Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Aplinkos garsų klasifikavimo sistemos sukūrimas

Baigiamasis bakalauro projektas

Žygimantas Marma

Projekto autorius

Doc. dr. Arūnas Lipnickas

Vadovas

Kaunas, 2022

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Aplinkos garsų klasifikavimo sistemos sukūrimas

Baigiamasis bakalauro projektas

Automatika ir valdymas (6121EX011)

Žygimantas Marma

Projekto autorius

Doc. dr. Arūnas Lipnickas

Vadovas

Doc. dr. Kastytis Ratkevičius

Recenzentas

Kaunas, 2022

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Žygimantas Marma

Aplinkos garsų klasifikavimo sistemos sukūrimas

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.

Žygimantas Marma

Patvirtinta elektroniniu būdu

TVIRTINU:
KTU Elektros ir elektronikos fakulteto
Automatikos katedros vedėjas

_____

2022 m. balandžio mėn. 29 d.

BAKALAURO BAIGIAMOJO PROJEKTO UŽDUOTIS

Išduota studentui: *Žygimantui Marmai* Grupė *EVS-8/1*

1. Darbo tema:

Lietuvių kalba: *Aplinkos garsų klasifikavimo sistemos sukūrimas*

Anglų kalba: *Development of environmental sounds classification system*

Patvirtinta 2022 m. balandžio mėn. 29 d. dekano potvarkiu Nr. V25-03-10

2. Darbo tikslas: *sukurti aplinkos garsų klasifikatorių naudojant dirbtinius neuroninius tinklus.*

3. Reikalavimai ir sąlygos: *darbas turi būti parengtas pagal KTU Elektros ir elektronikos fakulteto dekano patvirtintus „Baigiamųjų projektų rengimo ir gynimo metodinius reikalavimus“*

4. Projekto struktūra. *Turinys konkretizuojamas kartu su vadovu, atsižvelgiant į BBP pobūdį, pateiktą Metodinių reikalavimų 14 ir 15 punktuose.*

Siekiant įgyvendinti bakalauro baigiamojo projekto darbo tikslą privaloma atlikti šias užduotis:

1. išanalizuoti dirbtinio intelekto metodus garsų klasifikavimui;

2. įgyvendinti audio signalo konvertavimą į spektrogramą ir gautą rezultatą išsaugoti PNG formatu;

3. sudaryti spektrogramų klasifikatorių naudojant dirbtinius neuroninius tinklus;

4. atlikti bandymus su skirtingų tipų neuroniniais tinklais ir jų struktūromis;

5. išanalizuoti geriausius rezultatus pateikusio modelio veikimą su paaiškinamojo dirbtinio intelekto bibliotekomis.

5. Ekonominė dalis. *Jei reikia ekonominio pagrindimo; turinys ir apimtis konkretizuojama darbo eigoje kartu su vadovu.*

Nėra

6. Grafinė dalis. *Jei reikia, pateikiama schemos, algoritmai ir surinkimo brėžiniai; turinys ir apimtis konkretizuojama darbo eigoje kartu su vadovu.*

Sistemai sukurti reikiamų algoritmų veikimo paaiškinimo schemos, naudojamo duomenų rinkinio informacijos perteikimo paveikslėliai, modelių klasifikavimo tikslumo grafikai ir sumaišties matricų paveikslėliai.

5. Ši užduotis yra neatskiriama bakalauro baigiamojo projekto dalis

6. Projekto pateikimo gynimui kvalifikacinėje komisijoje terminas *Iki 2022-06-02*
(data)

Užduotį gavau: *Žygimantas Marma* *2022-03-01*
(studento vardas, pavardė, parašas) *(data)*

Vadovas: *Doc. dr. Arūnas Lipnickas* *2022-03-01*
(pareigos, vardas, pavardė, parašas) *(data)*

Marma Žygimantas. Aplinkos garsų klasifikavimo sistemos sukūrimas. Bakalauro baigiamasis projektas / vadovas Doc. dr. Arūnas Lipnickas; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): elektronikos inžinerija, inžinerijos mokslai.

Reikšminiai žodžiai: gilieji neuroniniai tinklai, konvoliucinis neuroninis tinklas, aplinkos garsų klasifikavimas, paaiškinamasis dirbtinis intelektas, MFCC.

Kaunas, 2022. 46 p.

Santrauka

Šiame bakalauro baigiamajame darbe yra sukariama aplinkos garsų klasifikavimo sistema, gebanti atskirti dažniausiai mieste pasitaikančius garsus. Darbe yra nagrinėjami aplinkos garsų klasifikavimo metodai naudojant dirbtinio intelekto algoritmus. Apžvelgiami esami šio uždavinio sprendimo metodai, egzistuojančios problemos ir mokslininkų pasiekti rezultatai. Analizuojami ir įgyvendinti garso signalo pavertimo į spektrogramas metodai. Darbe atliktiems eksperimentams buvo naudojamas viešai pasiekiamas miesto aplinkos garsų duomenų rinkinys *UrbanSound8k*. Aprašyti ir sukurti konvoliucinio ir rekurentinio ilgos-trumpos atminties tipų neuroniniai tinklai. Palyginti klasifikavimo rezultatai su skirtingų struktūrų neuroniniais tinklais ir skirtingomis spektrogramomis. Atlikus 10 - imčių kryžminį tikrinimą geriausią rezultatą užfiksavo konvoliucinis neuroninis tinklas, kuris kaip įvestį naudojo Melų spektrogramas. Pasiūlytas modelis pasiekė 74,6 % vidutinį tikslumą bei 74,3 % pasverto F1 rodiklio įvertį.

Geriausius rezultatus užfiksavusio konvoliucinio neuroninio tinklo klasifikavimo veikimas detaliai išanalizuotas ir paaiškintas naudojant paaiškinamojo dirbtinio intelekto (angl. *explainable artificial intelligence*) bibliotekas *Alibi* ir *Shap*.

Marma, Žygimantas. Development of environmental sounds classification system. Bachelor's Final Degree Project / supervisor Assoc. prof. Arūnas Lipnickas; Faculty of Electrical and Electronics Engineering, Kaunas University of Technology.

Study field and area (study field group): electronics engineering, engineering science.

Keywords: deep neural networks, convolutional neural network, classification of environmental sounds, explainable artificial intelligence, MFCC

Kaunas, 2022. 46.

Summary

In this bachelor's thesis, a classification system for environmental sounds is developed that is able to distinguish the most common sounds. The work deals with the classification of environmental sounds using different artificial intelligence algorithms. Existing methods for solving this classification problem, current implementation problems and results achieved by researchers are reviewed. Methods of converting an audio signal into a spectrogram are analysed and implemented. The publicly available urban environmental sound dataset *UrbanSound8k* was used for the experiments performed in the work. The neural networks of the convolutional and long-short memory recursive types are described and developed. The classification results were compared with neural networks of different structures and different spectrograms. In a 10 – fold cross-validation check, the best result was recorded by a convolutional neural network using Mel spectrograms as input. Suggested model achieved an 74,6 % mean accuracy and an 74,3 % weighted F1 estimate.

The working principle of the classification of the best-performing convolutional neural network was analysed and explained in detail using the explainable artificial intelligence libraries *Alibi* and *Shap*.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

Santrumpos:

MFCC – Mel‘ų dažnių kepstriniai koeficientai (angl. *Mel frequency cepstral coefficients*);

ANN – dirbtinis neuroninis tinklas (angl. *Artificial neural network*);

CNN – konvoliucinis neuroninis tinklas (angl. *Convolutional neural network*);

LSTM – ilgalaikė trumpalaikė atmintis (angl. *Long short-term memory*);

RNN – rekurentinis neuroninis tinklas (angl. *Recurrent neural network*);

STFT – trumpalaikė Furjė transformacija (angl. *Short-time Fourier transform*);

FFT – greitoji Furjė transformacija (angl. *Fast Fourier transform*);

ReLU – Ištaisyta tiesinė (angl. *Rectified Linear Unit*) aktyvacijos funkcija;

PNG – bitų masyvo formatas (angl. *Portable network graphics*);

XAI – paaiškinamasis dirbtinis intelektas (angl. *Explainable artificial intelligence*).

Terminai:

Perceptronas – smegenų modelis, padedantis tirti natūralųjį intelektą fizikinėmis ir matematinėmis priemonėmis.

Audio signalas – garso signalo reprezentacija naudojant kintantį elektros įtampos lygį arba dvejetainių skaičių seką reprezentuoti skaitmeninį signalą.

Įvadas

Gyvenant pasaulyje, apsuptame įvairių garsų iš skirtingų šaltinių, žmonių smegenys kartu su klausos sistema nuolat dirba identifikuojant kiekvieną girdimą garsą evoliucijos paremtu optimaliu būdu. Žmonių smegenys nuolat apdoroja gautus garso signalus taip įgyjant atitinkamų žinių apie supančią aplinką. Akivaizdu, kad žmonės gali lengvai atskirti garsus, tačiau kompiuterizuotoms sistemoms ši užduotis nėra tokia paprasta. Nors jau nuo šeštojo dešimtmečio[1] mokslininkai siekė skirtingų algoritmų pagalba sukurti išmaniuosius kompiuterizuotus įrenginius, kurie galėtų suprasti garsus, šie bandymai buvo nesėkmingi. Smegenų tikslumo lygį jiems pasiekti pavyko tik šiame dešimtmetyje naudojant moderniausias klasifikavimo metodikas.

Garso klasifikavimas – tai garso įrašų klausymosi ir analizės procesas. Šis procesas, taip pat žinomas kaip audio signalų klasifikavimas, yra pagrindas daugeliui šiuolaikinių dirbtinio intelekto technologijų, tokių kaip: virtualieji asistentai, automatinės kalbos atpažinimo sistemos ir teksto į kalbą vertimo aplikacijos. Garsų klasifikavimas jau daugelį metų yra didelės svarbos tyrimų sritis. Garso signalų klasifikavimo sistemos turi didžiulį panaudojimo potencialą realiame gyvenime sprendžiant įvairias problemas: išmaniųjų garso stebėjimo sistemos apsaugos srityje [1], sveikatos priežiūros srityje sprendžiant medicininės problemos [2]. Moderniausios šiuolaikinės garsų klasifikavimo sistemos yra įgyvendintos taip, kad neinvaziniu būdu, o pagal kalbos, kosulio ar širdies plakimo garsą gali nustatyti tokias ligas kaip COVID-19 [3] ar įvairias vėžinių ligų formas. Akivaizdu, kad garsų klasifikavimo panaudojimas daugelyje taikymo sričių rodo jo svarbą. Egzistuoja kelios pagrindinės garsų klasifikavimo sritys: tai muzikos žanrų klasifikavimas, automatinis kalbos atpažinimas bei aplinkos garsų klasifikavimas. Šiame tyrime dėmesys yra skiriamas būtent aplinkos garsų klasifikavimui.

Aplinkos garsų klasifikavimas yra viena iš svarbiausių problemų garsų atpažinimo srityje. Palyginus su įprastais ir struktūriškais garsais, tokiais kaip kalba ar muzika, aplinkos garsai neturi nei statinių laiko modelių, kaip melodijos ar ritmo, nei semantinių sekų, kaip fonemos. Todėl sunku rasti universalių bruožų, galinčių reprezentuoti įvairių tembrų modelius. Be to, aplinkos garsuose yra daug triukšmo ir pašalinių garsų nesusijusių su nagrinėjamu. To pasėkoje susidaro sudėtinga kompozicijos struktūra su nepastovumu, įvairumais ir nestruktūrizuotomis savybėmis. Siekiant išspręsti ankščiau išvardytas problemas, aplinkos signalų klasifikavimo užduotims atlikti buvo naudojami įvairūs signalų apdorojimo metodai ir mašininio mokymosi metodai.

Būtent pastaroji garsų klasifikavimo problema ir yra nagrinėjama šiame darbe. Projekte yra pasiūlomas konvoliucinio neuroninio tinklo modelis, kurio dėka galima klasifikuoti aplinkos garsų signalus naudojant spektrogramų paveikslėlius išsaugotus PNG formatu.

Nors mašininio mokymosi metodai leidžia išspręsti sudėtingus uždavinius, galutinis modelių veikimas nėra suprantamas. Šiai problemai spręsti mokslininkai kuria paaiškinamojo dirbtinio intelekto metodus gebančius pagrįsti modelio priimamų sprendimų pagrįstumą. Todėl šiame darbe taip pat yra nagrinėjami neuroninio tinklo priimami sprendimai pasitelkiant paaiškinamojo dirbtinio intelekto bibliotekas.

Darbo tikslas – sukurti aplinkos garsų klasifikatorių naudojant dirbtinius neuroninius tinklus.

[1] „Audrey“ kalbos atpažinimo sistema [žiūrėta 2022-05-13]. Prieiga per: https://www1.icsi.berkeley.edu/pubs/speech/audreytosiri12.pdf

Darbo tikslui pasiekti keliami šie **uždaviniai**:

1. išanalizuoti dirbtinio intelekto metodus garsų klasifikavimui;

2. įgyvendinti audio signalo konvertavimą į spektrogramą ir gautą rezultatą išsaugoti PNG formatu;

3. sudaryti spektrogramų klasifikatorių naudojant dirbtinius neuroninius tinklus;

4. atlikti bandymus su skirtingų tipų neuroniniais tinklais ir jų struktūromis;

5. išanalizuoti geriausius rezultatus pateikusio modelio veikimą su paaiškinamojo dirbtinio intelekto bibliotekomis.

1. Analitinė dalis

Šiame skyriuje yra pateikiami duomenų klasifikavimo metodai, mokslininkų naudojamos technikos pagerinti klasifikavimo tikslumą bei problemas su kuriomis yra dažniausiai yra susiduriama.

1.1. Garsų klasifikavimas

Garsų klasifikavimas apima didelę uždavinių sritį: akustinių įvykių aptikimas, muzikos žanrų atskyrimas, natūralios kalbos apdorojimas ir aplinkos garsų klasifikacija. Būtent šiame darbe yra analizuojama aplinkos garsų klasifikavimo specifiką, todėl dėmesys yra skiriamas analizuoti metodus taikomus šiems uždaviniams. Vis dėlto analizuojant garsų klasifikavimo istoriją pirmosios kalbos atpažinimo sistemos buvo orientuotos į skaičius, o ne į žodžius. Jau 1952 m. „Bell Laboratories“ sukūrė „Audrey“[1] sistemą, kuri galėjo atpažinti skaitmenų pavadinimus vienam žmogui ištarus juos garsiai.

Tuo tarpu pirmieji neuroninių tinklų panaudojimai sprendžiant garsų klasifikavimo uždavinius yra siejami su 1988 metų J. P. Lewis‘o ir P. M. Todd‘o darbais, kurie pasiūlė automatiniam muzikos kūrimui naudoti neuroninius tinklus. Lewis‘as savo darbe [4] naudojo daugiasluoksnį perceptroną savo algoritminiam požiūriui į kompoziciją. Tuo tarpu, Todd‘as naudojo rekurentinį neuroninį tinklą – RNN, kad muzika būtų nuosekliai generuojama [5]. Verta paminėti kad, šis principas, itin plačiai naudojamas ir šiais laikais.

Kalbant apie aplinkos garsų klasifikavimą, šiems uždaviniams spręsti egzistuoja tokie tradiciniai mašininio mokymo algoritmai kaip: sprendimų medžiai (angl. d*ecision trees*), atraminių vektorių mašina (angl. s*upport vector machine*), k-artimiausi kaimynai ir paslėpti Markovo modeliai (angl. *hidden Markov models*) [6]. Vis dėlto nuo tada, kai šiems klasifikavimo uždaviniams spręsti buvo pradėta taikyti neuroninius tinklus, gauti rezultatai pranoko seniau taikytus tradicinius algoritmus. Todėl šiame darbe yra analizuojami neuroniniais tinklais pagrįsti klasifikavimo algoritmai.

1.1.1. Spektrogramos

Šiuolaikinės dirbtinio intelekto sistemos garsų klasifikavimui dažnu atveju naudoja spektrogramas, kad nereikėtų dirbti su audio signalų neapdorotais duomenimis [7], todėl svarbu yra suprasti kaip jos gaunamos. Spektrograma (angl. *spectrogram*) yra vaizdinė signalo dažnių spektro, kuris kinta laikui bėgant, reprezentacija. Įprastai spektrogramoje viena ašis žymi laiką, o kita ašis – dažnį, trečiasis matmuo, nurodantis konkretaus dažnio amplitudę tam tikru metu, yra pavaizduotas kiekvieno pikselio intensyvumu arba spalva. Tai reiškia, kad ryškėjant paveikslo spalvoms, garsas stipriai susikoncentruoja aplink tuos konkrečius dažnius, tuo tarpu tamsiuose ruožuose garsas yra artimas tuščiam garsui – tylai. Taigi, spektrogramos leidžia mums vizualizuoti garso signalą, todėl galime suprasti garso formą ir struktūrą nesiklausant jo.

Detaliau analizuojant spektrogramas jos gali būti išgautos iš laiko srities audio signalo, panaudojant Furjė transformaciją. Garso signalams yra taikoma trumpalaikė Furjė transformacija (angl. *Short-time Fourier transform* – STFT*)*, kuri yra laiko ir dažnio analizės metodas, skirtas laike kintantiems signalams [8]. STFT padalina laiko srities įvesties signalą į kelis atskirtus arba persidengiančius kadrus (angl. *frames*), padaugina signalą iš lango funkcijos (angl. *window function*), o tada kiekvienam kadrui taiko greitąją Furjė transformaciją (angl. *fast Fourier transform – FFT*).

Panaudojus Furjė transformaciją yra apskaičiuojamas kiekvieno kadro dažnio spektro dydis. Tada šie spektrai (laiko grafikai) yra išdėstomi vienas šalia kito, kad susidarytų vaizdas visam signalui.

Kadangi Furjė transformacijos atliekamos kadrui slenkant per signalą, ši technika gali išmatuoti signalo dažnio turinio pokyčius laikui bėgant. Tai ir yra esminis skirtumas tarp šių dviejų transformacijų, kadangi STFT pateikia laike lokalizuotą dažnio informaciją, kai signalas kinta laike, o FFT pateikia dažnio informaciją, apskaičiuotą per visą signalo laiko intervalą.

Matematiškai diskretinė STFT aprašoma:

$$S(m,k) = \sum_{n=0}^{N-1} x(n+mH) \cdot w(n) \cdot e^{-i2\pi n\frac{k}{N}}; \quad (1)$$

čia *n* – diskretusis laikas, *x(n)* – įėjimo signalas, *w(n)* – lango funkcija, *N* – lango ilgis, *m* – laiko intervalo indeksas, ir *k* – dažnio indeksas, *H* – šuolio dydis (angl. *hop size*).

Lango funkcijai dažniausiai naudojama Hann‘o funkcija kuri aprašoma:

$$w(k) = \frac{1}{2}\left(1 - \cos(\frac{2\pi k}{K-1})\right), 0 \le k \le M-1. \quad (2)$$

Galiausiai, STFT rezultatas yra kompleksinių skaičių vektorius, todėl toliau dėmesys yra skiriamas signalo spektro galiai, kuri apskaičiuojama pagal absoliutinę reikšmę:

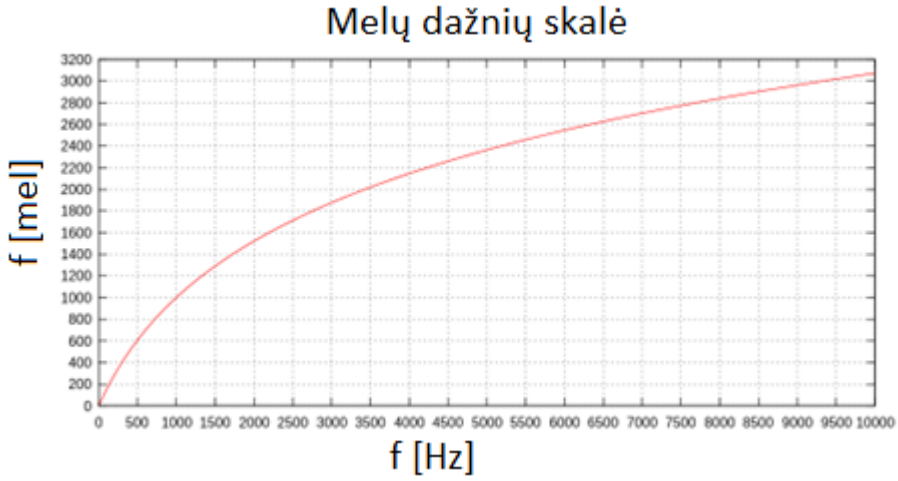
$$Y(m,k) = |S(m,k)|^2 . \quad (3)$$

1.1.2. Melų spektrogramos

STFT pagrindu linijinės garso spektrogramos idealiai tinka programoms, kuriose visi dažniai yra vienodai svarbūs. Šios spektrogramos savyje geba perteikti informaciją apie analizuojamą garsą, tačiau žmogaus garso suvokimas yra logaritminio pobūdžio. Pavyzdžiui, dauguma žmonių gali nesunkiai atskirti 200 Hz ir 300 Hz dažnio garso signalus, tačiau pasakyti skirtumą tarp 1000 ir 1100 Hz dažnio signalų jiems yra daug sunkiau. Būtent todėl dirbtinio intelekto sprendimams, siekiantiems modeliuoti žmogaus klausos suvokimą, yra naudojamos netiesinės – Melų spektrogramos (angl. M*el spectrograms).* Prieš aptariant Melų spektrogramas, pirmiausia reikia suprasti, kas yra Melų skalė ir kodėl ji yra naudinga.

Pirmiausia Melų skalė yra garso dažnio (arba natų) suvokimo skalė, kuriuos klausytojai vertina kaip esančius vienodais atstumais vienas nuo kito. Ši skalė buvo įvesta, kadangi žmonių klausa yra jautresnė garsų pasikeitimui žemų dažnių zonoje nei aukštųjų. Atskaitos taškas tarp šios skalės ir įprasto dažnio matavimo nustatomas priskiriant 1000 Melų suvokimo aukštį lygų 1000 Hz tonui. Kadangi, žmonėms yra sunkiau atskirti aukštesnių dažnių garsus, grafike (**1 pav**.) galima pastebėti, kad viršijus 500 Hz vis didesni intervalai klausytojų yra vertinami tarsi tolygūs padidėjimai. Nors oficialios formulės pakeisti dažnį į Melus nėra, dažniausiai literatūroje sutinkama formulė yra:

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) = 1127 \ln\left(1 + \frac{f}{700}\right). \quad (4)$$

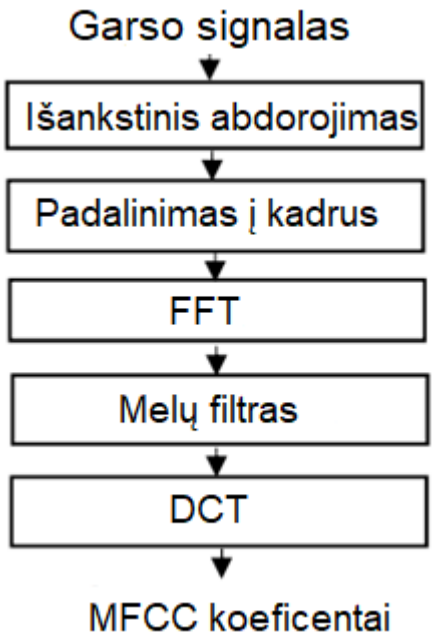


1 pav. Melų dažnių skalė

Kaip galima matyti iš grafiko (**1 pav.**), dažniai, kurie yra žemesni hercų skalėje, turi didesnį atstumą tarp jų Melų skalėje, o aukštesni dažniai hercų skalėje turi mažesnį atstumą tarp jų Melais. Būtent taip mašininio mokymosi sistemos imituoja žmogaus klausą.

1.1.3. Melų dažnių kepstriniai koeficientai

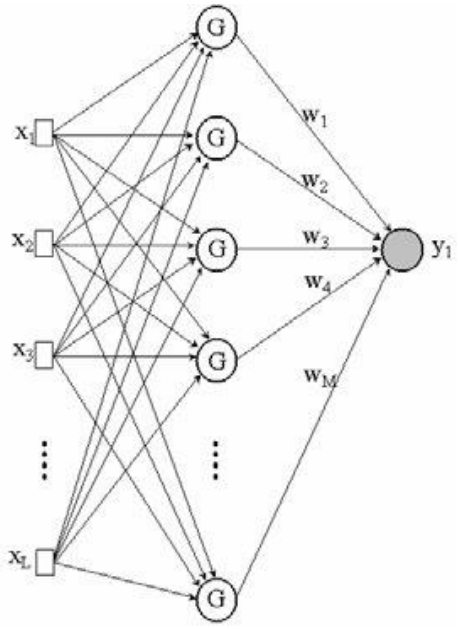
Melų dažnių kepstriniai koeficientai (angl. *Mel Frequency Cepstral Coefficients* – MFCC) yra energijos spektro realiosios kosinuso transformacijos logaritmo dalies rezultatas, išreikštas Melų dažnio skalėje. Anksčiau MFCC buvo naudojami įvairiose kalbos apdorojimo technikose, tačiau, kai įvairių garsų klasifikavimo sritis pradėjo vystytis kartu su mašininiu mokymusi, buvo nustatyta, kad MFCC gali gana gerai reprezentuoti tembrą. Standartinis Melų dažnių kepstrinių koeficientų skaičiavimo metodas naudoja diskrečiąją kosinuso transformaciją (angl. *discrete cosine transform* – DCT) Melų filtrų banko išvesties logaritminei energijai dekoduoti. Standartinis Melų dažnių kepstrinių koeficientų išgavimo algoritmas pavaizduotas paveikslėlyje (**2 pav**.).



2 pav. MFCC gavimo algoritmas [9]

1.2. Dirbtinis neuroninis tinklas

Dirbtiniai neuroniniai tinklai (angl. *Artificial neural network* – ANN), yra kompiuterinės sistemos, sukurtos pagal biologinius neuroninius tinklus, sudarančius gyvūnų smegenis. Šio tipo tinklas yra sudarytas iš mazgų (angl. *nodes*), kitaip dar vadinamųjų neuronų. Patys neuronai yra paprasti apdorojimo įrenginiai turintys svertinius ryšius, kuriais gali perduoti informaciją vienas kitam [10]. Šiomis jungtimis yra perduodami realūs skaičiai, o kiekvieno neurono išvestis apskaičiuojama pagal tam tikrą netiesinę jo įėjimų sumos funkciją. Jungčių, siejančių neuronus, stiprumas koreguojasi modeliui besimokant treniravimosi metu. Ši svorio reikšmė padidina arba sumažina signalo stiprumą jungtyje. Paprastai neuronai yra grupuojami į sluoksnius, kad skirtingi sluoksniai galėtų atlikti skirtingas įvesties transformacijas. Signalai keliauja iš pirmojo sluoksnio (įvesties sluoksnio) į paskutinįjį sluoksnį (išvesties sluoksnį), perėję tarpinius (paslėptus) sluoksnius. Idėja panaudoti dirbtinius neuroninius tinklus nėra nauja, dar 1943 metais neurofiziologas W. McCulloch'as ir matematikas W. Pitts'as parašė mokslinį straipsnį apie galimą neuroninių tinklų veikimą [11]. Siekdami apibūdinti, kaip smegenyse gali veikti neuronai, jie sumodeliavo paprastą neuronų tinklą naudodami elektros grandines.



3 pav. Standartinė dirbtinio neuroninio tinklo struktūra [12]

Naudojantis šiomis technologijomis tapo įmanoma išspręsti sudėtingas problemas, kurioms neužtenka vien paprastų matematinių skaičiavimo metodų. ANN yra taikomi klasifikavimui, grupavimui (angl. *clustering*), funkcijų aproksimavimui, prognozavimui, optimizavimui, medicininei diagnozei, finansinėms prognozėms, intelektinei paieškai, ir kitoms sritims [13].

Norint panaudoti dirbtinį neuroninį tinklą sprendžiant realias užduotis, pirmiausia būtina jį išmokyti. Sukūrus tam tikros struktūros tinklą, jį galima apmokyti. Norint pradėti šį procesą, pradiniai svoriai parenkami atsitiktine tvarka. Tada prasideda mokymosi procesas. Yra dvi pagrindinės mokymosi metodikos – prižiūrimas (angl. *supervised*) ir neprižiūrimas (angl. *unsupervised*) mokymasis. Prižiūrimas mokymas apima procesą, kurio metu norimas tikslumas pasiekiamas arba rankiniu būdu įvertinant tinklo našumą arba pateikiant norimus išėjimus kartu su įvesties vertėmis. Neprižiūrimas mokymas yra toks metodas, kai tinklas turi suvokti įvesties duomenis be pašalinės pagalbos. Didžioji dalis dirbtinio intelekto modelių naudoja prižiūrimus mokymus.

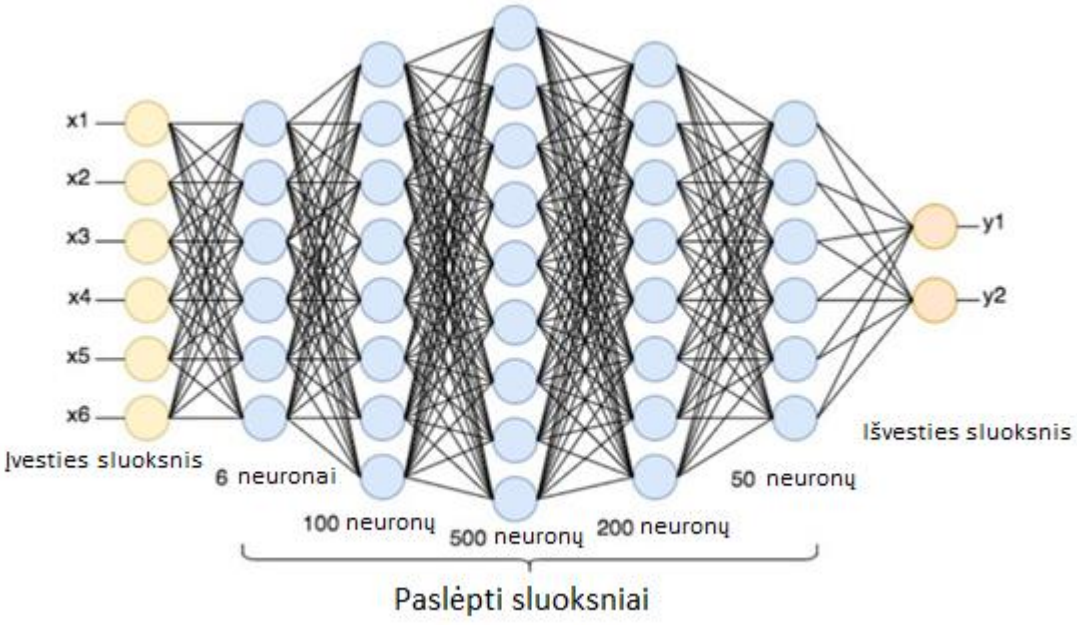
Prižiūrimo mokymosi metu tinklui pateikiami tiek įėjimų, tiek išėjimų duomenys. Tada modelis apdoroja įvestis ir lygina savo gautus išėjimus su pateiktomis išvesties duomenų vertėmis. Tada gaunama paklaida yra perduodama atgal į sistemą, kad neuroninis tinklas galėtų koreguoti sluoksnių svorius. Šis procesas kartojasi nuolat, kadangi treniravimo metu svoriai yra keičiami siekiant gauti mažiausią paklaidą. Duomenų rinkinys, įgalinantis mokymą, vadinamas mokymo rinkiniu (angl. *training set*), šis duomenų rinkinys yra apdorojamas daug kartų – kol vis tikslinami neuroninio tinklo sluoksnių svoriai.

Pastebima problema, kad kai kurie neuroniniai tinklai niekada nepasiekia norimo tikslumo – jie neišmoksta. Tam gali būti įvairių priežasčių. Tai gali atsitikti dėl to, kad įvesties duomenyse nėra konkrečios informacijos, iš kurios turėtų būti gaunama norima išvestis. Tinklai taip pat nekonverguoja, jei nėra pakankamai duomenų visapusiškam mokymuisi. Idealiu atveju turėtų būti pakankamai duomenų, kad dalį iš jų būtų galima būtų naudoti modelio testavimui. Egzistuoja problema, kad tinklas, turintis sudėtingą struktūrą ir daugelį sluoksnių (gilusis neuroninis tinklas), yra pajėgus tiesiog įsiminti duomenis. Norint stebėti tinklą ir nustatyti, ar sistema paprasčiausiai įsimena įvesties duomenis, prižiūrimo mokymosi metu būtina pasilikti duomenų rinkinį, kuris bus naudojamas sistemai testuoti po to, kai ji buvo apmokyta.

Jei tinklas tiesiog negali išspręsti problemos, programuotojas turi peržiūrėti įvesties ir išvesties duomenis, sluoksnių skaičių, neuronų skaičių sluoksnyje, ryšius tarp sluoksnių, sumavimo, perdavimo ir mokymo funkcijas arba net pačius pradinius svorius. Kita programuotojo sprendimų dalis yra reglamentuojant mokymo taisykles. Egzistuoja daug algoritmų, skirtų įgyvendinti adaptyvų grįžtamąjį ryšį, kuris yra reikalingas norint reguliuoti svorius treniravimo metu. Labiausiai paplitęs būdas yra sklidimas atgaline klaida (angl. *backward-error propagation*) labiau žinomas kaip sklidimas atgal (angl. *back-propagation*). Šie pokyčiai, reikalingi sėkmingam tinklui sukurti, vadinami tinklo derinimu.

1.2.1. Gilusis neuroninis tinklas

Sprendimui naudojamas neuroninis tinklas tampa gilesnis (angl. *deeper*), kai sprendžiamos užduotys tampa sunkesnės. Gilusis neuroninis tinklas reprezentuoja mašininio mokymosi (angl. *machine learning*) tipą, kai sistema naudoja daugelį neuronų sluoksnių, kad gautų aukšto lygio funkcijas iš turimos įvesties informacijos. Taigi, gilusis neuroninis tinklas yra neuroninio tinklo tipas, kuris turi ne vieną, o žymiai daugiau paslėptų sluoksnių. Tobulėjant kompiuterinio skaičiavimo galimybėms, taip pat vis plečiantis duomenų saugojimo talpos galimybėms, pastaraisiais metais sparčiai vystėsi giliojo mokymosi metodai [14]. Šio tipo tinklai yra pajėgūs išspręsti gana sudėtingus vaizdo segmentavimo (angl. *segmentation*), duomenų klasifikavimo, garso ir kitų signalų atpažinimo, spalvų atkūrimui nuotraukose [15] bei kitų tipų uždavinius.



4 pav. Giliojo neuroninio tinklo struktūros pavyzdys [16]

Aukštas DNN tikslumo lygis atsiranda dėl jų gebėjimo išskirti aukšto lygio požymius (angl. *features*) tiesiai iš neapdorotų įvesties duomenų [17]. Naudojantis giliuoju mokymusi su dideliu duomenų kiekiu, galima efektyviai atpažinti abstrakčius įvesties duomenų požymius. Tokia mokymosi metodika leidžia pasiekti ženkliai geresnius rezultatus, nei ANN mokantis iš jau atrinktų duomenų požymių.

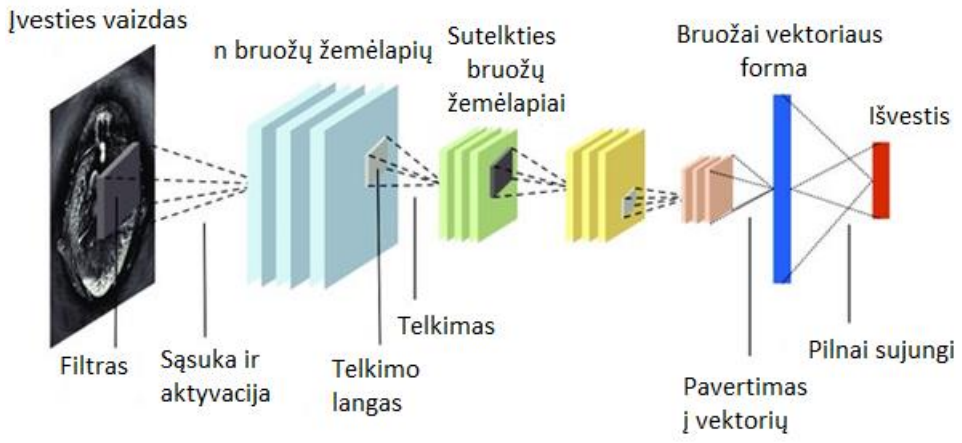
1.3. Konvoliucinis neuroninis tinklas

Problema su dirbtiniais neuroniniais tinklais sprendžiant vaizdų klasifikavimo uždavinius yra ta, kad nuotrauka, sudaryta iš pikselių matricos (dirbant su spalvotais vaizdais trimatės matricos), turi būti paversta į vienmatį vektorių, t.y. skaičių eilutę. Šio pavertimo metu yra prarandami nuotraukoje esantys erdviniai požymiai, kadangi pikselių, sudarančių vaizdą, išsidėstymas yra itin svarbus klasifikavimo ir segmentavimo uždaviniuose. Taip pat, keičiant nuotrauką į vektorių treniruojamų parametrų skaičius padidėja eksponentiškai. Akivaizdu, kad šiems parametrams laikyti ir apdoroti reikia didesnio kiekio kompiuterinių resursų. Norint išvengti šių problemų naudojamas konvoliucinis neuroninis tinklas (angl. *convolutional neural network* – CNN).

CNN jau buvo naudojami nuo praeito amžiaus devintojo dešimtmečio, tačiau tik pastaruoju metu jie pralenkė daugumą tradicinių klasifikatorių. Šio tipo dirbtiniai neuroniniai tinklai dažniausiai taikomi kompiuterinės regos srityje: vaizdų atpažinimui, objektų radimui ir identifikavimui nuotraukose, vaizdų segmentacijai ir natūralios kalbos atpažinimui. Pats pavadinimas „konvoliucinis neuroninis tinklas“ rodo, kad tinklui mokantis naudojama matematinė operacija, vadinama sąsūka (angl. *convolution*). CNN yra specializuotas neuroninių tinklų tipas, kuriame konvoliucija naudojama vietoj bendros matricos daugybos bent viename iš jų sluoksnių [18].

Nors šio tipo tinklas turi panašią struktūrą į ANN, tinklas susideda iš įėjimo sluoksnio, paslėptų sluoksnių ir išėjimo sluoksnio. Esminis skirtumas yra tas, kad modelio įvestis yra ne duomenų vektorius, o dvimatė (ar aukštesnio laipsnio) matrica arba kitaip – nuotrauka. Šias matricas sumažinti taip pat standartiškai yra naudojami telkimo (angl. *pooling*) sluoksniai, o vienuose iš paskutinių

sluoksniuose bruožus pavertus į vienmatį vektorių yra naudojami pilnai sujungti (angl. *fully connected*) sluoksniai.



5 pav. Tipinė konvoliucinio neuroninio tinklo struktūra [19]

Kaip vieną svarbiausių mokslinių darbų CNN srityje reikia paminėti A. Krizhevsk'io bendradarbiaujant su kitais mokslininkais 2012 metais pasiūlytą CNN architektūrą *AlexNet* [20]. Autoriai dalyvavo bei laimėjo „ImageNet Large Scale Visual Recognition Challenge" varžybas[2], o antroje vietoje likusį modelį aplenkė daugiau nei 10,8 procentinių punktų pagal geriausių penkių spėjimų klaidų rodiklį (angl. *top-5 error rate*). Pačio modelio architektūra yra panaši į *LeNet-5*, tačiau turinti dar gilesnę struktūrą, kuri buvo labai svarbi jo aukštam tikslumui.

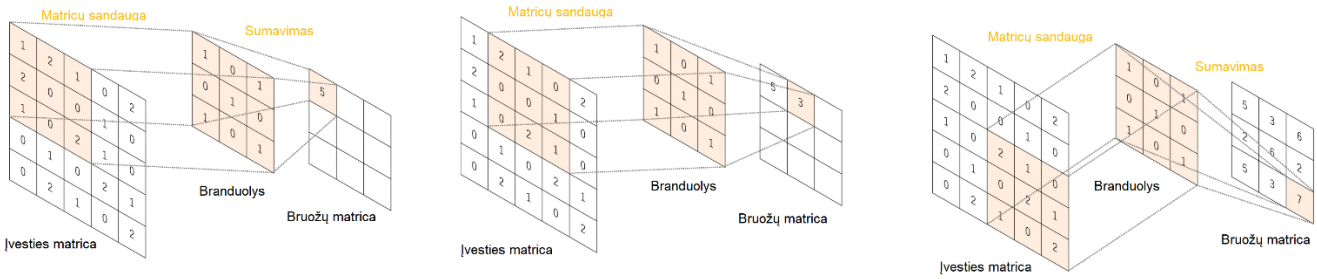
Toks aukštas modelio klasifikavimo tikslumas padarė ženklią įtaką būsimiems darbams, kurie stengėsi pagerinti *AlexNet* modelio veikimą [21]. Keturi žymūs šio tipo darbai yra: *ZFNet*[3] , *VGGNet*[4], *GoogleNet*[5] ir *ResNet*[6] . Architektūriniu požiūriu matoma, kad mokslininkų siūlomi tinklai vis gilėja. *ResNet*, kuris laimėjo „ILSVRC 2015" čempionatą, yra maždaug 20 kartų gilesnis nei *AlexNet* ir 8 kartus gilesnis nei *VGGNet*. Didinant gylį, tinklas geba labiau priartėti prie tikslo funkcijos, padidėjant netiesiškumui ir išgaunant reikšmingesnius bruožus. Tačiau tai taip pat padidina tinklo sudėtingumą, todėl jį sunkiau optimizuoti ir lengviau permokyti (angl. *overfit*).

1.3.1. Dvimatis konvoliucinis sluoksnis

Konvoliucijos sluoksnis yra pagrindinė CNN architektūros sudedamoji dalis, kuri atlieka bruožų išskyrimą, kurią paprastai sudaro tiesinių ir netiesinių operacijų derinys, t. y. konvoliucijos operacija ir aktyvinimo funkcija. Konvoliucija (angl. *convolution*) arba sąsūka yra specializuotas tiesinės matematinės operacijos tipas, naudojamas ypatybėms išgryninti, kai įvestį apdoroja nedidelis skaičių masyvas, vadinamas branduoliu (angl. *kernel*) [22]. Čia minimas branduolys yra tiesiog maža svorių matrica. Sąsūkos operacijos metu yra atliekama elementų sandauga tarp kiekvieno branduolio elemento ir persidengiančio įvesties matricos elemento, tada gauti rezultatai yra sumuojami į vieną išvesties pikselį. Šis veiksmas yra kartojamas branduoliui slenkant per įvesties matricą iš kairės į

[2] „ImageNet" varžybos [žiūrėta 2022-05-13]. Prieiga per: https://image-net.org/challenges/LSVRC/2012/results.html
[3] *ZFNet* modelis [žiūrėta 2022-05-13]. Prieiga per: https://paperswithcode.com/method/zfnet
[4] *VGGNet* modelis [žiūrėta 2022-05-13]. Prieiga per: https://paperswithcode.com/method/vgg
[5] *GoogleNet* modelis [žiūrėta 2022-05-13]. Prieiga per: https://paperswithcode.com/method/googlenet
[6] *ResNet* modelis [žiūrėta 2022-05-13]. Prieiga per: https://paperswithcode.com/method/resnet

dešinę ir per visas galimas eilutes. Naudojant šį signalų apdorojimo metodą yra išgaunama nauja, mažesnių dimensijų matrica, kaip pavaizduota paveikslėlyje (**6 pav.**).



6 pav. Konvoliucijos pavyzdys naudojant 3x3 dydžio branduolį [22]

Paprasčiausiu dvimačiu atveju konvoliucija išreiškiama:

$$G[m,n] = (f * h)[m,n] = \sum_j \sum_k h[j,k] f[m-j, n-k]\,; \tag{5}$$

čia f – įvestis, o h yra branduolys.

Naudojant konvoliucinį neuroninį tinklą yra svarbu tinkamai parinkti aktyvacijos funkciją. Egzistuoja trys pagrindiniai netiesinių aktyvavimo funkcijų tipai: *Sigmoid*, *Tanh* ir *ReLU*. Kompiuterinėje regoje, siekiant duomenų normalizacijos, populiariausia yra *ReLU* (angl. *Rectified Linear Unit*) aktyvavimo funkcija arba jos modifikacijos (*LeakyReLu*, *eReLu*). Ši funkcija yra įvardijama, kaip geriausiai tinkanti neuroninio tinklo mokymo efektyvumui, o matematiškai ji paprasčiausiai išreiškiama kaip:

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}. \tag{6}$$

Akivaizdu, kad ši funkcija gautas neigiamas įėjimo reikšmės, po sąsūkos operacijos, pakeičia į 0. Būtent netiesiškos aktyvavimo funkcijos naudojimas yra privalomas paslėptuose neuroniniuose sluoksniuose, kad modelis gebėtų išmokti sudėtingus įvesties bruožus.

1.4. Rekurentinis neuroninis tinklas

Rekurentinis neuroninis tinklas – RNN yra dirbtinio neuroninio tinklo tipas, kuriame naudojami nuoseklūs arba laiko eilučių duomenys. Šie gilaus mokymosi modeliai, yra labai veiksmingi modeliuojant sekos duomenis, tokius kaip tekstas ir kalba [23]. RNN turi vidinę būseną, kurioje saugomos ankstesnės reikšmės, kurios kartu su dabartine įvesties reikšme naudojamos rezultatui apskaičiuoti [24]. Nors tradiciniuose giliuosiuose neuroniniuose tinkluose daroma prielaida, kad įėjimai ir išėjimai yra nepriklausomi vienas nuo kito, RNN išvestis priklauso nuo ankstesnių sekos elementų. Šio tipo tinklai turi grįžtamąjį ryšį, kai paskutinė paslėpta būsena (sluoksnis) yra įvestis į sekančią būseną. Būsenų atnaujinimą galima apibūdinti taip:

$$h_t = \sigma(W w_t + U h_{t-1} + b); \tag{7}$$

čia $\mathbf{x}_t \in R^M$ ir $\mathbf{h}_t \in R^N$ yra atitinkamai įvestis ir paslėpta būsena laiko momentu t. $W \in R^{N \times M}$, $U \in R^{N \times N}$ ir $\mathbf{b} \in R^N$ yra atitinkamai dabartinės būsenos įvesties svoriai, pasikartojančios įvesties svoriai ir

neuronų paklaida (angl. *bias*). σ yra neuronų aktyvinimo funkcija, o N yra neuronų skaičius šiame RNN sluoksnyje [25].

Priešingai nei ankstesniuose skyriuose aprašyti tiesinio sklidimo metodai, rekurentiniai neuroniniai tinklai turi bent vieną grįžtamąjį (uždarojo ciklo) ryšį, todėl RNN gali atlikti sekos atpažinimo, sekos atkūrimo ir teksto interpretavimo veiklą. Analizuojant rekurentinio tinklo architektūrą galime pastebėti įvairių išdėstymo formų. Grįžtamasis ryšys gali būti gaunamas iš tiesioginio sklidimo tinklo išvesties neuronų į įvesties sluoksnį, taip pat grįžtamasis ryšys taip pat gali įgyvendintas iš paslėptų tinklo neuronų į įvesties sluoksnyje esančius neuronus [26].

Vis dėlto RNN yra ypač jautrūs nykstančio gradiento problemai: būsenos, kurios yra per toli nuo dabartinės būsenos, nieko neprisideda prie mokymosi, tačiau tinklas turi išmokti ilgalaikes duomenų priklausomybes. Šios problemos sprendimas grindžiamas ilgalaikės – trumpalaikės atminties (angl. *Long–short term memory* – LSTM) koncepcija [24].

1.4.1. Ilgalaikės – trumpalaikės atminties modelio tinklas

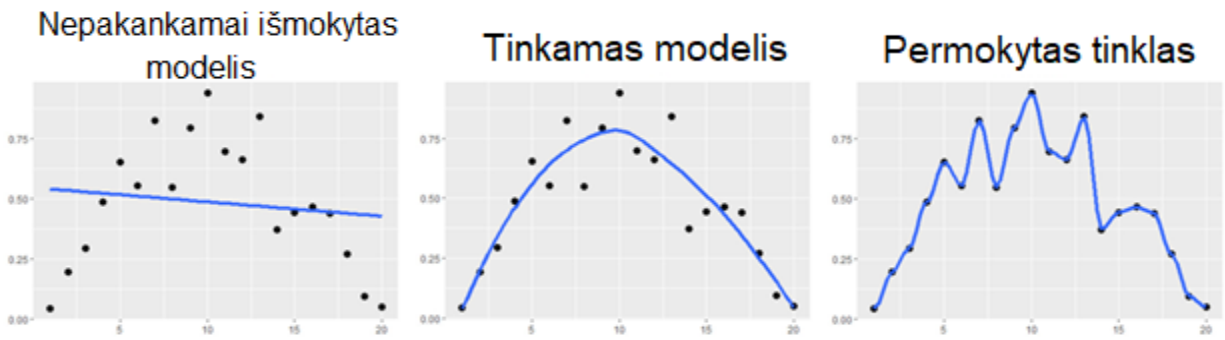
1997 metais S. Hochreiter‘is ir J. Schmidhuber‘as pasiūlė pažangų RNN modelį, vadinamą ilgalaike trumpalaike atmintimi (angl. *Long short-term memory – LSTM*) [27]. LSTM yra specialiai sukurti siekiant išvengti ilgalaikės priklausomybės problemos. Pavyzdžiui, du žodžiai, svarbūs norint teisingai klasifikuoti filmo apžvalgoje esančias nuotaikas, ilgame sakinyje gali būti atskirti daugybe žodžių. Nuotaikų klasifikavimo modelis, naudojant įprastą RNN, turės sunkumų užfiksuoti tokią ilgalaikę žodžių priklausomybę. Įprastas RNN yra naudingas, kai priklausomybė tarp žodžių ar sveikųjų skaičių sekoje yra tiesioginė arba kai du svarbūs žodžiai yra vienas šalia kito.

Kalbant apie LSTM tinklo struktūrą, ją sudaro atminties blokai, kitaip vadinamos ląstelės (angl. *cells*), kurie yra sujungiami nuosekliai, kad būtų galima valdyti informacijos srautą. Kiekviena ląstelė naudoja tris: užmiršimo, įvesties ir išvesties vartus, kad valdytų informacijos pridėjimo arba pašalinimo iš tinklo procesą. Tokia struktūra leidžia perduoti esamos ląstelės būseną ir paslėptą būseną į sekančią ląstelę [28]. Šis veikimo principas leidžia perduoti informaciją nenaudojant didelio kiekio skaičiavimų, o tai sudaro galimybes greitesniam atgaliniam sklidimui ir senesnių žingsnių informacijos įsiminimui.

1.5. Modelio persimokymo problema

Persimokymas (angl. *overfitting*) yra duomenų mokslo konceptas, kuris atsiranda, kai statistinis modelis tiksliai atitinka mokymo duomenis, tačiau negali tiksliai veikti su nematytais (testavimo) duomenimis. Ši problema yra labai aktuali, nes modelio gebėjimas apibendrinti (angl. *generalize*) naujus duomenims leidžia mums realiame gyvenime naudoti mašininio mokymosi algoritmus prognozėms, duomenims klasifikuoti ir spręsti kitas problemas.

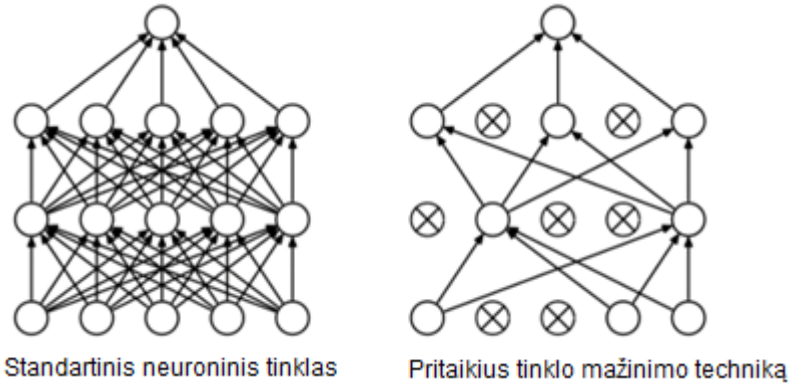
Modelis persimoko, kai akcentuojasi į bruožus (angl. *features*), kylančius dėl duomenų triukšmo ar dispersijos, o ne į esminius bruožus, kurie leistų atlikti numatytas užduotis. Todėl persimokymas paprastai pasižymi testavimui naudotų duomenų tikslumo trūkumu. Tai gali atsitikti, kai modelis yra per ilgai treniruojamas su pavyzdiniais duomenimis arba kai modelis yra per sudėtingos struktūros. Šie veiksniai lemia, kad neuroninis tinklas gali pradėti mokytis signalo triukšmo arba nesusijusios informacijos duomenų rinkinyje. Kadangi persimokymo problema yra būdinga daugeliui mašininio mokymosi paradigmų, verta išsamiai aptarti būdus kaip ją galima spręsti.



7 pav. Modelio prisitaikymo prie treniravimo duomenų pavyzdys [29]

1.5.1. Tinklo mažinimo technika

Tinklo mažinimas arba parametrų išmetimas (angl. *dropout*) yra neuroninių tinklų mokymo algoritmas, kuris remiasi stochastiniu neuronų pašalinimu treniravimo metu, kad būtų išvengta tarpusavio priklausomybės tarp neuronų porų [30]. Taip yra stengiamasi, kad kiekvienas paslėpto sluoksnio neuronas būtų reikšmingas ir gebėtų identifikuoti naudingus bruožus, o ne pasikliauti kitais paslėptais neuronais, kad ištaisytų klaidas. Ši technika buvo pasiūlyta 2014 metais siekiant išvengti persimokymo, naudojant giliuosius neuroninius tinklus mokymuisi [31]. Šį išmetimą galime interpretuoti kaip tikimybę, kad tam tikras sluoksnio neuronas bus paliktas mokymuisi. Kur 0 reiškia, kad nėra iškritimo, o 0,5 reiškia, kad 50% sluoksnio neuronų yra ignoruojami. Išmesdami neuroną, turima omenyje jo pašalinimą iš tinklo kartu su tiek įeinančiomis tiek ir išeinančiomis jungtimis.



8 pav. Tinklo mažinimo technikos vizualizacija [31]

1.5.2. Ankstyvas mokymo stabdymas

Ankstyvas mokymo stabdymas (angl. *early stopping*) yra plačiai naudojamas metodas, skirtas išvengti prastų apibendrinimo rezultatų, treniruojamas sudėtingas modelis naudojant gradientu pagrįstą optimizavimą [32]. Ši technika yra plačiai naudojama, nes ją paprasta suprasti ir įgyvendinti, taip pat daugeliu atveju ši technika yra pranašesnė už alternatyvius reguliavimo metodus [33]. Nepageidaujamas modelio persimokymo poveikis paprastai pašalinamas anksti sustabdant optimizavimo procesą, tai reiškia, kad modelio treniravimas sustabdomas, jei yra tenkinamas vartotojo sukurtas ankstyvo sustabdymo kriterijus. Didelė tikimybė, kad sustabdžius mokymą būtent šiame taške, modelis turės geriausias apibendrinimo savybes. Šis metodas gali būti naudojamas arba interaktyviai, t.y. remiantis žmogaus sprendimu, arba automatiškai – remiantis kokiu nors formaliu

stabdymo kriterijumi [33]. Dažniausiai yra pasirenkami automatiniai stabdymo būdai, kurie būna pateikti naudojamosiose mašininio mokymosi bibliotekose tokiose kaip *Tensorflow* ar *PyTorch*.

Metodo esmė yra apmokant tinklą, kiekvieną kartą pasiekus geresnį rezultatą (vertinant pagal nuostolių funkciją arba modelio tikslumą) išsaugoti tarpinį tinklo modelį. Jeigu pastebima, kad modelio rezultatas su testavimo duomenimis jau kelintą mokymosi epochą negerėja, tolimesnis mokymas yra nutraukiamas. Tada galutiniam tinklui naudojami svoriai, su kuriais buvo pasiektas geriausias rezultatas. Kaip minėta anksčiau, šiuo metodu siekiama pristabdyti mokymą, kol modelis pradeda mokytis duomenų triukšmo. Verta paminėti, kad naudojant šį metodą treniruočių procesas gali būti sustabdytas per anksti, o tai sukels priešingą – nepakankamo pritaikymo (angl. *underfitting*) problemą. Todėl naudojant ankstyvo stabdymo techniką yra svarbu rasti optimalų tašką tarp persimokymo ir nepakankamo pritaikymo.

1.5.3. Imties normalizavimas

Imties normalizavimas (angl. *batch normalization*) yra metodas padaryti giliųjų neuronų tinklų mokymosi procesą greitesniu, stabilesniu ir sumažinti modelio prisitaikymo prie treniravimo duomenų problemą. Paketų normalizavimo sluoksnis adaptyviai normalizuoja kito sluoksnio įvesties reikšmes, sumažindamas persimokymo riziką, taip pat pagerindamas gradiento srautą tinkle, būtent todėl yra pasiekimas didesnis mokymosi greitis ir sumažinama priklausomybė nuo pradinių paskirtų reikšmių [34]. Metodą sudaro aktyvavimo vektorių normalizavimas paslėptuose sluoksniuose, naudojant pirmąjį ir antrąjį imties statistinius momentus (vidurkį ir dispersiją). Šią techniką 2015 m. pasiūlė S. Ioff'as ir C. Szegedy'as [35]. Detalų matematinį metodo paaiškinimą galima rasti pirminiame autorių darbe [35].

1.5.4. K-imčių kryžminis tikrinimas

Norint išvengti duomenų prisitaikymo problemos taip pat yra svarbu teisingai parinkti testavimo ir treniravimo duomenis. Vietoje įprasto duomenų padalijimo taip pat galima naudoti k-imčių kryžminį tikrinimą [36].

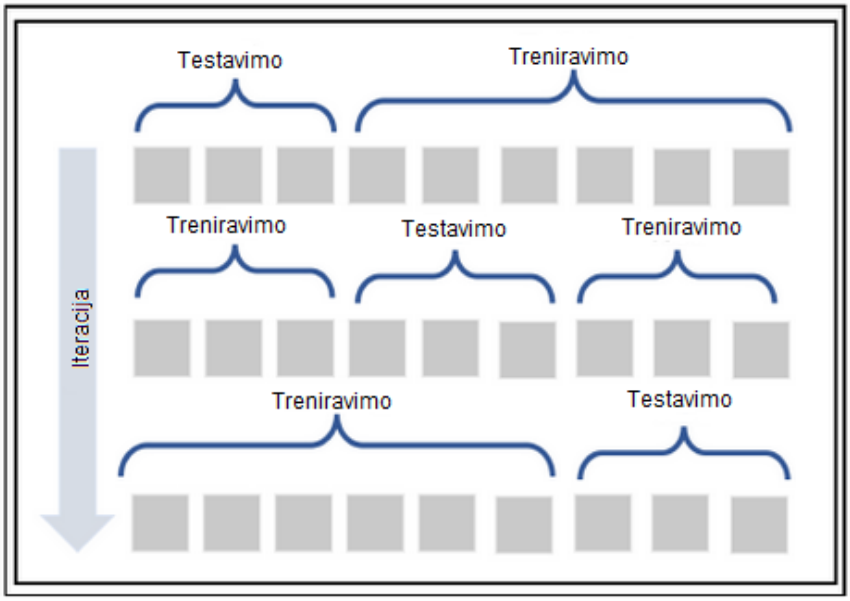
K-imčių kryžminio patvirtinimo procedūra standartiškai apima šiuos tris žingsnius:

1. Mokymo duomenų rinkinį padalijame į k-ą dalių.

2. Pirmosios k – 1 imtys (angl. *folds*) naudojamos modeliui treniruoti, o k–oji imtis naudojama kaip testavimo rinkinys.

3. Modelis yra įvertinamas naudojant vieną imtį ir gauti analizuojami rodikliai yra išsaugomi.

Antro ir trečio punkto procesai kartojami kiekvienai imčiai suteikiant galimybė būti panaudotai kaip testavimo rinkiniu. Galiausiai iš viso yra apmokyti ir įvertinti k modeliai, o galutinis tinklo našumas apskaičiuojamas kaip šių gautų rodiklių aritmetinis vidurkis. Šis algoritmas leidžia gauti labiau tikėtinus modelio rezultatus, kuriems mažesnę įtaką gali padaryti netinkamas treniravimo/testavimo duomenų paskirstymas. Metodas vizualiai pavaizduotas **9 pav**. naudojant 3-imčių kryžminį patikrinimą.



9 pav. K-imčių kryžminės patikros vizualizacija [36]

1.5.5. Duomenų rinkinio netolygumo problema

Klasių disbalansas yra klasikinė problema, su kuria galima susidurti sprendžiant mašininio mokymosi problemas. Klasių disbalanso atveju kiekviena klasė nėra vienodai atstovaujama duomenų taškų skaičiumi. Nustatyta, kad klasių disbalansas gali turėti ženklų neigiamą poveikį klasifikatorių mokymui, taip pat ši problema turi įtakos ir modelio gebėjimui apibendrinti (angl. *generalize*) testavimo rinkinį [37]. Siekiant išvengti šios disbalanso problemos yra pasiūlyta klasių svorių technika.

Klasės svoriai koreguoja modelio nuostolių funkciją (angl. *loss function*) taip, kad už neteisingą mažesnės klasės klasifikavimą būtų labiau baudžiama nei padarius klaidą su klase, turinčia daugiau įrašų. Šis metodas paprastai padeda pagerinti modelio tikslumą iš naujo subalansuojant klasių pasiskirstymą. Tačiau svarbu pažymėti, kad klasės svoriai nesukuria naujų duomenų įrašų ir negali kompensuoti ženklaus duomenų trūkumo.

Kiekvienai klasei svorius galima apskaičiuoti pagal formulę:

$$w_i = \frac{n_s}{C \cdot n_i}; \tag{8}$$

čia w_i i–tosios klasės svoris, n_s – įrašų kiekis treniravimo duomenų rinkinyje, C – klasių skaičius, n_i – i-tosios klasės įrašų kiekis treniravimo duomenų rinkinyje.

1.5.6. Modelio (algoritmų) įvertinimas

Mašininio mokymosi klasifikavimo algoritmo tikslumas (angl. *accuracy*). yra vienas iš būdų įvertinti, kaip dažnai algoritmas teisingai klasifikuoja duomenų rinkinį. Tikslumas yra teisingai klasifikuotų duomenų kiekio santykis su visų duomenų kiekiu.

Klasifikuojant duomenų rinkinį, kurį sudaro daugiau nei dvi klasės, kiekvienos klasės tikslumas apskaičiuojamas pagal (9) formulę.

$$Tikslumas(c_i) = \frac{TP(c_i) + TN(c_i)}{TP(c_i) + TN(c_i) + FP(c_i) + FN(c_i)}; \quad (9)$$

čia c_i – i-toji klasė, $TP(c_i)$ – teisingas teigiamas (angl. *true positive*) reiškia, kad modelis teisingai priskyrė klasę, $TN(c_i)$ – teisingas neigiamas (angl. *true negative*) modelis teisingai priskyrė stebėjimą neigiamai klasei, $FP(c_i)$ – klaidingas teigiamas (angl. *false positive*) reiškia, kad modelis stebėjimą klasifikavo kaip teigiamą, nors iš tikrųjų jis buvo neigiamas, $FN(c_i)$ – klaidingas neigiamas (angl. *false negative*) reiškia, kad modelis neteisingai priskyrė stebėjimą kaip neigiamą, nors jis turėjo būti klasifikuojamas kaip teigiamas.

Tuo atveju, kai duomenų rinkinys pasižymi duomenų netolygumu, tikslumo rodiklis gali neparodyti neteisingai klasifikuojamos klasės, jei jos įrašų kiekis testavimo rinkinyje yra ženkliai mažesnis palyginti su kitomis klasėmis. Todėl svarbu naudoti netik tikslumo rodiklį, bet ir F1 rodiklį (angl. *F-score* arba *F-measure*), kuris apskaičiuojamas pagal (10) formulę.

$$F1\ (c_i) = \frac{2 \cdot Preciziškumas(c_i) \cdot Atšaukimas(c_i)}{Preciziškumas(c_i) + Atšaukimas(c_i)}; \quad (10)$$

$$Preciziškumas(c_i) = \frac{TP(c_i)}{TP(c_i) + FP(c_i)}; \quad (11)$$

$$Atšaukimas(c_i) = \frac{TP(c_i)}{TP(c_i) + FN(c_i)}; \quad (12)$$

F1 rodiklis yra apskaičiuojamas pagal testo preciziškumą (angl. *precision*) ir atšaukimą (angl. *recall*). Preciziškumas yra teisingų teigiamų rezultatų skaičius, padalytas iš visų teigiamų rezultatų, įskaitant ir neteisingai identifikuotus. Tuo tarpu atšaukimas yra teisingų teigiamų rezultatų skaičius, padalytas iš visų įrašų, kurie turėjo būti klasifikuojami kaip teigiami, skaičiaus. Atšaukimas binarinėje klasifikacijoje taip pat žinomas kaip jautrumas (angl. *sensitivity*).

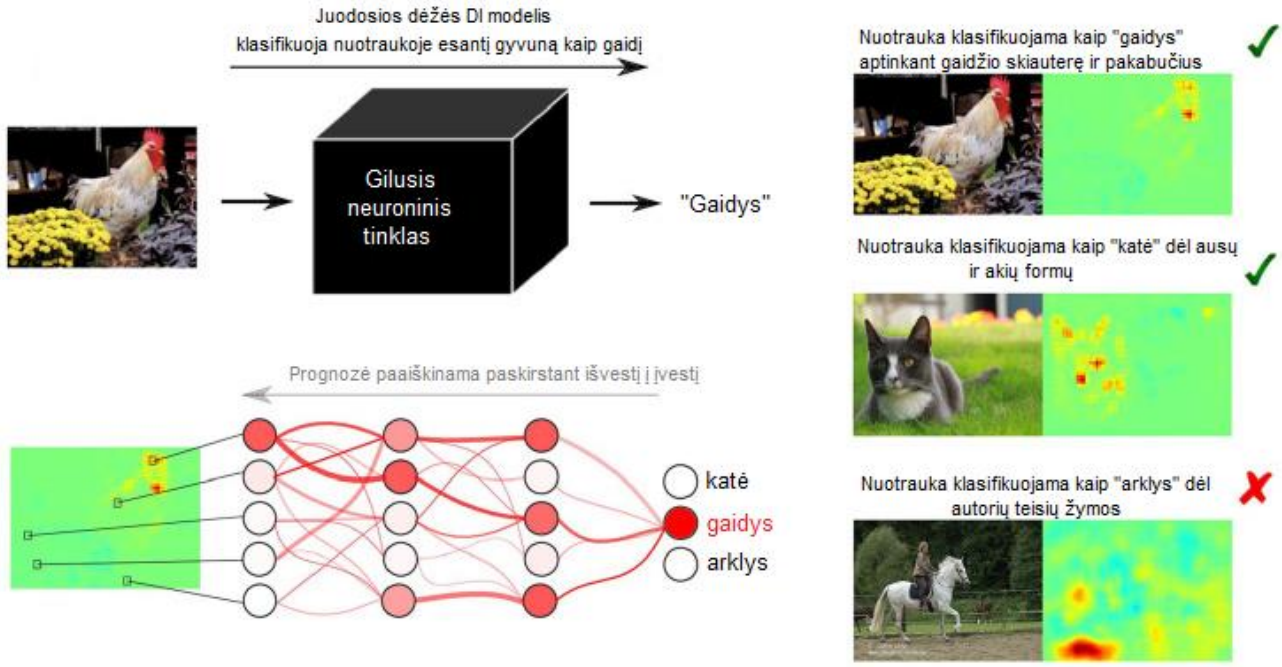
1.6. Paaiškinamas dirbtinis intelektas

Mašininiu ir giliuoju mokymosi pagrįsti dirbtinio intelekto metodai įrodė savo naudingumą pasiekdami aukštus tikslumo rodiklius. Tačiau šių modelių netiesines ir sudėtingas struktūras dažnai sunku interpretuoti. Todėl mokslininkai sukūrė daugybę metodų, kaip paaiškinti jų veikimą bei gaunamų rezultatų logiką. Pastaraisiais metais daugelis mokslininkų skyrė savo pastangas ieškodami naujų metodų, galinčių atskleisti ir paaiškinti logiką, kurią seka duomenimis valdomi (angl. *data – driven*) mašininiai modeliai, taip sukurdami naują dirbtinio intelekto skirtį, žinomą kaip paaiškinamas dirbtinis intelektas (angl. e*xplainable artificial intelligence* – XAI) [38].

Dauguma mašininio ir giliojo mokymosi modelių yra vadinami „juodosios dėžes“ (angl. *black box*) modeliai, kurių veikimas dėl sudėtingų, nelinijinių struktūrų, yra nežinomas ir nesuprantamas. Ši problema yra ypač aktuali dirbtinio intelekto sistemoms naudojamoms realiame gyvenime. Sunku pasitikėti sistemomis, kurių sprendimai negali būti tinkamai interpretuojami, ypač tokiuose sektoriuose kaip sveikatos priežiūra ar savaeigiai automobiliai (angl. *self-driving cars*), kur taip pat reikia atsižvelgi į moralės ir sąžiningumo aspektus [39].

Savaeigės transporto priemonės turi priimti sprendimus kelių milisekundžių tikslumu, remdamosi tuo, kaip jų programinė įranga klasifikuoja regos lauke esančius objektus. Jei toks automobilis priims netinkamą sprendimą dėl neteisingo klasifikavimo kaltės, pasekmės gali būti sukelti pavojų žmonėms [40]. Tai nėra tik teorinė tikimybė, o jau vyksta JAV. 2018 metais dėl ,,Uber“ kompanijos savaeigės transporto priemonės kaltės žuvo moteris Arizonos valstijoje. Ekspertų teigimu automobilio programinė įranga užregistravo objektą priešais transporto priemonę, tačiau klasifikavo jį, kaip plastikinį maišelį ar vėjaritį augalą (angl. *tumbleweed*). Tik paaiškinama dirbtinio intelekto sistema gali išspręsti tokios situacijos aplinkybes ir neleisti incidentui įvykti.

Analizuojant CNN veikimą galima pastebėti, kad pirmojo konvoliucinio neuroninio tinklo sluoksnio svoriai gali būti lengvai suprantami kaip konvoliucijos branduoliai (angl. *kernels*), tačiau kitų sluoksnių svorius darosi sunkiau interpretuoti [41]. Naujausi XAI metodai paverčia modelio rezultatus atgal į įvesties erdvę ir paaiškina prognozę šilumos žemėlapiu (angl. *heatmap*), vizualizuodami, kurie įvesties kintamieji (pikseliai) buvo svarbiausi prognozei. Tai leidžia atskirti prasmingas ir saugias atpažinimo ar klasifikavimo strategijas [42]. Pavyzdžiui, gaidžių atvaizdų klasifikavimas aptinkant gaidžio skiauterę ir pakabučius (angl. wattles) arba kačių atvaizdų klasifikavimas sutelkiant dėmesį į katės ausis ir nosį. Tuo tarpu, arklių atvaizdų klasifikavimas pagal autorių teisių žymą turi būti laikomas kaip neteisingas modelio veikimas, kuris realioje sistemoje neturėtų funkcionuoti.



10 pav. Paaiškinamojo dirbtinio intelekto pavyzdys gyvūnų klasifikavimui [42]

Taigi, kuo geriau dirbtinio intelekto sistemos yra paaiškinamos tuo lengviau yra suvokti, kaip buvo priimti tam tikri sprendimai ar prognozės.

2. Eksperimentuose naudojamas duomenų rinkinys

Norint sukurti aplinkos garsų klasifikavimo sistema ir įvertinti jos tikslumą pirmiausia reikia turėti duomenų rinkinius, kad galima būtų apmokyti neuroninius tinklus. Todėl darbe buvo naudojamas viešai pasiekiamas aplinkos garsų įrašų duomenų rinkinys *UrbanSound8k*.

2.1. *UrbanSound8k* duomenų rinkinys

Aplinkos garsų klasifikavimo sistemos kūrimui darbe naudojamas *UrbanSound8k* duomenų rinkinys [43]. Šiame duomenų rinkinyje yra 8732 pažymėtos miesto aplinkos garsų ištraukos (kurių trukmė neilgesnė nei 4 sekundės) iš 10 kasdiena girdimų garsų tipų:

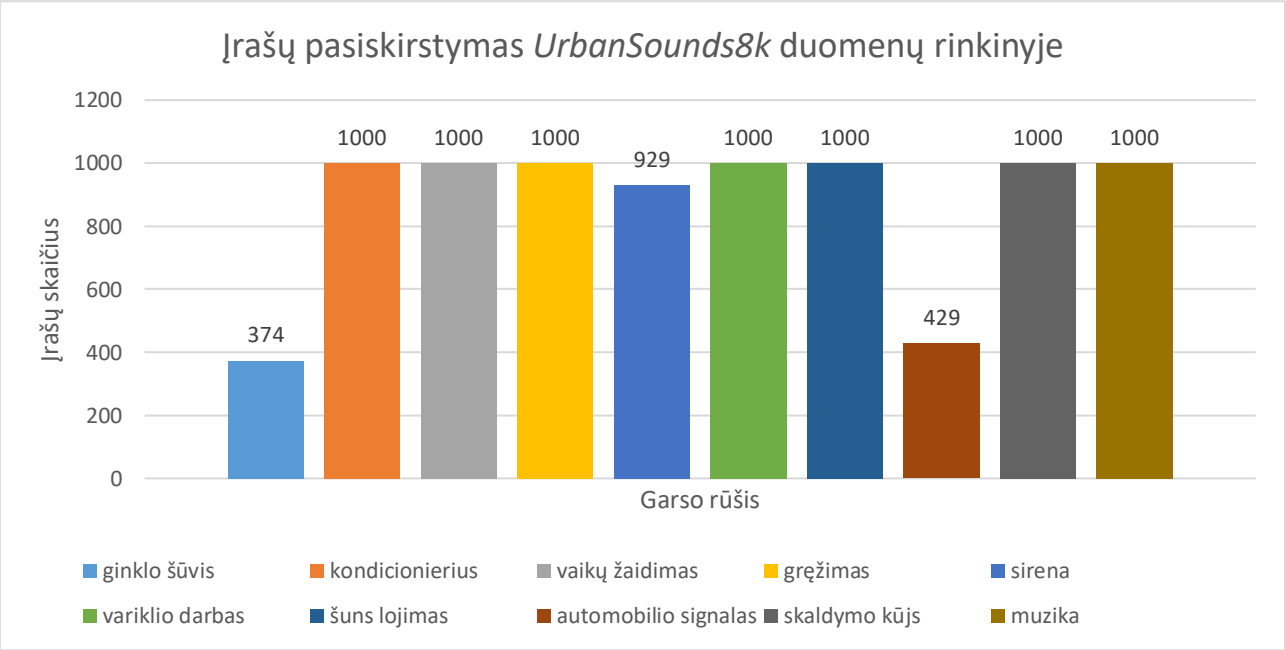
- kondicionieriaus ūžimas
- vaikų žaidimo garsai
- gręžimo garsai
- ginklo šūvis
- sirenos garsai
- transporto priemonių garsinis signalas
- šunų lojimas
- variklio darbas tuščiąja eiga
- skaldymo kūjo (angl. *jackhammer*) darbas
- gatvėje grojanti muzika

Rinkinyje visos ištraukos paimtos iš įrašų, įkeltų į *www.freesound.org* tinklalapį. Failai yra iš anksto surūšiuoti į dešimt aplankalų (aplankai, pavadinti *fold1 – fold10*), kad būtų lengviau atkurti ir palyginti su automatinio klasifikavimo rezultatais, aprašytais moksliniame straipsnyje, analizavusiame šį duomenų rinkinį [43]. Duomenų rinkinyje taip pat pateikiami metaduomenys (**1 lentelė**) kiekvienam garso įrašui, kurių pagalba galima lengviau naudotis duomenų rinkiniu.

1 lentelė. *UrbanSound8k* duomenų rinkinio metaduomenų fragmentas

slice_file_name	fsID	start	end	salience	fold	classID	class
133494-2-0-45.wav	133494	22.5	26.5	1	6	2	children_playing
133797-6-0-0.wav	133797	3.254115	4.66165	1	6	6	gun_shot
133797-6-1-0.wav	133797	6.895626	8.845512	1	6	6	gun_shot
133797-6-2-0.wav	133797	9.116689	10.885791	1	6	6	gun_shot
134717-0-0-0.wav	134717	0	4	1	1	0	air_conditioner

Analizuojant *UrbanSound8k* duomenų rinkinį galima pastebėti klasių disbalanso problemą (**11 pav.**). Duomenų rinkinyje kiekvienos klasės yra po 1000 įrašų išskyrus tris klases: automobilio signalas (429 įrašai), šūvis iš ginklo (374 įrašai) ir sirena (929 įrašai). Siekiant išvengti netikslaus modelio veikimo minimizuojant nuostolių funkciją tolimesniuose modeliavimuose buvo naudoti klasių svoriai gauti naudojanti 1.5.5 skyrelyje aprašytomis formulėmis.



11 pav. UrbanSound8k duomenų rinkinio įrašų kiekio pasiskirstymas tarp klasių

2.2. Modelio testavimas naudojant duomenų rinkinį

Oficialioje *UrbanSound8K* duomenų rinkinio svetainėje[7] autoriai griežtai skelbia, kad naudojant šį duomenų rinkinį reikia atlikti 10-imčių kryžminį patvirtinimą (angl. *10–fold cross validation*), kad rezultatai būtų patikimi, ir galima būtų modelio rezultatus palyginti su kitų autorių darbais bei publikuoti straipsnį naudojant šį duomenų rinkinį. Autoriai teigia, jog pertvarkius duomenis (pavyzdžiui sujungus duomenis iš visų aplankų ir sugeneravus atsitiktinį mokymo/testavimo padalijimą), neteisingai įdėsite susijusius įrašus (pavyzdžiui garso įrašus darytus toje pačioje vietoje) tiek į mokymo, tiek į bandymų rinkinius. Vadinasi modelio rezultatai bus pernelyg dideli – jie neatvaizduos tikrojo jūsų modelio našumo su nematytais duomenimis. Todėl naudosime 1.5.4 skyrelyje aprašyta 10-imčių kryžminį testavimą

[7] Oficiali UrbanSound8k duomenų rinkinio svetainė [žiūrėta 2022-05-13]. Prieiga per: https://urbansounddataset.weebly.com/urbansound8k.html

3. Projektinė dalis

Šiame darbe yra analizuojami skirtingų tipų neuroninius tinklai, technikos naudojamos sumažinti prisitaikymą prie treniravimo duomenų ir skirtingos pačių tinklų architektūras. Kadangi buvo dirbama ir su konvoliuciniais neuroniniais tinklais, iš pradžių apžvelgiami metodai kaip garso signalą galima paversti į paveikslėlį (šiuo atveju į spektrogramą) kurį galimą būtų naudoti kaip dvimačio konvoliucinio sluoksnio įvestį.

3.1. Garso analizės metodai

Išanalizavus kitų mokslininkų naudojamas technikas aplinkos garsų klasifikavimui, šiame darbe dėmesys buvo akcentuojamas į spektrogramas. Analizuojamos skirtingų tipų spektrogramos – standartinės gautos taikant STFT, Melų spektrogramos ir MFCC.

3.1.1. Spektrogramų išgavimas

Norint išgauti 1.1.1 skyriuje aprašytas spektrogramas, šiame darbe yra naudojama *Python* programavimo kalba ir *Librosa* biblioteka garso signalų apdorojimui. Todėl yra sukuriama vieną funkciją išgauti pradines spektrogramas, ir kita, kad galima būtų atvaizduoti amplitudės priklausomybę nuo laiko ir ją palyginti su spektrograma. Šios funkcijos yra pavaizduotos 12 pav. ir 13 pav.

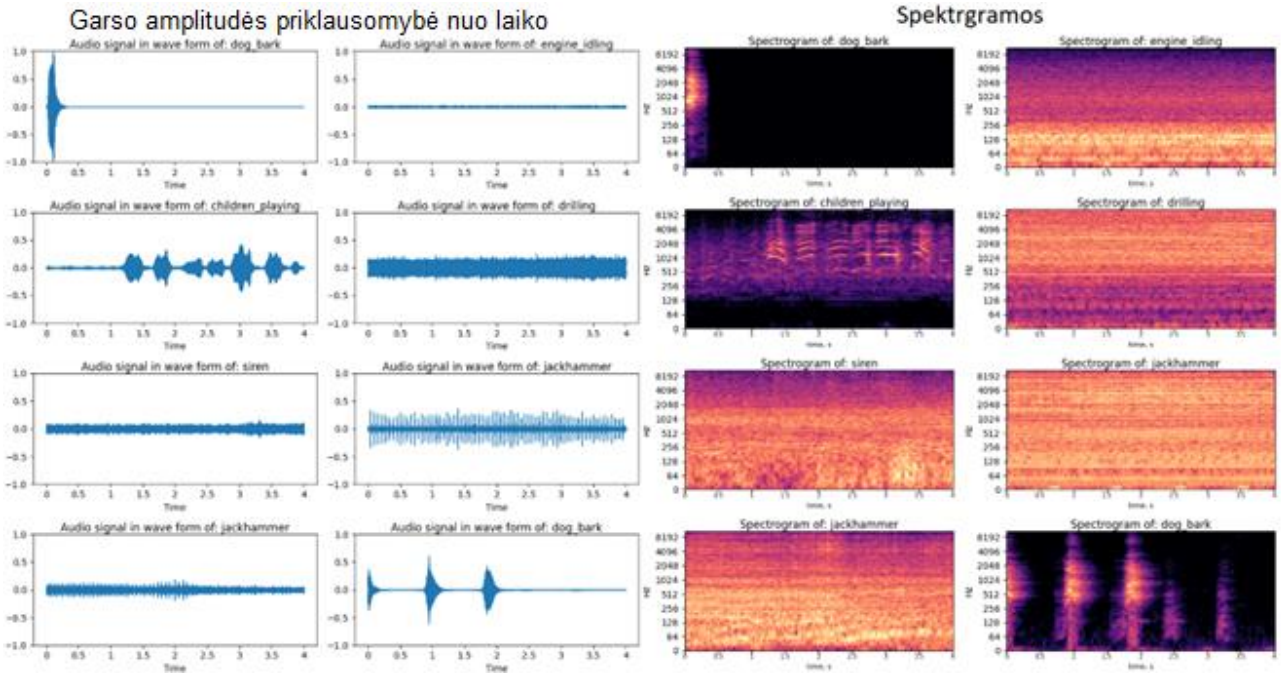
```
def plot_wave_from_audio(self):
    """
    Function plots audio file in wave form
    """
    fig, axs = plt.subplots(4, 2, figsize=(10, 10))
    index = 0
    for col in range(2):
        for row in range(4):
            path = self.urDb.BASE_PATH + "//audio//fold" + str(self.urDb.df["fold"][index]) + '//' + \
                self.urDb.df["slice_file_name"][index]
            data, sr = librosa.load(path)
            data = librosa.util.utils.fix_length(data, 4*sr)

            librosa.display.waveshow(
                data, sr=sr, x_axis='time', ax=axs[row][col], offset=0.0, marker='', where='post')
            axs[row][col].set_ylim(-1, 1)
            axs[row][col].set_title(
                "Audio signal in wave form of: " + str(self.urDb.df["class"][index]), fontsize=12, pad=0)
            index += 60  # get some differerent sounds
    plt.tight_layout()
    plt.show()
```

12 pav. *Python* funkcija vizualizuoti *UrbanSound8k* duomenų rinkinio garsus kaip amplitudę nuo laiko

```python
def plot_basic_spectrograms(self):
    """
    Function calculates STFT for audio file and plots spectrogram
    """
    FRAME_SIZE = 2048
    HOP_SIZE = 512

    fig, axs = plt.subplots(4, 2, figsize=(10, 10))
    index = 0
    for col in range(2):
        for row in range(4):
            file_name = self.urDb.BASE_PATH + "//audio//fold" + str(self.urDb.df["fold"][index]) + '//' + \
                self.urDb.df["slice_file_name"][index]

            audio_file, sample_rate = librosa.load(file_name)
            audio_file = librosa.util.utils.fix_length(
                audio_file, 4*sample_rate)

            stft = librosa.stft(audio_file, n_fft=FRAME_SIZE, hop_length=HOP_SIZE)  # STFT of y
            S_db = librosa.amplitude_to_db(np.abs(stft), ref=np.max)
            librosa.display.specshow(S_db,
                                     sr=sample_rate,
                                     hop_length=HOP_SIZE,
                                     x_axis="time",
                                     y_axis='log',
                                     ax=axs[row][col])
            axs[row][col].tick_params(axis='x', labelsize=7, pad=0)
            axs[row][col].set_xlabel('time, s', fontsize=8)
            axs[row][col].set_title('Spectrogram of: {}'.format(
                self.urDb.df["class"][index]), fontsize=12, pad=0)
            index += 60  # get some differerent sounds
    plt.tight_layout()
    plt.show()
```

13 pav. *Python* funkcija vizualizuoti *UrbanSound8k* duomenų rinkinio garsus spektrogramos formatu

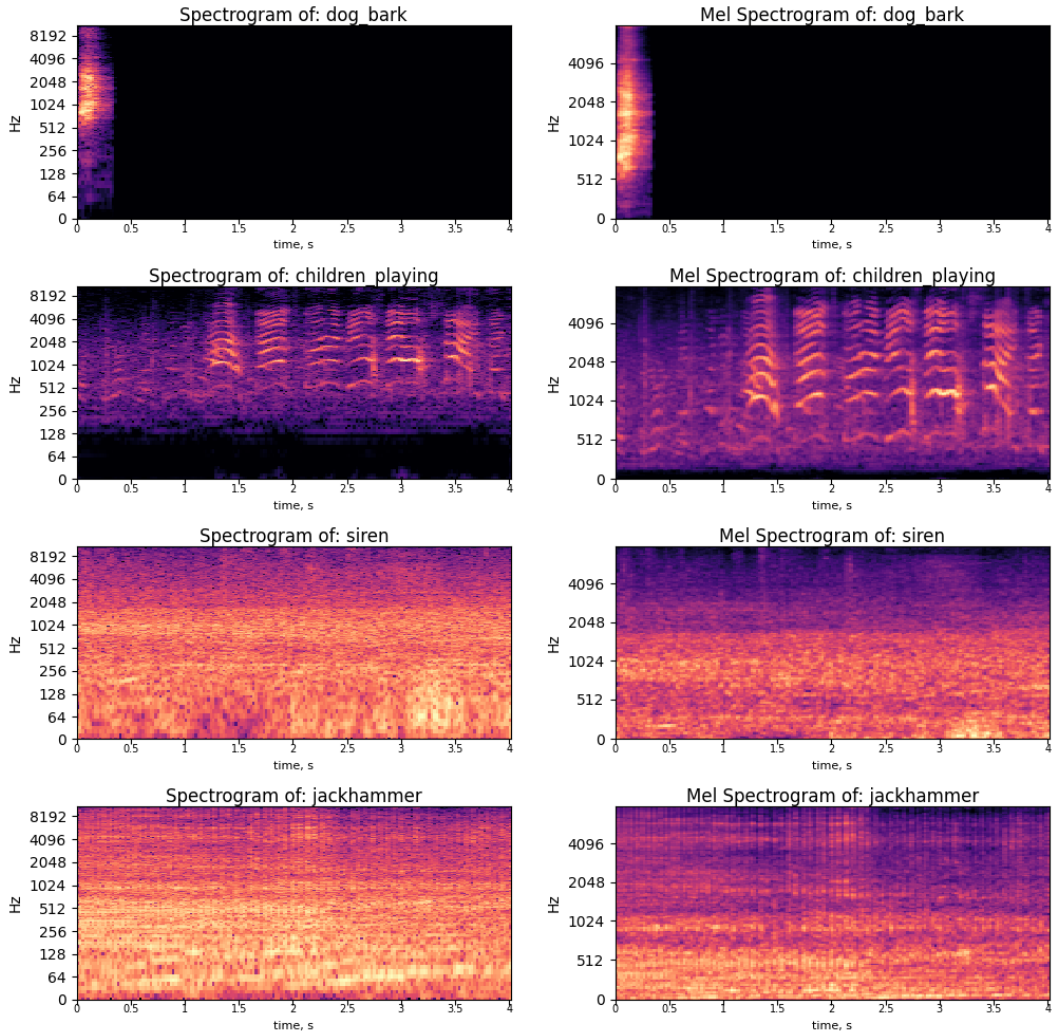
Gauti rezultatai yra pateikiami paveikslėlyje (**14 pav.**), kaip galime matyti spektrogramos išlaiko audio signalų savybes. Pavyzdžiui, pasikartojantis šuns lojimas vaizduojamas kaip periodinės ryškesnės zonos ir tarp jų esanti tuštuma, o tuo tarpu nepertraukiamas gręžimas ar skaldymo kūjo naudojimas (angl. jackhammer) yra atvaizduojamas visoje laiko skalėje bei ties skirtingais dažniais.



14 pav. Gautų spektrogramų ir garso amplitudės priklausomybes nuo laiko grafikų palyginimas

Kaip ir anksčiau aprašytas spektrogramas Melų spektrogramas iš audio signalo galima išgauti pasinaudojus *Librosa* biblioteka, tačiau šį kartą užuot skaičiavus STFT galima pasinaudoti

librosa.feature.melspectrogram() funkcija, kuri šiuos skaičiavimus atlieka funkcijos viduje. Gauti rezultatai pateikiami paveikslėlyje (**15 pav**.).

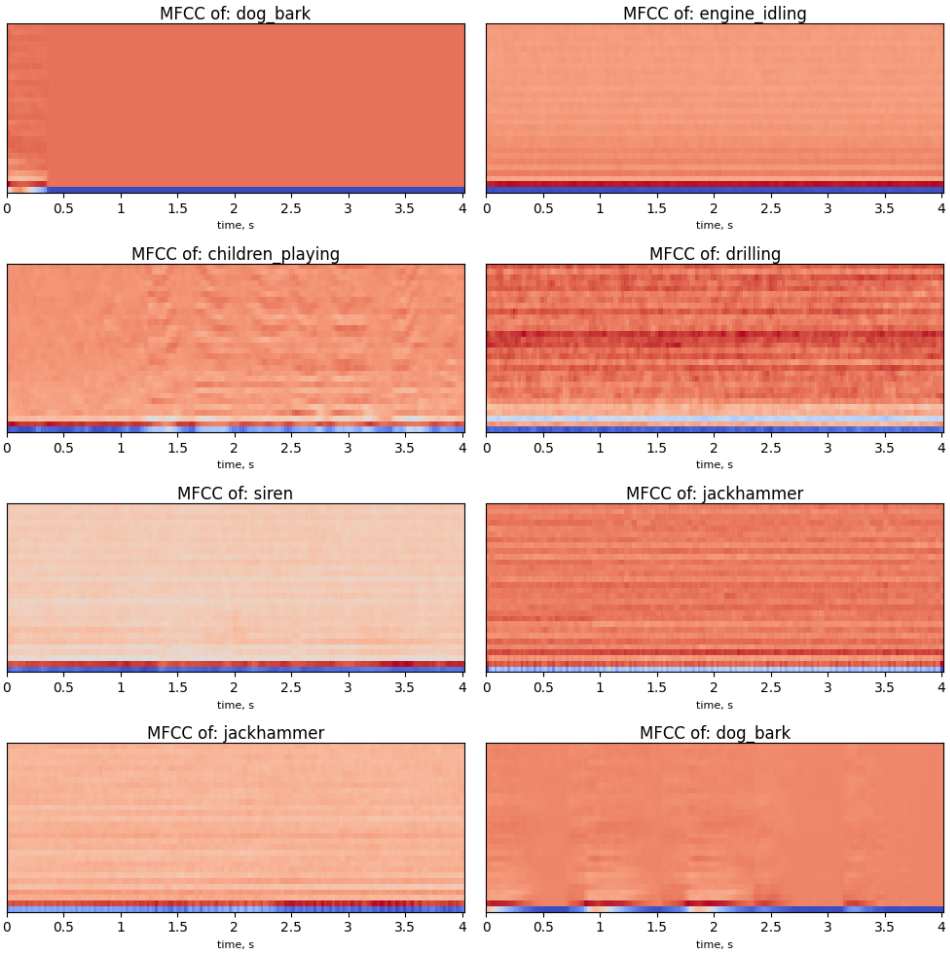


15 pav. Garso signalų (kairėje) ir garso signalų naudojant Melų skalę (dešinėje) spektrogramos

Kaip galima matyti iš paveikslėlyje (**15 pav.)** pateikiamų abiejų tipų spektrogramų, naudojant Melų skalę yra gaunamos spektrogramos su daugiau bruožų esančių žemesnių dažnių zonoje. Palyginus vaikų žaidimų garsus tarp skirtingų spektrogramų galima matyti, jog didesnis paveikslėlio plotas yra užpildomas naudinga informacija. Tai itin svarbu dirbtiniams neuroniniams tinklams, kadangi tuštuma neperteikia naudingų bruožų kuriuos tinklas galėtų išmokti, norint sukurti realiomis sąlygomis veikiančią sistemą.

3.1.2. MFFC spektrogramų gavimas

Norint išgauti MFFC tipo spektrogramas darbe yra naudojama *Librosa* bibliotekos funkcija: *librosa.feature.mfcc()*. Gauti rezultatai pateikiami paveikslėlyje (**16 pav**.) Svarbu paminėti, kad naudojamų koeficientų skaičius priklauso nuo sprendžiamos problemos bei triukšmo lygio garso signale. Paprastai jis būna tarp 20-80, šiame darbe yra naudojama 40 koeficientų kadangi su tokiu spektrogramos dydžiu (spektrogramos aukštis tiesiogiai priklauso nuo koeficientų kiekio) mokymasis vyksta sąlyginai greitai ir atlikus pradinius klasifikavimo eksperimentus šis koeficientų skaičius parodė geresnius rezultatus lyginant su 60 koeficientų spektrogramomis.



16 pav. MFCC tipo spektrogramos, naudojant 40 kepstrinių koeficientų

3.2. Neuroninio tinklo įvestis

Analizuojamų garsų klasifikavimui iš pradžių buvo naudojamas dvimatis konvoliucinis neuroninis tinklas. Kadangi šio tipo tinklo įvestis yra vaizdas (dvimatė matrica) išgaunamos spektrogramos yra konvertuojamos į PNG tipo nuotraukas.

Toliau pateikiamas algoritmas, kaip iš audio signalo yra išgaunamos spektrogramos ir konvertuojamos į PNG tipo nuotraukas:

1) Duomenų rinkinio garso signalai apdorojami ir įkeliami iš kompiuteris atminties į operatyviąją atmintį naudojant *Librosa* biblioteką.

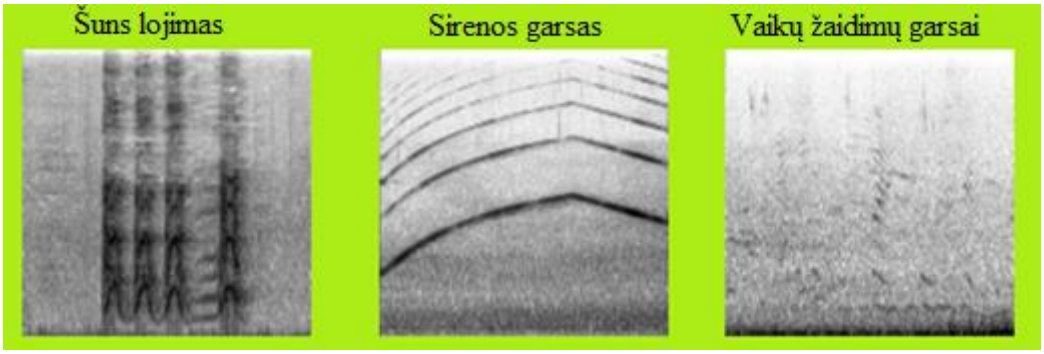
2) Jei garso signalai yra trumpesni nei 4 sekundžių trukmės, jie užpildomi tuštuma, kad spektrogramų duomenys būtų vienodų dimensijų.

3) Išgaunama pasirinkto tipo spektrograma.

4) Gaunamos spektrogramos reikšmes konvertuojamos siekiant, kad jos tilptų į 8 bitų diapazoną (0 – 255).

5) Paveikslėlis yra apverčiamas, kad žemi dažniai atsirastu nuotraukos apačioje.

6) Spalvos yra invertuojamos, kad juodi pikseliai reikštų didesnį energijos kiekį, o tylą – baltas vaizdas.



17 pav. Gaunami PNG paveikslėliai iš Melų spektrogramų

3.3. Duomenų klasifikavimas

Turint modelio įvestis kaip PNG tipo paveikslėlius ir garso signalo klasės numerį iš duomenų rinkinio metaduomenų naudojantis neuroninių tinklų prižiūrimu mokymu galima klasifikuoti šios aplinkos garsus. Kadangi *UrbanSound8k* duomenų rinkinys pasižymi duomenų netolygumu modelio rezultatams įvertinti buvo naudojamas ne tik tikslumo rodiklis bet ir 1.5.6 skyrelyje aptartas F1 rodiklis. Apskaičiuoti šiuos rodiklius buvo naudojama *sklearn*[8] biblioteka. Taip pat pasinaudojant šia biblioteka buvo įvertintas pasvertas F1 rodiklis, kuris atsižvelgia į kiekvienos klasės įrašų skaičių. Šie rodikliai buvo gauti pasinaudojant: *f1_score(y_true, y_pred, average='macro')* ir *f1_score(y_true, y_pred, average='weighted')* funkcijomis.

3.3.1. Konvoliucinio neuroninio tinklo mokymas

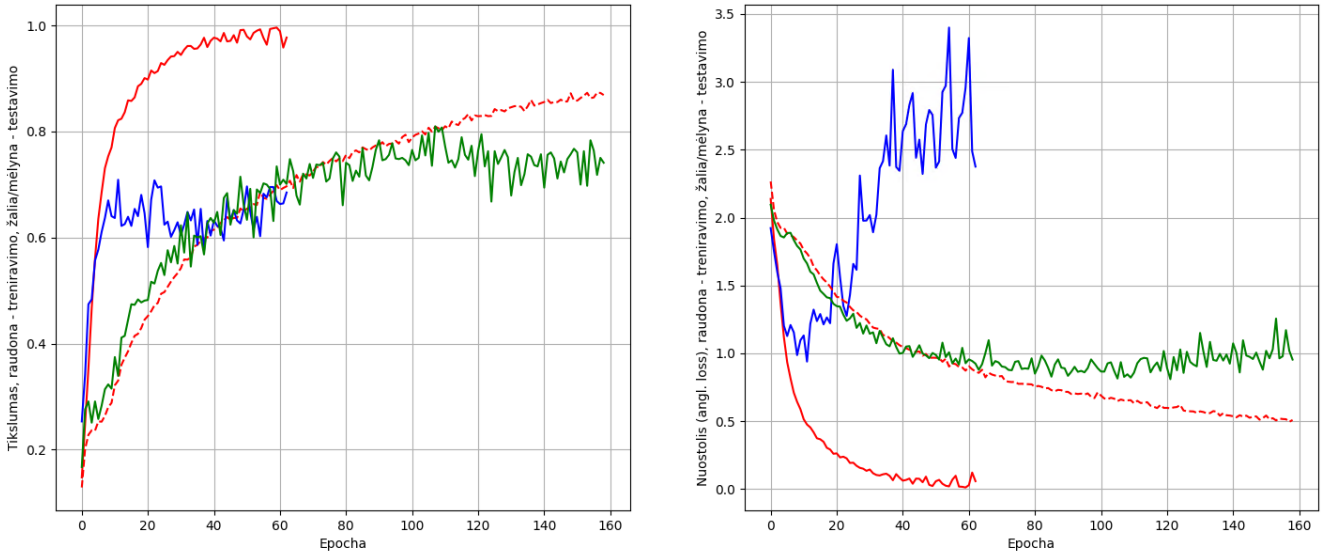
Pradiniam tinklo mokymosi įvertinimui nebuvo naudotas 10-imčių kryžminis patikrinimas, nes šis modelio įvertinimo metodas užtrunka ganėtinai ilgai ir reikalauja didelio kiekio kompiuterinių resursų. Tačiau pradiniam veikimui įvertinti treniravimui buvo naudoti 2-10 rinkinių duomenys, o pirmasis rinkinys paliktas modelio testavimui. Būtent šis rinkinys buvo pasirinktas, nes oficialioje *UrbanSound8k* duomenų rinkinio svetainėje rašoma, kad modeliai dažniausiai pasiekia daug aukštesnį tikslumą, treniruojant su 1-9 rikiniais ir testuojant su 10 rinkiniu, palyginant, kai treniruojama su 2-10 rinkiniais ir testuojant su 1 rinkiniu.

Pats konvoliucinis neuroninis tinklas buvo sukurtas naudojant *Keras* biblioteką iš *Tensorflow*. Kaip modelio įvestis buvo naudota 173 pikselių pločio ir 128 pikselių aukščio Melų spektrogramos. Tinklą sudarė keturi konvoliuciniai sluoksniai (atininkamai su 64, 128, 256 ir 512 filtrais) po kiekvieno iš jų buvo naudojamas telkimo sluoksnis. Po konvoliucinių sluoksnių įvesties dimensijas paversti į vektorių buvo naudojamas *GlobalAveragePooling2D* sluoksnis, kuris palyginus su įprastai naudojamu *Flatten* sluoksniu geba sumažinti parametrų skaičių naudojant telkimo operaciją. Po šio sluoksnio buvo naudota pilnai sujungtas sluoksnis su 128 neuronų. Paskutinis – aktyvavimo sluoksnis buvo sudarytas iš 10 neuronų (klasių skaičiaus kiekio duomenų rinkinyje) ir naudojo *softmax* aktyvavimo funkciją.

Nors sukūrus pradinį tinklą ir atlikus treniravimą buvo pasiekti neblogi rezultatai – 70,9 % testavimo tikslumas ir 70,6% pasverto F1 rodiklio rezultatas. Tačiau buvo pastebėta prisitaikymo prie treniravimo duomenų problema. Todėl modeliui buvo panaikintas ketvirtasis konvoliucinis sluoksnis,

[8] *sklearn* biblioteka [žiūrėta 2022-05-13]. Prieiga per: https://scikit-learn.org/stable/

pritaikytos parametrų išmetimo ir reguliavimo technikos. Tai leido pasiekti tolygesnį mokymosi procesą bei užfiksuoti aukštesnius testavimo tikslumo 80,9 % ir F1 įverčio – 80,7 % rodiklius.



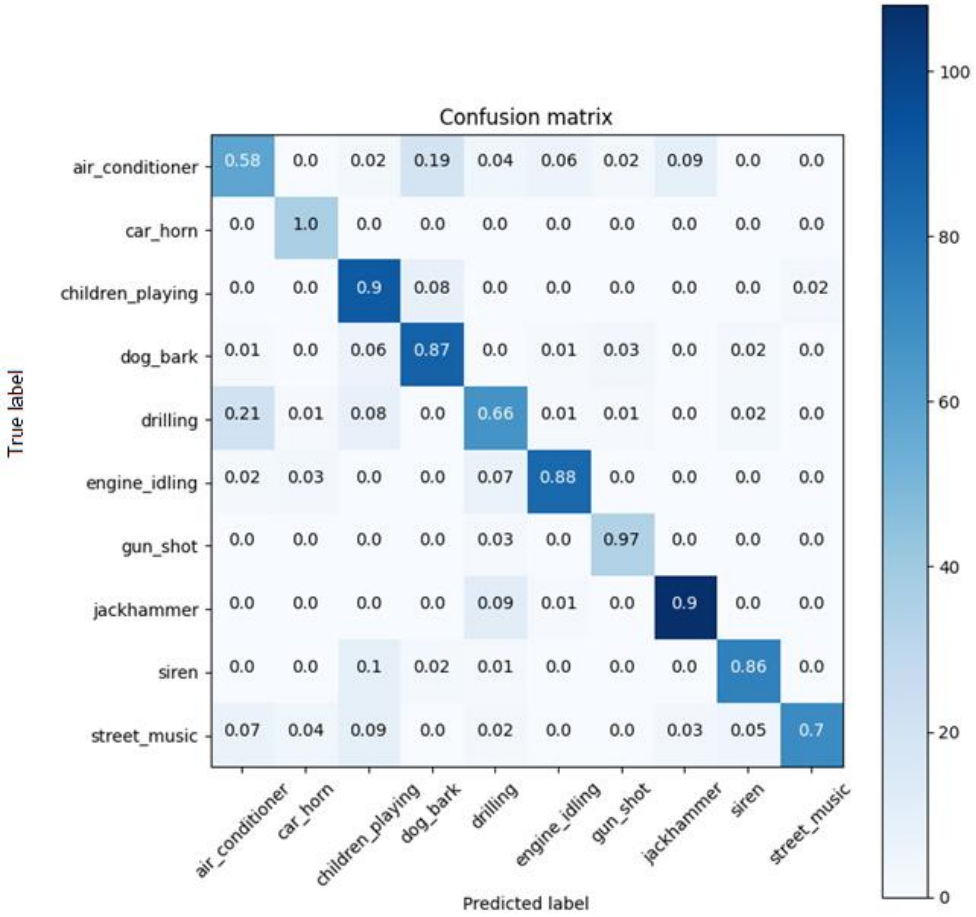
18 pav. Modelių rezultatų palyginimas. Raudona – treniravimo, mėlyna – testavimo kreivės pradiniam modeliui. Punktyrinė raudona – treniravimo, žalia – testavimo kreivės sumažintam modeliui

Iš grafikų (**18 pav.)** galima matyti, kad šie pakeitimai padarė įtaką treniravimo tikslumui, kuris sumažėjo nuo 98 % iki 86 %, bet svarbiausia pagerėja testavimo rodikliai (su nematytais modelio duomenimis treniravimo metu). Verta paminėti, kad su pirmuoju modeliu mokymosi procesas vyko ženkliai trumpiau, nes ankstyvo stabdymo funkcija nutraukė mokymąsi, kadangi nebuvo pasiekti geresni tikslumo rezultatai besitęsiant mokymuisi (ankstyvas stabdymas buvo nustatytas su 30 % mokymosi epochų tolerancija t.y. 170 · 0,3 = 51) .

2 lentelė. Pradinių modelių rezultatų palyginimas

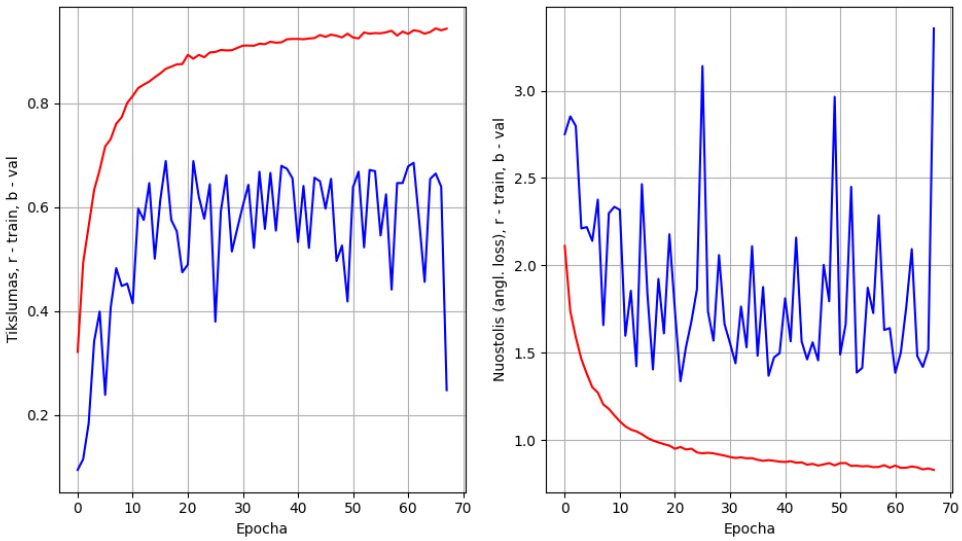
Modelio tipas	**Tikslumas, %**	**F1 rodiklis, %**	**F1 pasvertas rodiklis, %**	**Nuostolių funkcijos vertė**
CNN modelis 4 konvoliucijos sluoksniai	70,9	73,9	70,6	0,98
CNN modelis 3 konvoliucijos sluoksniai + išmetimas + reguliavimas	80,9	82	80,7	0,83

Detalūs modelių rezultatų palyginimai pateikiami lentelėje (**2 lentelė**), o geresniojo modelio gauta sumaišties matricą pateikta **19 pav**. Iš šios sumaišties matricos akivaizdu, kad modelis geba gerai klasifikuoti tokias klases kaip: ginklo šūvis, automobilio signalo garsas, šuns lojimas ar sirenos ūžimas. Tačiau maišo kondicionieriaus veikimo garsą su gręžimo garsais. Tai galima paaiškinti tuo, kad šie signalai dažnu atveju turi pašalinio triukšmo bei neturi aiškių garso bruožų.



19 pav. Sumaišties matrica

Atlikus pradinius testus taip pat buvo pastebėta, kad imties normalizavimo technika išbalansuoja neuroninio tinklo testavimo rinkinio tikslumą (**20 pav**.). Matomai atsiranda pastebimas skirtumas tarp treniravimo ir testavimo kreivių pobūdžio. Tai galėjo atsitikti dėl neteisingo veikimo kartu su *ReLU* aktyvavimo funkcija ar parametrų mažinimo technika. Taip pat įtakos gali turėti specifinis *UrbanSound8k* duomenų rinkinio netolygumas. Kiti mokslininkai analizavę neatitikimą tarp testavimo ir treniravimo tikslumo naudojant imties normalizavimo techniką teigia, kad tai atsitinka treniravimui naudojant mažus imties dydžius (angl. *minibatches*) [44]. Šios problemos išspręsti negalima dėl turimų kompiuterinių resursų, kurie gebėtų apdoroti dideles duomenų imtis, todėl tolimesnėje analizėje imties normalizavimo technikos naudojimo buvo atsisakyta.

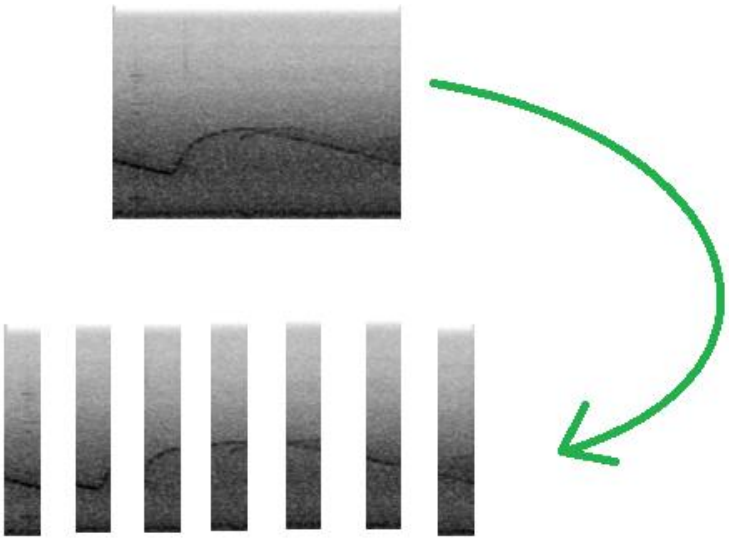


20 pav. Gauti modelio rezultatai naudojant imties normalizavimo techniką

3.3.2. LSTM neuroninio tinklo mokymas

Nors konvoliucinis neuroninis tinklas pasiekė patenkinamus pradinius rezultatus, darbe taip pat analizuojami rekurentiniai – ilgos-trumpos atminties neuroniniai tinklai. Buvo analizuojama dviejų skirtingų tipų LSTM tinklai – dvimatis konvoliucinis LSTM tinklas (LSTM – CNN 2d), kuriame įvestis buvo padalinti spektrogramų paveikslėliai ir vienmatį konvoliucinį LSTM tipo neuroninį tinklą (LSTM – CNN), kuris kaip įvestį geba priimti neapdorotą audio signalą.

Konvoliuciniame LSTM tinklo variante naudojame 16 atminties ląstelių o spektrogramas daliname į aštuonerias dalis pagal horizontaliąją (laiko) ašį. Šiuo veiksmu yra stengiamasi atrasti pasikartančius bruožus analizuojant signalą nuosekliai laike. Šios operacijos vizualizacija pateikiama **21 pav**. padalinus sirenos spektrogramą.



21 pav. Spektrogramos padalinimo vizualizacija

Akivaizdu, kad šio tipo neuroninio tinklo įvestis bus aštuonerios spektrogramų dalys, kurių aukštis nepakinta, tačiau kiekvieno paveikslėlio plotis sumažėja 8 kartus. Taigi modelio įvestis buvo: aštuoni

PNG tipo paveikslėliai kurių kiekvieno plotis – 20 pikselių, o aukštis –128 pikseliai. Tuo tarpu MFCC tipo spektrogramose paveikslėlio aukštis priklausė nuo kepstrų koeficientų skaičiaus, kuris vertė buvo 40, todėl ir paveikslėlio aukštis buvo 40 pikseliai. Svarbu paminėti, kad sumažėjus paveikslėlio dimensijos negalima naudoti tiek pat telkimo sluoksnių ir tokių pačių filtrų dydžių kaip anksčiau analizuotame CNN tipo tinkle. Todėl buvo atlikti pakeitimai, kad telkimo sluoksniai mažintų įvesties aukštį labiau nei plotį. Kaip sukuriamas šis modelis galima matyti iš **22 pav**. pateikiamo programinio kodo.

```
def get_lstm(x_train, class_cnt):
    lstm = Sequential()
    cnn = Sequential()

    # Layer 1
    cnn.add(Conv2D(filters=64, kernel_size=4, kernel_regularizer=regularizers.l2(1e-3), activation='relu', input_shape = (x_train
        .shape[2], x_train.shape[3], x_train.shape[4])))
    cnn.add(MaxPooling2D((3, 2),strides=3 ))
    cnn.add(Dropout(0.25))

    # Layer 2
    cnn.add(ZeroPadding2D((1, 2)))
    cnn.add(Conv2D(filters=88, kernel_size=3, kernel_regularizer=regularizers.l2(1e-3), activation='relu', padding='valid' ))
    cnn.add(MaxPooling2D((2, 1), strides=2 ))
    cnn.add(Dropout(0.25))

    # Layer 3
    cnn.add(Conv2D(filters=128, kernel_size=2, kernel_regularizer=regularizers.l2(1e-3), activation='relu', padding='valid' ))
    cnn.add(GlobalAveragePooling2D())

    lstm.add(TimeDistributed(cnn, input_shape=(x_train.shape[1], x_train.shape[2], x_train.shape[3], x_train.shape[4])))
    lstm.add(LSTM(16, dropout=0.0, recurrent_dropout=0.2))

    lstm.add(Dense(class_cnt, activation='softmax'))
    return cnn, lstm
```

22 pav. LSTM tipo neuroninio tinklo sukūrimas naudojant *Python*

Analogiškai yra sukuriamas vienmatis konvoliucinins neuroninis tinklas, tačiau jame įvestis yra padalintas audio signalas į keturias dalis ir vykdoma vienmatė sąsūkos operacija.

4. Tyrimo rezultatai

Gilaus mokymosi modelių, skirtų aplinkos garsų klasifikacijai, projektavimas ir testavimas buvo atliktas *Python 3.10.4* aplinkoje naudojant mašininio mokymosi biblioteką *Keras*, o ši biblioteka naudojo *Tensorflow 2.8* versiją. Galutiniai eksperimentai buvo atlikti stacionariame kompiuteryje su *Intel Core i5-2400K* 3,10 GHz procesoriumi, 16 GB RAM atmintimi ir 6 GB *NVIDIA GeForce GTX 1060* vaizdo plokšte.

Darbe analizuojami ir palyginami skirtingi spektrogramų tipai, vertinamas dešimties imčių kryžminio testavimo klasifikavimų tikslumų vidurkis, pateikiami tikslumų standartiniai nuokrypiai. Taip pat pateikiami modelio F1 rodiklio įverčiai.

Algoritmų rezultatų palyginimui su kitų autorių darbais buvo pasirinkti keli baziniai modeliai, pasirinkimui prioritetas buvo teikiamas pagal darbo citavimo kiekį ir programinio kodo prieinamumą. Remiantis šiais kriterijais buvo pasirinkti: 2015 metų K. J. Piczak‘o darbas „Aplinkos garsų klasifikacija su konvoliuciniais neuroniniais tinklais“ [45], 2017 metų J. Salamon‘o darbas „Gilieji konvoliuciniai neuroniniai tinklai ir duomenų augmentacija aplinkos garsų klasifikavimui“ [46] ir moderniausias bei vienus iš geriausių rezultatus[9] pasiekęs su *UrbanSound8k* duomenų rinkiniu S. Abdoli‘o ir kitų autorių atvirojo kodo darbas [47].

Verta paminėti, kad LSTM dvimačio konvoliucinio tinklo mokymas užtruko ženkliai ilgiau todėl teko sumažinti treniravimo epochų skaičių apie 10 %. Dvimačio konvoliucinio tinklo vieno imties mokymas truko apie 45 minutes, o LSTM dvimačio bei vienmačio konvoliucinio tinklo apie 60 minučių. Gauti rezultatai pateikiami lentelėje (**3 lentelė**).

3 lentelė. Skirtingų modelių galutinių rezultatų palyginimai naudojant 10 imčių kryžminį patikrinimą

Modelio tipas	**Įvesties tipas**	**Tikslumas, %**	**Standartinis nuokrypis, %**	**F1 rodiklis, %**	**F1 pasvertas rodiklis, %**
CNN (2d)	Spektrograma	69,5	4,5	71,24	69,2
	Melų spektrograma	74,6	4,4	76,2	74,3
	MFCC	63,5	4,3	64,3	62,5
LSTM – CNN (1d)	Audio signalas	60	5,3	60,7	59,3
LSTM – CNN (2d)	Spektrograma	66	5,1	66,7	65,5
	Melų spektrograma	67,9	5,8	68,8	67,2
	MFCC	58,7	3,8	58,8	57,8
CNN (2d) (bazinis modelis), 2015 m. [45]	Melų spektrograma + augmentacija	73,1	~3*	–	–
Gilusis CNN (2d) (bazinis modelis), 2017m. [46]	Melų spektrograma + augmentacija	79	~5*	–	–
CNN (1d) 2022 [47]	Audio signalas + augmentacija	89	~2*	–	–

* Mokslininkai standartinį nuokrypį pateikia grafine forma, o ne skaitine išraiška.

[9] Viešai prieinamo programinio kodo modelių rezultatai [žiūrėta 2022-05-13]. Prieiga per: https://paperswithcode.com/sota/environmental-sound-classification-on

Lyginant konvoliucinio tinklo rezultatus su skirtingo tipo įvestimis akivaizdu, kad naudojant Melų spektrogramas buvo pasiekti geresni nei naudojant paprastas spektrogramas. Tikslumo rodiklis naudojant Melų spektrogramas buvo 5,1 procento aukštesnis, o tiek F1 tiek pasverto F1 gauti rodikliai buvo apie 5 procentais aukštesni. Analizuojant MFCC spektrogramas galima matyti, kad tikslumas (63,5 %) ženkliai mažesnis nei Melų (74,6 %) ar paprastos spektrogramos (69,5 %).

Lyginant LSTM-CNN dvimačio tipo tinklą su CNN galima matyti, kad CNN dvimatis tinklas pasiekė bent keliais procentiniais punktais geresnius rezultatus su kiekvienu įvesties tipu. Mažiausias tikslumo skirtumas matomas tarp paprastos spektrogramos įvesties (3,5 %), todėl galima teigti jog garsams kuriems nėra tikslinga naudoti Melų spektrogramas LSTM tipo tinklas yra vertas išbandyti. Tačiau su Melų spektrogramomis yra matomas 6,7 procento tikslumo sumažėjimas, todėl galima teigti, kad šio tipo tinklas nėra tinkamas analizuoti garsų signalus naudojant Melų spektrogramas.

Analizuojant neuroninį tinklą kurio įvestis buvo ne spektrogramos, neapdorotas audio signalas galime matyti, kad šis tinklas pasiekė tik 60 % tikslumo rodiklį. Tačiau tokio tipo tinklas yra tinkamiausias realizuoti sistemoje, turinčioje ribotą kiekį skaičiavimo resursų, kadangi nereikia atlikti skaičiavimų kurie turi paversti audio signalą į spektrogramą.

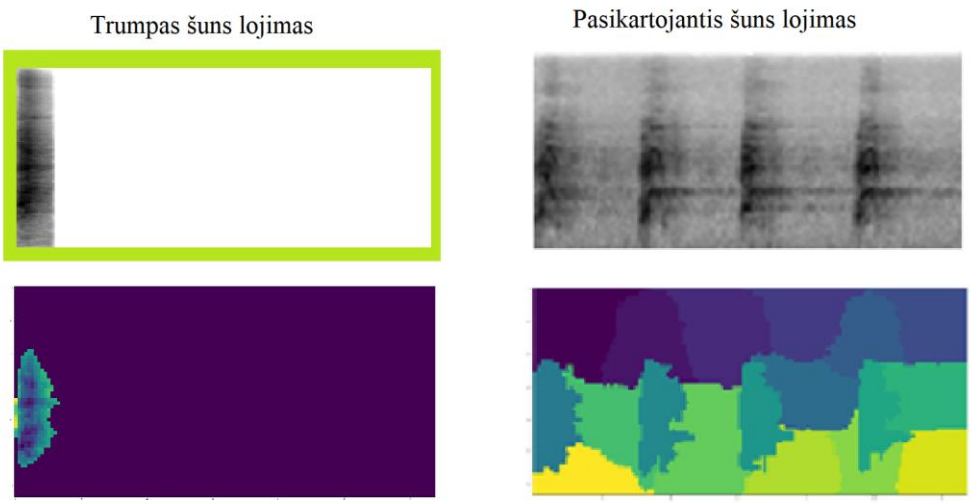
4.1. Paaiškinamojo dirbtinio intelekto rezultatai

Norint pagrįsti šiame darbe sukurtų modelių rezultatų patikimumą buvo panaudotos bibliotekos gebančios paaiškinti neuroninio tinklo sprendimus pagal įvestį. Šiame skyrelyje pateikiami tik konvoliucinio neuroninio tinklo sprendimų paaiškinimai naudojant Melų spektrogramas kaip įvesti, nes šio tipo tinklas pasiekė geriausius rezultatus. Taip pat nuotraukos naudojimas grindžiamas, tuo kad nuotraukoje yra lengviau pažymėti svarbiausius bruožus nei išskiriant esminius skaičius iš įvesties vektoriaus. Rezultatų pateikimas vaizdo formatu yra taip pat naudingas tuo, kad jį geba suprasti žmonės neturintys inžinerinio išsilavinimo, bet turi priimti sprendimus apie modelio pasitikėjimą (pavyzdžiui teisininkai ar galutiniai programos vartotojai).

Paaiškinti modelio rezultatus buvo naudojamos atviro kodo bibliotekas: *Alibi*[10] ir jos inkaro paaiškinimai (angl. *anchor explanations*) metodą ir *Shap*[11] bibliotekos padalijimo paaiškinimo (angl. *partition explainer*) metodą.

Naudojant *Alibi* biblioteką galima išskirti pagrindines nuotraukoje esančias zonas (vadinamus „inkarus“) darančias didžiausią įtaką klasifikatoriaus pasirinkimui. Siekiant išgauti šias zonas biblioteka vaizdą segmentuoja (suskirsto į zonas) pagal neuroninio modelio veikimą. Gauti rezultatai analizuojant šuns lojimo garso signalus pasinaudojus *Alibi* biblioteka pateikiami **23 pav**.

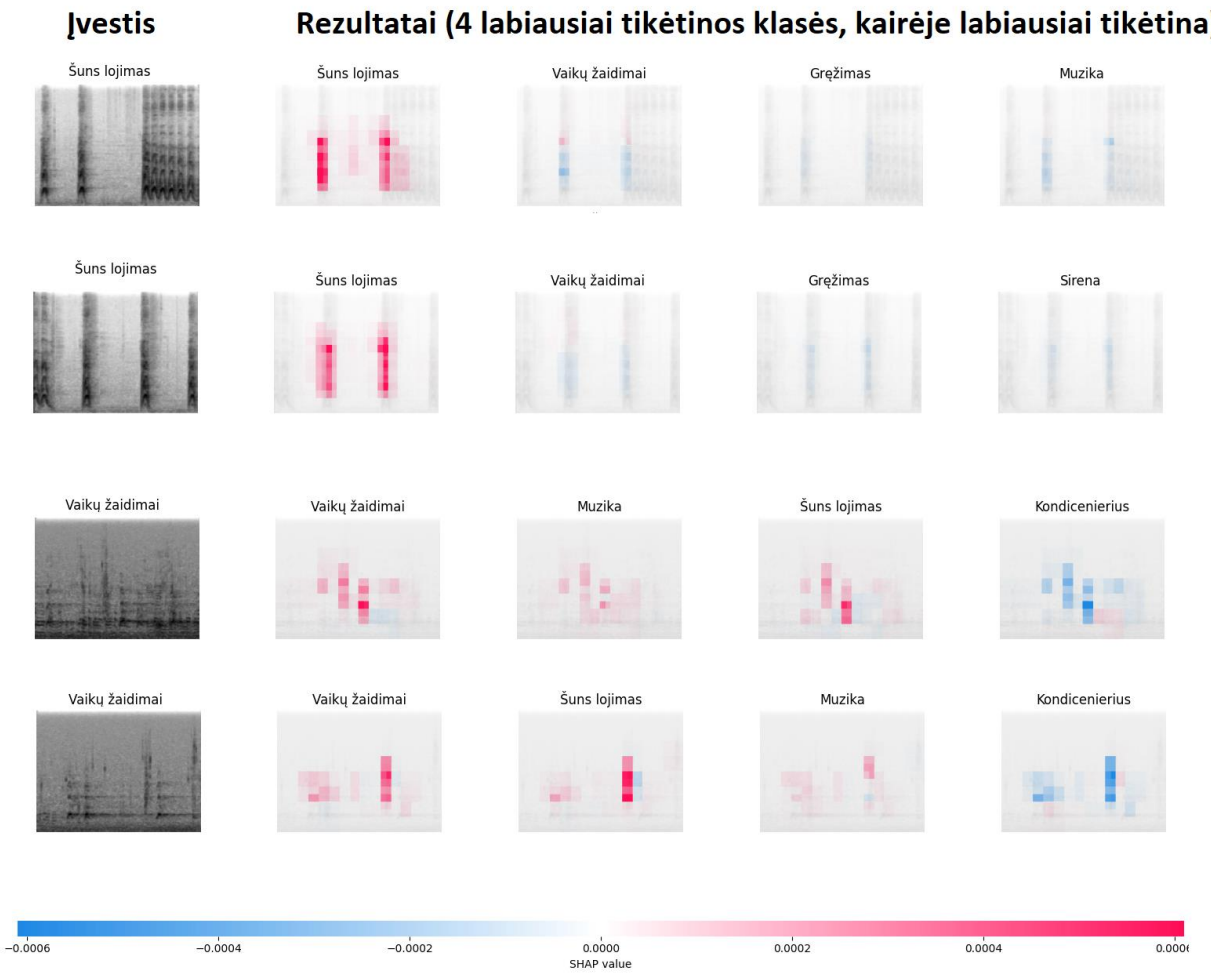
[10] *Albi* biblioteka [žiūrėta 2022-05-13]. Prieiga per: https://github.com/SeldonIO/alibi
[11] *Shap* biblioteka [žiūrėta 2022-05-13]. Prieiga per: https://github.com/slundberg/shap



23 pav. Šuns lojimo garsų analizė naudojant *Alibi* biblioteką

Šiuo atveju (**23 pav**.) analizuojant modelio veikimo paaiškinimą kaip įvestis buvo naudojamos Melų spektrogramos. Trumpo šuns lojimo garso įrašo spektrograma yra pateikiama žaliame fone, kad būtų akivaizdus tylos (balto vaizdo) egzistavimas. Kairėje (**23 pav**.) esančiame apatineme paveikslėlyje yra pavaizduota zona kuri daro didžiausią įtaką modelio priimamam sprendimui. Šiuo atveju galima matyti, kad modeliui garsas yra svarbesnis nei tuštuma. Tuo tarpu iš pasikartojančio šuns lojimo spektrogramos segmentavimo, esančios paveikslėlio (**23 pav**.) dešinėje, galima matyti, kad modelis teisingai geba atskirti garso zonas ir pastebėti keturis atskirus šuns sulojimus. Šiuo atveju yra pateikiama ne viena svarbiausia zona, o kaip paveikslėlis yra suskirstomas yra atskiras zonas naudojantis *Alibi* biblioteką.

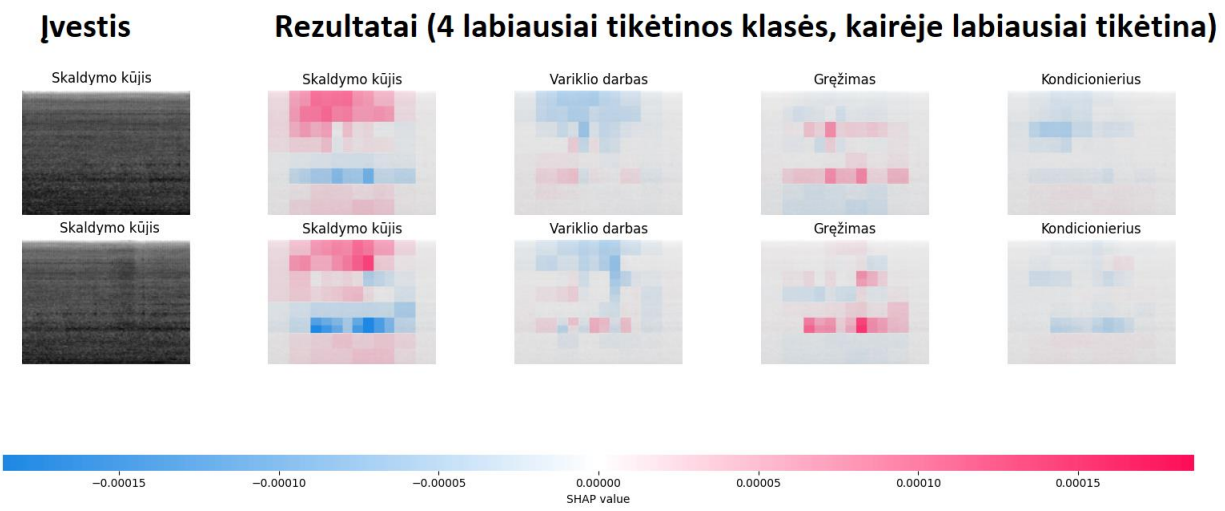
Naudojant *Shap* bibliotekos padalijimo paaiškinimo (angl. *partition explainer*) metodą yra suteikiama galimybė modelio klasifikavimo rezultatus paaiškinti paryškinant zonas kurios turi didžiausią teigiamą ir neigiamą įtaką klasės pasirinkimui.



24 pav. *Shap* bibliotekos paaiškinamojo dirbtinio intelekto rezultatai

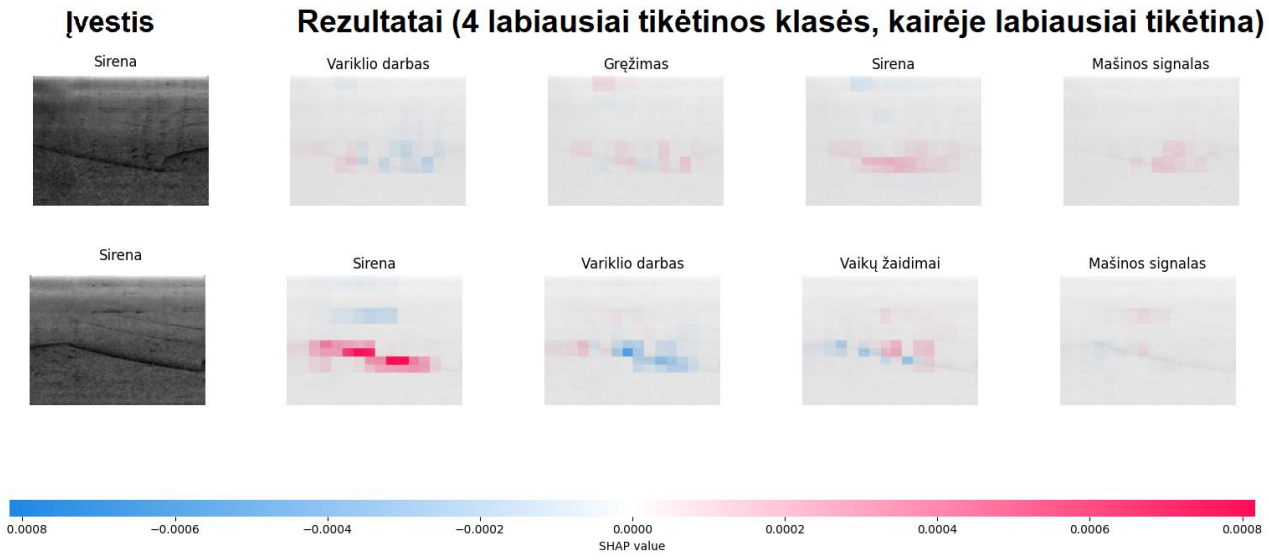
Kaip galima matyti **24 pav**. modelis teisingai klasifikuoja tiek šuns lojimo garsus tiek vaikų žaidimų garsus pagal garso signalo amplitudės aukščiausias vietas visoje dažnio juostoje atsirandančias trumpais laiko intervalais. Verta paminėti, kad modelis šuns lojimą trečiu ir ketvirtu labiausiai tikėtinu klasifikavimu priskiria gręžimo ar sirenos garsams, galima teigti, kad tai atsitinka kadangi šie signalai yra periodinio pobūdžio. Tuo tarpu vaikų žaidimų garsai yra atsitiktinio pobūdžio, todėl modelis gali juos maišyti su gatvės muzikos garsais.

Toliau analizuojant kitas klases (**25 pav**.) galima pastebėti, kad konvoliucinio neuroninio tinklo modeliui skaldymo kūjo klasės pasirinkimui iš spektrogramos įvesties didžiausią įtaką turi aukšto ir žemo dažnio garso dedamosios. Tuo tarpu garso bruožai, esantys ties vertikaliosios ašies (dažnių ašies) viduriu, daro didžiausią teigiamą įtaką gręžimo klasės pasirinkimui.



25 pav. Skaldymo kūjo (angl. *jackhammer*) garsų klasifikavimo analizė

Šiame darbe net ir pasiūlytas geriausio tikslumo modelis nepasiekia 100 % tikslumo bei daro klaidas. Todėl svarbu suprasti, kodėl modelis jas daro. Kaip galima matyti **26 pav**. neuroninio tinklo modelis sirenos signalą vienu iš atveju klasifikuoja neteisingai – kaip variklio darbo garsą. Nors galima matyti, kad ties sirenos klasės bruožų žemėlapiu yra paryškintos vietos darančios teigiamą įtaką klasės pasirinkimui šios informacijos nepakako modeliui priimti teisingą sprendimą.



26 pav. Sirenos garso klasifikavimo analizė

Taigi, kaip įvertinus modelio veikimą su paaiškinamojo intelekto bibliotekomis *Alibi* ir *Shap* galima tvirtai teigti, kad konvoliucinio neuroninio tinklo priimami sprendimai klasifikuojant aplinkos garsus priklauso nuo garso signalo pobūdžio o ne nuo pašalinio triukšmo ar tylos. Todėl šį modelį galima taikyti gyvenime sprendžiant realias problemas.

Išvados

1. Atlikus literatūros analizę ir ištyrus aplinkos garsų klasifikavimo metodus, išsiaiškinta, kad šiems uždaviniams spręsti dažnai naudojami gilieji konvoliuciniai neuroniniai tinklai ir ilgos-trumpos atminties rekurentiniai konvoliuciniai neuroniniai tinklai kaip įvesti naudojantys spektrogramas, arba gilieji konvoliuciniai neuroniniai tinklai kurie kaip įvesti naudoja neapdorotą garso signalą.
2. Užuot pasinaudojus egzistuojančiais jau apmokytais modeliais, pasitelkiant viešą *UrbanSound8k* duomenų rinkinį buvo apmokyti ir sukurti skirtingų struktūrų gilieji neuroniniai tinklai, gebantys klasifikuoti aplinkos garsus.
3. Atlikus pradinius eksperimentus buvo pastebėta, jog konvoliucinis neuroninis tinklas pasiekia geresnius rezultatus (80,9 %) su trimis konvoliucijos sluoksniais ir parametrų šalinimo bei reguliavimo technikomis nei gilesnis konvoliucinis neuroninis su keturiais konvoliucijos sluoksniais (70,9 %), kuris nenaudoja technikų mažinančių tinklo persimokymą.
4. Eksperimentiškai išbandyta imties normalizavimo technika nedavė norimų rezultatų sumažinti prisitaikymą prie treniravimo duomenų, o kaip tik buvo pastebėtas ženklus skirtumas tarp treniravimo ir testavimo tikslumo rodiklių vykstant tinklo apmokymui. Todėl šios technikos naudojimo tolimesniems eksperimentams buvo atsisakyta.
5. Naudojant 10-imčių kryžminį tikrinimą eksperimentiškai ištyrus tinklus buvo nustatyta, kad geriausius testavimo rezultatus pasiekia dvimatis konvoliucinis tinklas, kuris kaip įvesti naudoja Melų spektrogramas: tikslumas – 74,6 procento, F1 rodiklis – 76,2 %. Tuo tarpu šios struktūros tinklas naudojantis spektrogramas ne Melų skalėje pasiekia žemesnį – 69,5 procentų tikslumą, o tinklas naudojantis MFCC tipo spektrogramas tik 63,5 procentų tikslumą.
6. Sukūrus ir eksperimentiškai ištyrus LSTM tipo neuroninius tinklus pastebėta, kad šio tipo tinklai pasiekia prastesnius rezultatus. Taip pat pastebėta, kad klasifikavimo tikslumo skirtumas tarp LSTM tinklo naudojančio spektrogramas ir Melų spektrogramas yra žymiai mažesnis (1,9 %) nei skirtumas tarp CNN tipo tinklo naudojančio spektrogramas ir Melų spektrogramas (4,9 %).
7. Pasinaudojus paaiškinamojo dirbtinio intelekto bibliotekomis buvo įvertintas modelio priimamų sprendimų pagrįstumas. Ši analizė atskleidė, jog modelis aplinkos garsus klasifikuoja pagal garso bruožus esančius spektrogramose, o ne pagal tyla ar kokią pašalinę informaciją.

Literatūros sąrašas

1. CROCCO, M., CRISTANI, M., TRUCCO, A. and MURINO, V. Audio Surveillance: A Systematic Review. *ACM Computing Surveys (CSUR)*, 2016, vol. 48, no. 4. pp. 1-46.

2. ZEINALI, Y. and NIAKI, S.T.A. Heart Sound Classification using Signal Processing and Machine Learning Algorithms. *Machine Learning with Applications*, 2022, vol. 7. pp. 100206.

3. RAHMAN, T., et al. QUCoughScope: An Intelligent Application to Detect COVID-19 Patients using Cough and Breath Sounds. *Diagnostics*, 2022, vol. 12, no. 4. pp. 920.

4. Lewis. *Creation by Refinement: A Creativity Paradigm for Gradient Descent Learning Networks.* , 1988 ISBN NULL-. DOI 10.1109/ICNN.1988.23933.

5. TODD, P. *A Sequential Network Design for Musical Applications.* , 1988.

6. DIEZ GASPON, I., SARATXAGA, I. and LOPEZ DE IPIÑA, K. *Deep Learning for Natural Sound Classification.* Institute of Noise Control Engineering, 2019.

7. ZHANG, Z., XU, S., CAO, S. and ZHANG, S. *Deep Convolutional Neural Network with Mixup for Environmental Sound Classification.* Springer, 2018.

8. JEON, H., JUNG, Y., LEE, S. and JUNG, Y. Area-Efficient Short-Time Fourier Transform Processor for Time–Frequency Analysis of Non-Stationary Signals. *Applied Sciences*, 2020, vol. 10, no. 20. pp. 7208.

9. WINURSITO, A., HIDAYAT, R. and BEJO, A. *Improvement of MFCC Feature Extraction Accuracy using PCA in Indonesian Speech Recognition.* IEEE, 2018.

10. MARUGÁN, A.P., MÁRQUEZ, F.P.G., PEREZ, J.M.P. and RUIZ-HERNÁNDEZ, D. A Survey of Artificial Neural Network in Wind Energy Systems. *Applied Energy*, 2018, vol. 228. pp. 1822-1836. Available from: https://www.sciencedirect.com/science/article/pii/S0306261918311048 ISSN 0306-2619. DOI https://doi.org/10.1016/j.apenergy.2018.07.084.

11. MCCULLOCH, W.S. and PITTS, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. *The Bulletin of Mathematical Biophysics*, 1943, vol. 5, no. 4. pp. 115-133.

12. PAPADOPOULOU, M.P., NIKOLOS, I.K., KARATZAS, G.P. and KYRITSAKAS, G. Artificial Intelligence Methodologies used for the Solutions of Environmental Water Resources Management Problems.

13. MEDVEDEV, V. Tiesioginio Sklidimo Neuroninių Tinklų Taikymo Daugiamačiams Duomenims Vizualizuoti Tyrimai, 2008.

14. YU, W., et al. *Automatic Classification of Leukocytes using Deep Neural Network.* IEEE, 2017.

15. XIE, J., XU, L. and CHEN, E. Image Denoising and Inpainting with Deep Neural Networks. *Advances in Neural Information Processing Systems*, 2012, vol. 25.

16. BAHI, M. and BATOUCHE, M. *Deep Learning for Ligand-Based Virtual Screening in Drug Discovery.* IEEE, 2018.

17. SZE, V., CHEN, Y., YANG, T. and EMER, J.S. Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proceedings of the IEEE*, 2017, vol. 105, no. 12. pp. 2295-2329.

18. GOODFELLOW, I., BENGIO, Y. and COURVILLE, A. *Deep Learning.* MIT press, 2016.

19. YANG, C., LAN, H., GAO, F. and GAO, F. Review of Deep Learning for Photoacoustic Imaging. *Photoacoustics*, 2021, vol. 21. pp. 100215. Available from: https://www.sciencedirect.com/science/article/pii/S2213597920300550 ISSN 2213-5979. DOI https://doi.org/10.1016/j.pacs.2020.100215.

20. KRIZHEVSKY, A., SUTSKEVER, I. and HINTON, G.E. Imagenet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, 2012, vol. 25.

21. GU, J., et al. Recent Advances in Convolutional Neural Networks. *Pattern Recognition*, 2018, vol. 77. pp. 354-377. Available from: https://www.sciencedirect.com/science/article/pii/S0031320317304120 ISSN 0031-3203. DOI https://doi.org/10.1016/j.patcog.2017.10.013.

22. YAMASHITA, R., NISHIO, M., DO, R.K.G. and TOGASHI, K. Convolutional Neural Networks: An Overview and Application in Radiology. *Insights into Imaging*, 2018, vol. 9, no. 4. pp. 611-629.

23. MING, Y., et al. *Understanding Hidden Memories of Recurrent Neural Networks.* IEEE, 2017.

24. NAJMI, A. and MOON, T.K. Advanced Signal Processing: A Concise GuideFirst edition. ed. New York: McGraw-Hill Education, 2020 *Chap. 12, Neural Networks* ISBN 9781260458930.

25. LI, S., et al. *Independently Recurrent Neural Network (Indrnn): Building a Longer and Deeper Rnn.* , 2018.

26. GOPAL, M. Applied Machine Learning1st edition. ed. New York: McGraw-Hill Education, 2019 *Chap. 5.3, NETWORK ARCHITECTURES* ISBN 9781260456844.

27. HOCHREITER, S. and SCHMIDHUBER, J. Long Short-Term Memory. *Neural Computation*, 1997, vol. 9, no. 8. pp. 1735-1780.

28. PETMEZAS, G., et al. Automated Lung Sound Classification using a Hybrid CNN-LSTM Network and Focal Loss Function. *Sensors*, 2022, vol. 22, no. 3. pp. 1232.

29. MUTASA, S., SUN, S. and HA, R. Understanding Artificial Intelligence Based Radiology Studies: What is Overfitting?. *Clinical Imaging*, 2020, vol. 65. pp. 96-99.

30. BALDI, P. and SADOWSKI, P.J. Understanding Dropout. *Advances in Neural Information Processing Systems*, 2013, vol. 26.

31. SRIVASTAVA, N., et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, 2014, vol. 15, no. 1. pp. 1929-1958.

32. MAHSERECI, M., BALLES, L., LASSNER, C. and HENNIG, P. Early Stopping without a Validation Set. *arXiv Preprint arXiv:1703.09580*, 2017.

33. PRECHELT, L. Automatic Early Stopping using Cross Validation: Quantifying the Criteria. *Neural Networks*, 1998, vol. 11, no. 4. pp. 761-767.

34. YAMASHITA, R., NISHIO, M., DO, R.K.G. and TOGASHI, K. Convolutional Neural Networks: An Overview and Application in Radiology. *Insights into Imaging*, 2018, vol. 9, no. 4. pp. 611-629.

35. IOFFE, S. and SZEGEDY, C. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.* PMLR, 2015.

36. BONACCORSO, G.Birmingham: Packt Publishing, Limited, 2018 *Mastering Machine Learning Algorithms : Expert Techniques to Implement Popular Machine Learning Algorithms and Fine-Tune Your Models*, pp. 14-16 ISBN 9781788625906.

37. BUDA, M., MAKI, A. and MAZUROWSKI, M.A. A Systematic Study of the Class Imbalance Problem in Convolutional Neural Networks. *Neural Networks*, 2018, vol. 106. pp. 249-259.

38. VILONE, G. and LONGO, L. Classification of Explainable Artificial Intelligence Methods through their Output Formats. *Machine Learning and Knowledge Extraction*, 2021, vol. 3, no. 3. pp. 615-661.

39. LINARDATOS, P., PAPASTEFANOPOULOS, V. and KOTSIANTIS, S. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy*, 2020, vol. 23, no. 1. pp. 18.

40. A. Adadi and M. Berrada. *Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI).* , 2018 ISBN 2169-3536. DOI 10.1109/ACCESS.2018.2870052.

41. SCHORR, C., GOODARZI, P., CHEN, F. and DAHMEN, T. Neuroscope: An Explainable Ai Toolbox for Semantic Segmentation and Image Classification of Convolutional Neural Nets. *Applied Sciences*, 2021, vol. 11, no. 5. pp. 2199.

42. SCHWENDICKE, F.a., SAMEK, W. and KROIS, J. Artificial Intelligence in Dentistry: Chances and Challenges. *Journal of Dental Research*, 2020, vol. 99, no. 7. pp. 769-774.

43. SALAMON, J., JACOBY, C. and BELLO, J.P. *A Dataset and Taxonomy for Urban Sound Research.* , 2014.

44. SINGH, S. and SHRIVASTAVA, A. *Evalnorm: Estimating Batch Normalization Statistics for Evaluation.* , 2019.
45. PICZAK, K.J. *Environmental Sound Classification with Convolutional Neural Networks.* IEEE, 2015.
46. SALAMON, J. and BELLO, J.P. Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Processing Letters*, 2017, vol. 24, no. 3. pp. 279-283.
47. ABDOLI, S., CARDINAL, P. and KOERICH, A.L. End-to-End Environmental Sound Classification using a 1D Convolutional Neural Network. *Expert Systems with Applications*, 2019, vol. 136. pp. 252-263.