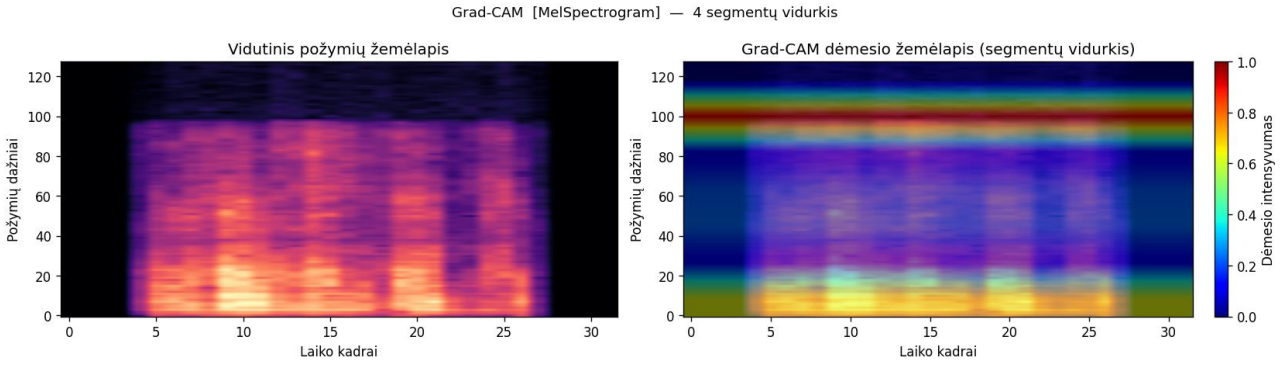


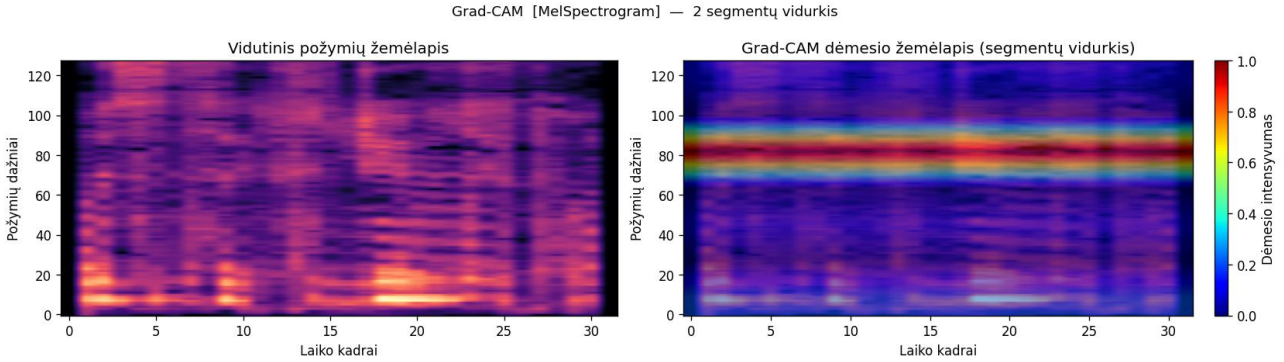
6 priedas. Triukšmingo klastoto garso įrašo klasifikavimo be triukšmo mažinimo rezultatas

7 priedas. Triukšmingo klastoto garso įrašo klasifikavimo su triukšmo mažinimu rezultatas

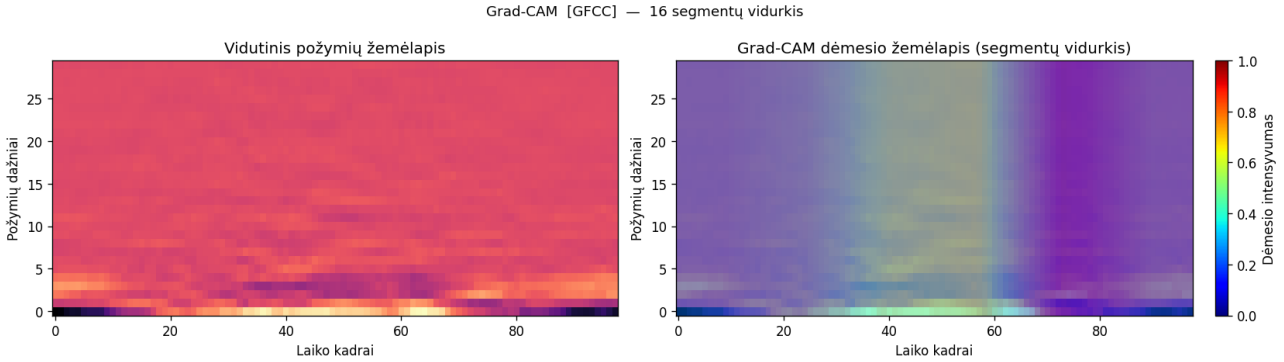
8 priedas. Melų skalės spektrogramų gradCAM įvertinimai įvairiuose garso failuose

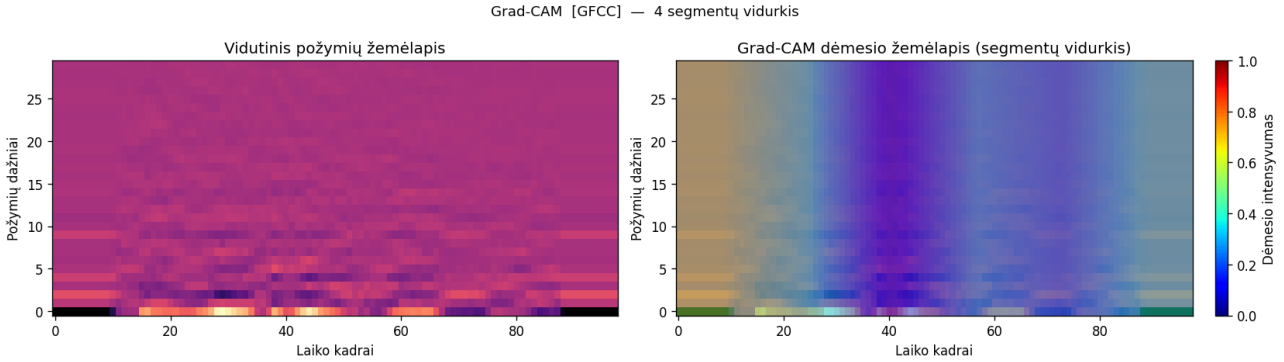


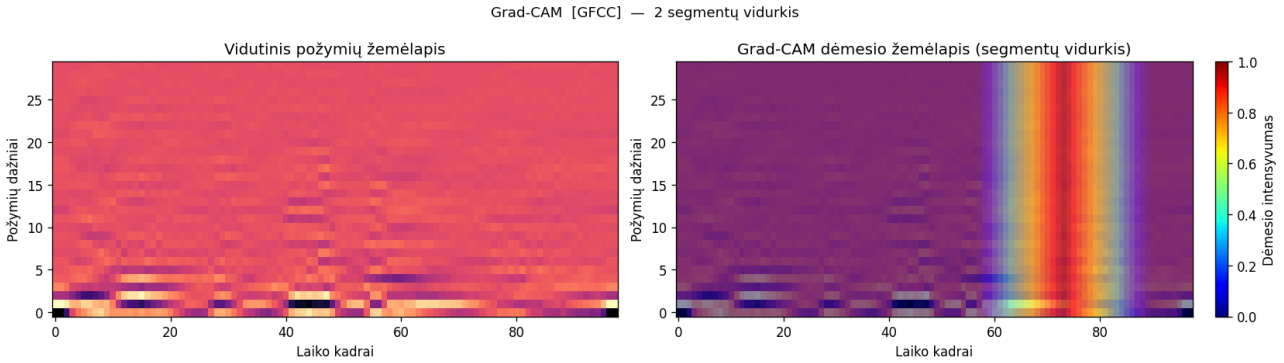




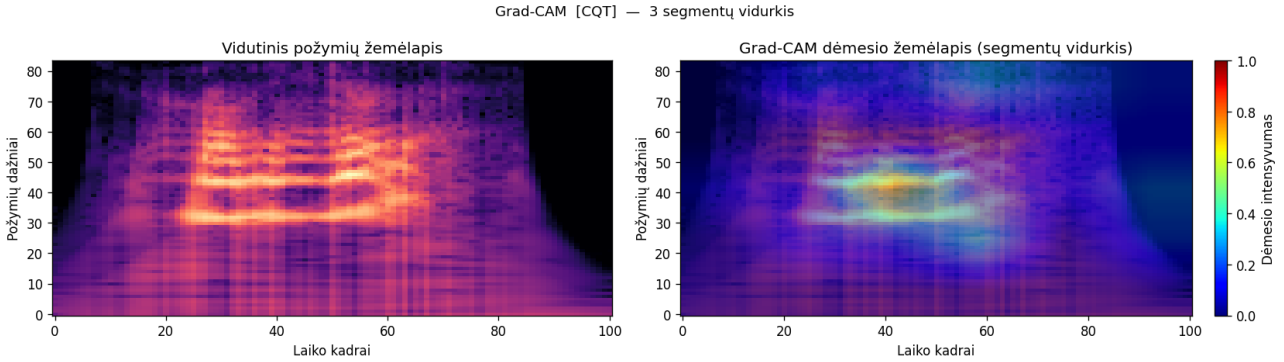
9 priedas. GFCC gradCAM įvertinimai įvairiuose garso failuose

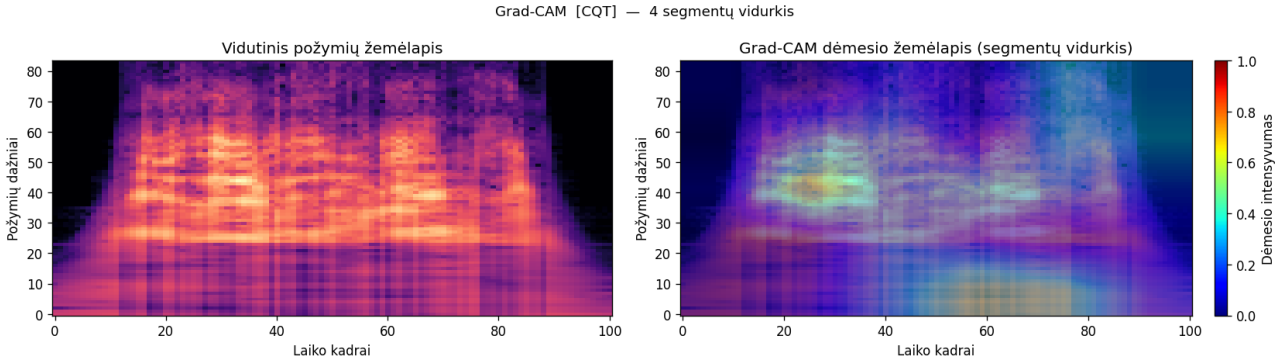


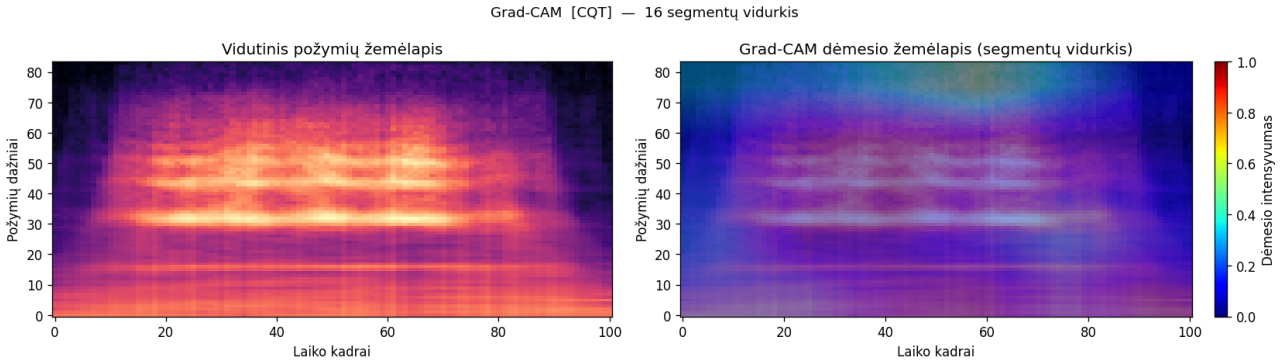




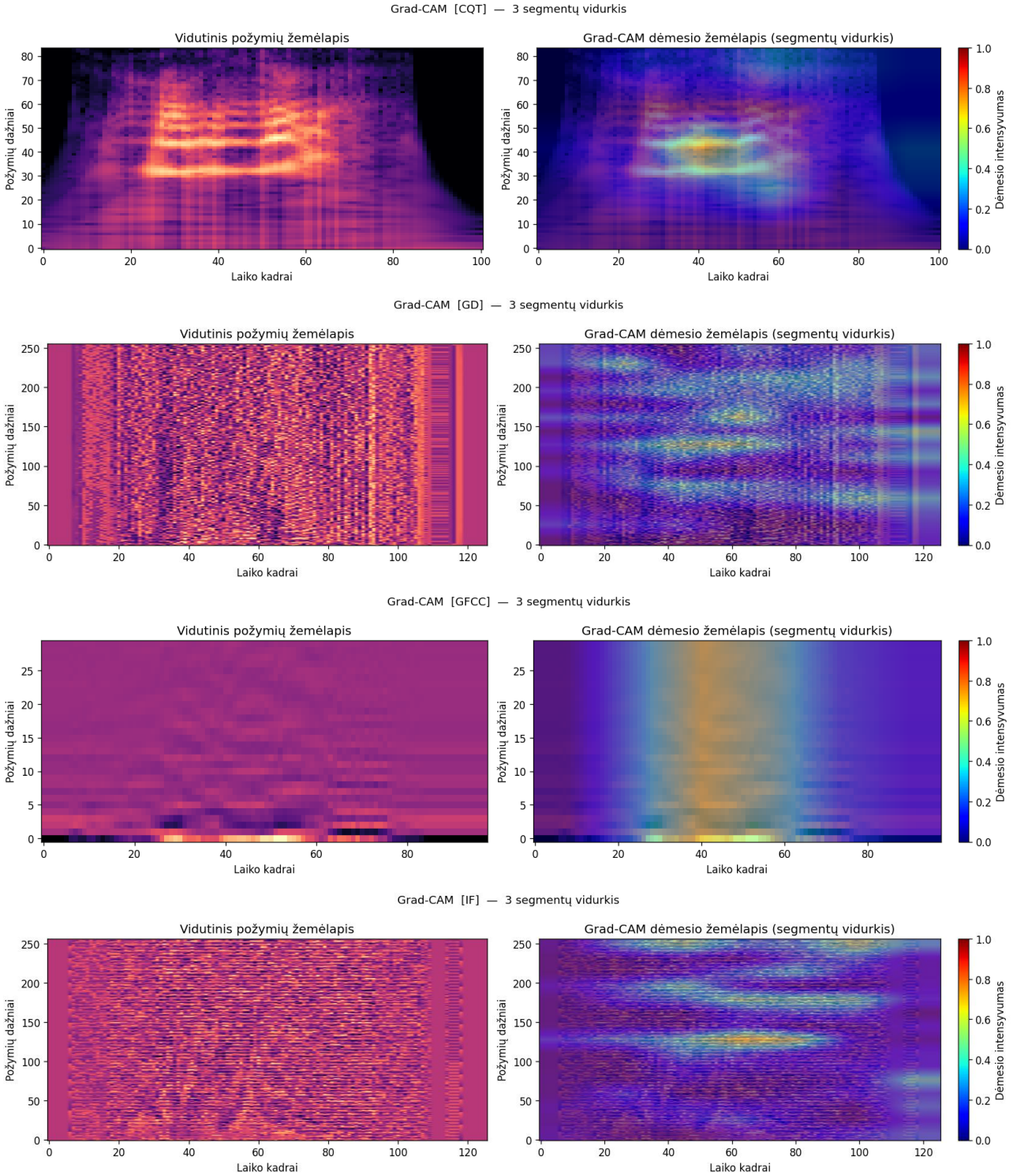
10 priedas. CQT gradCAM įvertinimai įvairiuose garso failuose

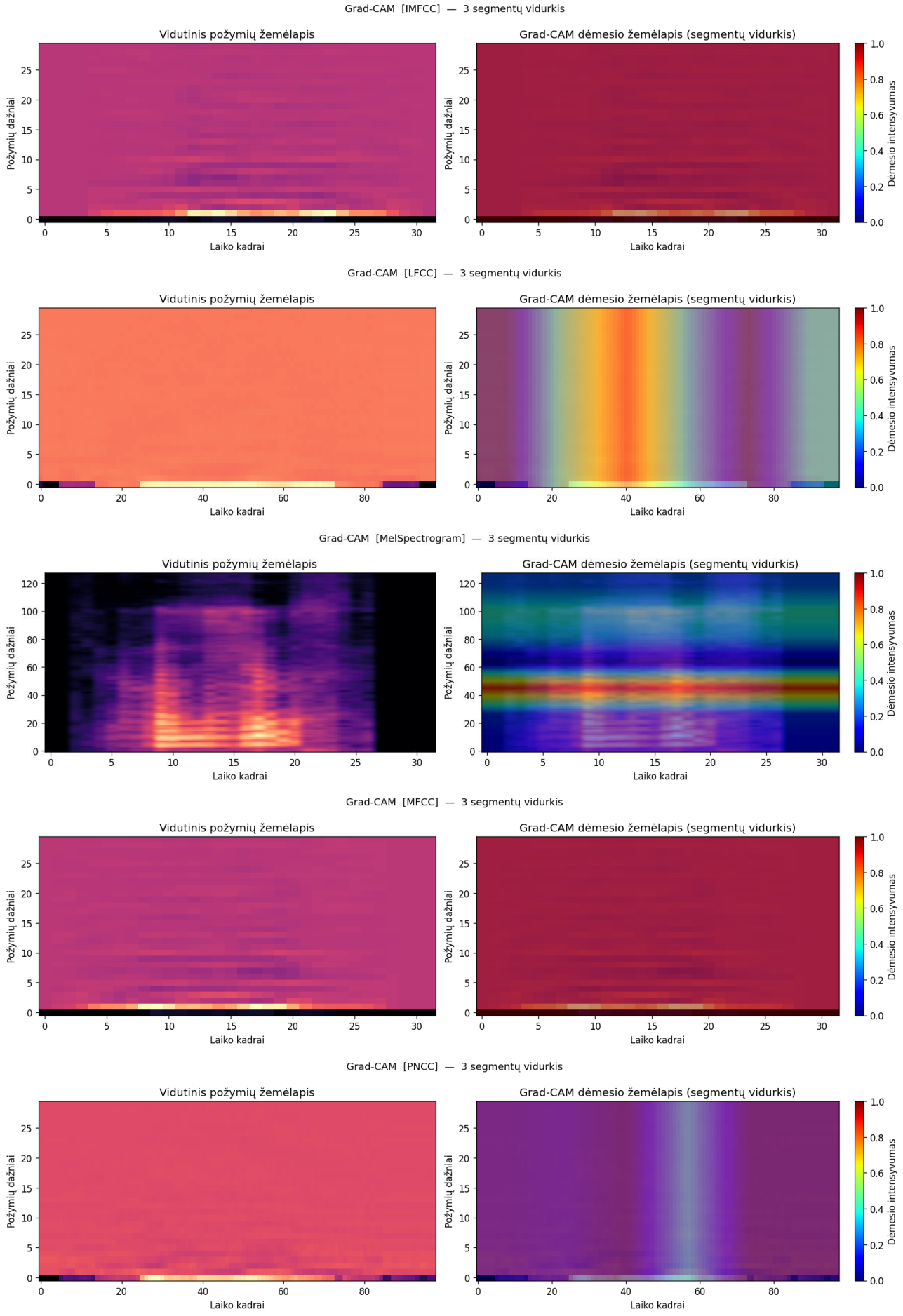


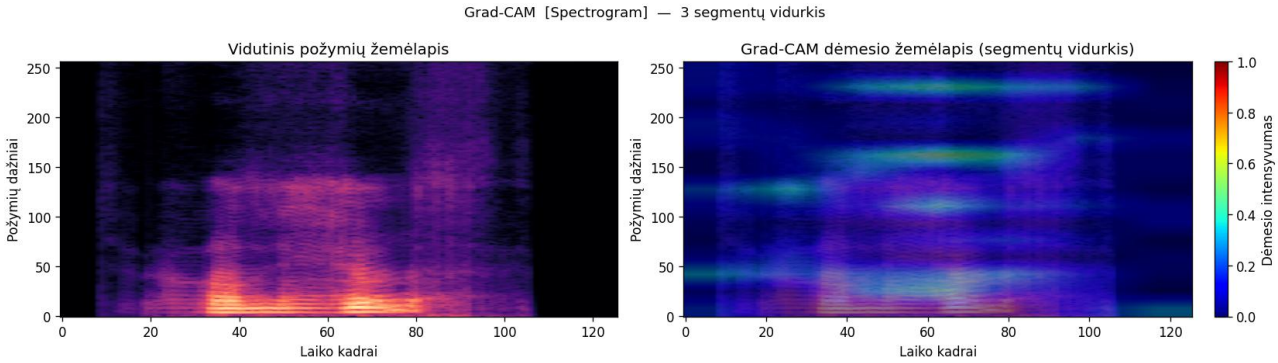




11 priedas. Klastoto garso failo gradCAM įvertinimai visiems modeliams




Grad-CAM [IMFCC] — 3 segmentų vidurkis
Vidutinis požymių žemėlapis
Grad-CAM dėmesio žemėlapis (segmentų vidurkis)
Požymių dažniai
Laiko kadrai
Dėmesio intensyvumas
Grad-CAM [LFCC] — 3 segmentų vidurkis
Vidutinis požymių žemėlapis
Grad-CAM dėmesio žemėlapis (segmentų vidurkis)
Požymių dažniai
Laiko kadrai
Dėmesio intensyvumas
Grad-CAM [MelSpectrogram] — 3 segmentų vidurkis
Vidutinis požymių žemėlapis
Grad-CAM dėmesio žemėlapis (segmentų vidurkis)
Požymių dažniai
Laiko kadrai
Dėmesio intensyvumas
Grad-CAM [MFCC] — 3 segmentų vidurkis
Vidutinis požymių žemėlapis
Grad-CAM dėmesio žemėlapis (segmentų vidurkis)
Požymių dažniai
Laiko kadrai
Dėmesio intensyvumas
Grad-CAM [PNCC] — 3 segmentų vidurkis
Vidutinis požymių žemėlapis
Grad-CAM dėmesio žemėlapis (segmentų vidurkis)
Požymių dažniai
Laiko kadrai
Dėmesio intensyvumas


Grad-CAM [Spectrogram] — 3 segmentų vidurkis
Vidutinis požymių žemėlapis
Grad-CAM dėmesio žemėlapis (segmentų vidurkis)
Požymių dažniai
Laiko kadrai
Dėmesio intensyvumas
0
20
40
60
80
100
120
0
50
100
150
200
250
0.0
0.2
0.4
0.6
0.8
1.0