VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Baigiamasis bakalauro darbas

Urbanistinių garsų klasifikavimas naudojant dirbtinius neuroninius tinklus

(Urban sound classification using artificial neural networks)

Atliko:	Tomas Gilys	(parašas)
Darbo vadovas:	Partn. Prof., Dr. Rimantas Kybartas	(parašas)
Recenzentas:	Prof., Dr. Aistis Raudys	(parašas)

Vilnius
2019

Turinys

Įvadas……..……..……..……..……..……..……..……..……..3

1. Dirbtinis intelektas……..……..……..……..……..……..……....3

2. Mašinų mokymasis……..……..……..……..……..……..……... 4
2.1. Prižiūrimas mokymasis……..……..……..……..……..……....4
2.2. Neprižiūrimas mokymasis……..……..……..……..……..……... 5
2.3. Skatinamasis mokymasis……..……..……..……..……..……...5

3. Dirbtiniai neuroniniai tinklai……..……..……..……..……..…… 5
3.1. Dirbtiniai neuronai……..……..……..……..……..……...5
3.2. Aktyvavimo funkcijos……..……..……..……..……..……... 6
3.2.1. ReLU……..……..……..……..……..……... 6
3.2.2. Softmax ……..……..……..……..……..……..7
3.3. Dirbtiniai neuroniniai tinklai……..……..……..……..……..……... 7
3.3.1. Gilusis dirbtinis neuroninis tinklas……..……..……..……... 7
3.3.2. Konvoliucinis dirbtinis neuroninis tinklas……..……..……..……... 8

4. Garso charakteristikos ……..……..……..……..……..…… 8
4.1. Perėjimas per nulį ……..……..……..……..……..……... 9
4.2. MFCC ..……..……..……..……..……..…… 9
4.3. Chromagrama ……..……..……..……..……..……... 10
4.4. Melo spektrograma (angl. „Mel spectrogram”) ……..……..……..……. 11

5. Dirbtinių neuroninių tinklų implementavimas……..……..……..……. 11
5.1. Google Colaboratory ……..……..……..……..……..……11
5.2. Duomenų bazė „UrbanSound8K” ……..……..……..……... 11
5.3. Duomenų paruošimas ……..……..……..……. 15
5.3.1. Charakteristikų gavimas ir jų išsaugojimas ……..……. 15
5.3.2. Žymių paruošimas naudojant „One-hot encoding” ……..……. 15

6. Tyrimas ……..……..……..……..……..……... 16
6.1. Gilieji dirbtiniai neuroniniai tinklai ……..……..……..…… 16
6.1.1. DNN-256-256 ……..……..……..……..……..……...16
6.1.2. DNN-256-256-256 ..……..……..……..……..……... 17
6.1.3. DNN-256-512-256 ..……..……..……..……..……... 18
6.1.4. DNN-512-1024-512 ..……..……..……..……..…… 19
6.1.5. DNN-400-200 ..……..……..……..……..……. 20
6.1.6. DNN-128-256-512-256-128 ..……..……..……..……... 21
6.1.7. Tarpinė DNN išvada ..……..……..……..……..... 22
6.2. Konvoliuciniai dirbtiniai neuroniniai tinklai ..……..……..……. 23
6.2.1. CNN-16-32-64-32 ..……..……..……..……..... 23

6.2.2. CNN-32-32 ...……...……...……...……... 24
6.2.3. CNN-32-64-32 ...……...……...……...……25
6.2.4. CNN-64-128-256 ……...……...……...……... 26
6.2.5. CNN-128-256-512 ……...……...……...……. 27
6.2.6. CNN-128-256-512-1024 ……...……...……...…… 28
6.2.7. Tarpinė CNN išvada ……...……...……...……... 29
6.3. Rezultatai ……...……...……...……... 30

Išvada ……...……...……...……... 32

Summary ……...……...……...……... 33

Literatūros sąrašas ……...……...……...……..... 34

Įvadas

Žmogaus smegenis sudaro šimtai milijardų tarpusavyje sujungtų neuronų, kurie leidžia žmogui nuolat apdoroti gaunamą informaciją. Šiais laikais vis labiau stengiamasi atkartoti mūsų smegenų veiklą naudojant kompiuterį. Įvairių neuroninių tinklų pritaikymo pavyzdžių galime rasti dar ir 1950 metais [Bis15], tačiau tik neseniai buvo pradėta naudoti šiuos tinklus įvairioms klasifikavimo užduotims, įskaitant, bet neapsiribojant kelio ar kelio ženklų, rašytinių simbolių, žmonių veidų atpažinimui. Nors paminėti klasifikavimo uždaviniai yra labiau susiję su vaizdine nagrinėjamo objekto reprezentacija, dirbtiniai neuroniniai tinklai taip pat pradėti naudoti garsų atpažinimui, nors prieš tai šiems uždaviniams buvo naudojami kitokios architektūros negu dirbtiniai neuroniniai tinklai - paslėptieji Markovo modeliai (angl. “Hidden Markov models”) [Bar15] arba dinaminis laiko skalės kraipymas (angl. “Dynamic Time Warping”) balso atpažinimui [MA16]. Tačiau aplinkos garsų klasifikavimui, kur nėra būtinybės atskirti fonemų žodžiuose, dirbtiniai neuroniniai tinklai bei panašios šių tinklų implementacijos (pvz.: rekurentiniai ar konvoliuciniai dirbtiniai neuroniniai tinklai) pradėjo susilaukti daugiau dėmesio, bei panaudos atvejų dar vėliau, dėl vis daugiau viešai prieinamų duomenų (ESC [Pic15], UrbanSounds8k [SJB14]) bei skirtingų garsų duomenų bazių.

Automatinis urbanistinių garsų klasifikavimas turi įvairių pritaikymo galimybių - miesto priežiūros sistemose norint identifikuoti svarbius įvykius [CNK09], galimybė rinkti garso duomenų pavyzdžius tolimesniam miesto vystymui.

Nors didžioji tyrimų dalis susijusi su garso atpažinimu yra dažniausiai siejama su sritimis, kurios yra susijusios su balso, muzikos atpažinimu, urbanistinių garsų analizių yra mažiau. Taip pat svarbu paminėti, kad nors ir yra įvairių tyrimų, susijusių su šio tipo garsų atpažinimu, jie yra labiau skirti aplinkos atpažinimui - ar ta aplinka, kurioje yra įrašytas tam tikras garsas yra parkas, gatvė [CNK09], priešingai negu kad atskirų garso šaltinių identifikavimui (mašinos įspėjamasis signalas, šuns lojimas, oro kondicionierius). Vėlesnė minėta užduotis yra sudėtinga dėl įvairių priežasčių, kaip papildomo triukšmo buvimas, taip pat šie garsai yra dažniausiai girdimi vienu metu dėl skirtingų šių garsų šaltinių. Tačiau teisingai implementavus tokį garso šaltinio atpažinimo algoritmą, būtų įmanoma tai pritaikyti minėtuose kontekstuose, sulaukiant tikslesnių ir patikimesnių rezultatų.

Šio darbo tikslas yra surasti dirbtinių neuroninį tinklą labiausiai tinkantį urbanistinių aplinkos garsų klasifikavime.

Darbo metu bus aptarta teorinė dalis, susijusi su garso apdorojimu, garso charakteristikų išskyrimu bei neuroniniais tinklais. Bus kuriami dviejų tipų dirbtiniai neuroniniai tinklai (gilieji, konvoliuciniai) naudojantis Python programavimo kalba Google Colaboratory platformoje, kurie bus lyginami naudojant tas pačias charakteristikas urbanistinių garsų klasifikavimo uždaviniui. Gauti rezultatai bus lyginami tarpusavyje, bei bus atliekama gautų rezultatų analizė.

1. Dirbtinis intelektas

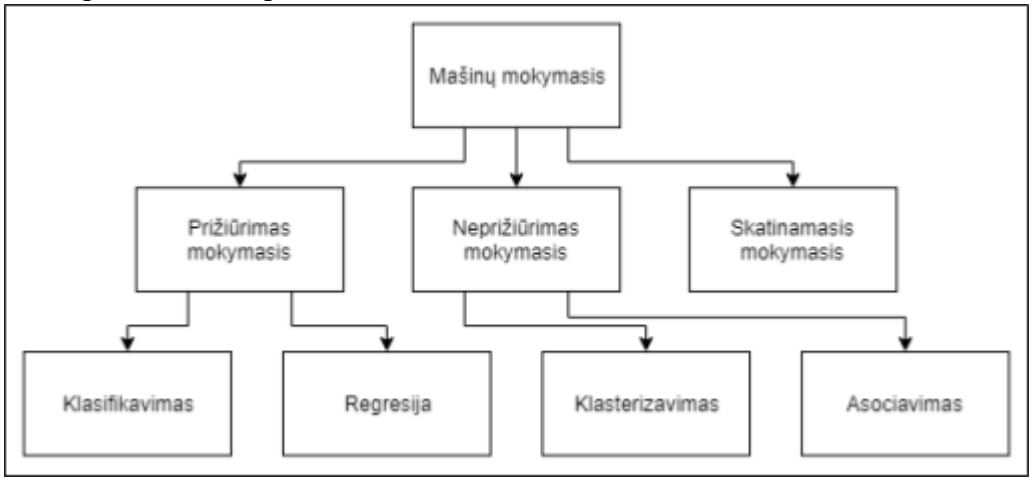
John McCarthy, vienas iš dirbtinio intelekto pradininkų, dar 1956 metais apibrėžė šią discipliną kaip intelektualių mašinų inžineriją bei mokslą apie jas. Intelektuali mašina (kartais vadinama intelektualiu agentu) yra bet koks prietaisas, kuris suvokia aplinką ir bando maksimizuoti galimybę pasiekti jam iškeltus tikslus. Dirbtinis intelektas, šiuo atveju, yra mašinų bandymas atkartoti mūsų kognityvines funkcijas kaip mokymasis bei problemų sprendimas. Dirbtinio intelekto tikslas yra priimti optimalų sprendimą.

Tradiciškai dirbtinis intelektas yra siejamas su Alan Turing dar 20 amžiaus viduryje iškeltu klausimu - “ar mašina gali mąstyti?”, bei pasiūlytu testu norint įvertinti dirbtinio intelekto

galimybes. Jis suformulavo testą, kuris vėliau ir buvo pavadintas „Turingo testu”. Testas buvo atliekamas taip: žmogus A sėdi atskirame kambaryje ir per terminalą bei klaviatūrą bendrauja su dviem agentais B ir C iš kurių vienas yra žmogus, kitas - mašina. Žmogus A nežino, kuris yra kuris. Žmogaus A tikslas yra nustatyti, kuris agentas yra žmogus ir kuris - mašina. Testas yra atliekamas daug kartų, ir jeigu žmogui A nepavyksta teisingai įvardinti B ir C agentų mažiau nei pusę kartų, tai mašina sėkmingai “praeina” Turingo testą ir yra laikoma, kad mašina gali mąstyti [Cyr08].

2. Mašinų mokymasis

Mašinų mokymasis yra dirbtinio intelekto pritaikymas bei kompiuterių mokslo atšaka, kurioje yra kuriamos kompiuterinės programos, kurios bando automatiškai mokytis bei tobulėti nereikalaudami perprogramavimo - suteikiant programai įvesties duomenis, ji bando atrasti įvairių pasikartojimų duomenyse ir to pasekoje priimti sprendimą. Mašino mokymosi algoritmų pritaikymų galime rasti nuo elektroninių laiškų filtravimų iki audio klasifikavimo - vietose, kur neįmanoma sukurti specifinių instrukcijų, pagal kurią turėtų korektiškai veikti programa. Mašinų mokymąsi galime pavaizduoti 1 paveikslėliu.



1 pav. Mašinų mokymosi tipų medis.

Matome, kad mašinų mokymuisi galime išskirti tris pagrindinius tipus, kuriuos dabar aptarsime.

2.1. Prižiūrimas mokymasis

Prižiūrimas mokymasis (angl. „Supervised learning”). Šį mokymosi metodą galime suprasti kaip įvesties duomenų x ir išeities duomenų y padavimą mašinai, o mokymosi metu yra bandoma atrasti algoritmą (ar funkciją) f, kuris sugebėtų atvaizduoti x į y taip, kad:

$$y = f(x) \quad (1)$$

Šio mokymosi tikslas yra bandyti surasti tokį f, kuris leistų nuspėti naujiems įvesties duomenims (x) teisingą išvestį (y).

Šį tipą vadina prižiūrimu mokymu dėl to, kad mes iš pradžių suteikiame mašinai įvesties ir išeities duomenis - mes suprantame, kad jie teisingi, o algoritmas iteratyviai stengiasi daryti spėjimus atsižvelgiant į tai, kokie yra teisingi atsakymai.

Prižiūrimo mokymosi uždaviniai toliau gali būti suskirstytas į šias grupes:

- Klasifikavimas - mašinai yra iškeltas tikslas duomenis priskirti žymę ar klasę (pvz.: ar duotoji nuotrauka yra šuo ar katė, ar duomenys apie pacientą leidžia suprasti, kad jis serga ar neserga).

- Regresija - mašinai yra iškeltas tikslas grąžinti kokį nors realų skaičių kaip spėjimą, atsižvelgiant į pateiktus duomenis (pvz.: bandymas nuspėti kitos dienos rinkos kainas, orų prognozės).

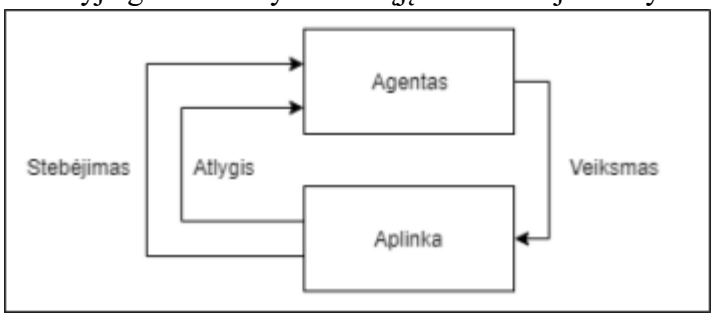
2.2. Neprižiūrimas mokymasis

Neprižiūrimas mokymasis (angl. „Unsupervised learning"). Skirtingai negu kad prižiūrimas mokymasis, šiuo mokymosi metu įvesčiai nėra suteikiami „teisingi" atsakymai - tikslas yra atrasti struktūras bei įvairius pasikartojimus pateiktuose duomenyse be jokių išorinių nurodymų. Šio mokymosi uždavinius galime toliau klasifikuoti į:

- Klasterizavimą - norint atrasti paslėptus pasikartojimus ar šablonus duotuose duomenyse (pvz.: vartotojų grupavimas atsižvelgiant į jų pirkimų istoriją).
- Asociacijų ieškojimą - norint atrasti specifines taisykles kurios sugeba paaiškinti santykį (asociacijas) tarp duomenų (pvz.: žmonės, kurie perka *X* taip pat yra linkę pirkti *Y*).

2.3. Skatinamasis mokymasis

Skatinamasis mokymasis (angl. „Reinforcement learning"). Šio mokymosi būdu yra stengiamasi suteikti algoritmui atgalinį ryšį (skatinimą), kuris leistų algoritmui tobulėti. Skatinimą galime suprasti kaip funkciją, kuri grąžina kokią nors reikšmę po įvykio su aplinka, pvz. yra sukuriamas robotas, kuris stengiasi kuo ilgiau nesusidurti su kitais robotais - jo skatinimo funkciją būtų galima apibrėžti kaip bandymą išbūti kuo ilgiau (didinti funkcijos grąžinamą laiko kintamąjį) neprilietus kitų aplinkos objektų. Šio mokymo tikslas – atrasti tokią strategiją, jog vėliau ANN renkantis veiksmus (mūsų pvz.: atpažinti žodžius) jų kaina būtų mažiausia. Antrame paveikslėlyje galime matyti bendrąjį skatinamojo mokymosi atvejį:



2 pav. Skatinamojo mokymosi pavyzdys

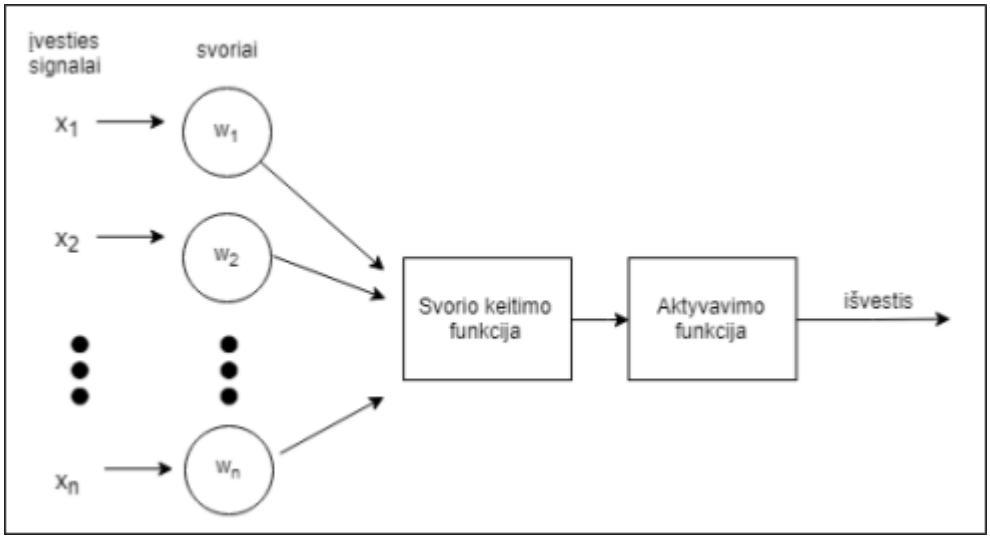
3. Dirbtiniai neuroniniai tinklai

3.1. Dirbtiniai neuronai

Dirbtiniai neuroniniai tinklai (toliau - DNT) yra bandymas kompiuteriu atkartoti gyvūnų smegenų veiklą. Norint suprasti, kas yra DNT, visų pirma reikėtų apibrėžti, kas yra dirbtinis neuronas. Dirbtinis neuronas yra pagrindinis elementas norint sukurti neuroninį tinklą. Kaip ir gyvūnų smegenyse esantys neuronai, dirbtiniai neuronai buvo konstruojami norint replikuoti pagrindines biologinių neuronų savybes. To pasekoje dirbtiniai neuronai atlieka šias pagrindines funkcijas:

1. Įvesties - neuronui yra svarbu sugebėti priimti duomenis iš kitų neuronų.
2. Svorio keitimo - atsižvelgiant į gautus duomenis yra keičiamas stiprumas tarp skirtingų neuronų.

3. Aktyvavimo - nusako kokia bus išvestis, atsižvelgiant į tai, kokia aktyvavimo funkcija yra parinkta.
4. Išvesties - perduodamas rezultatas tolimesniems neuronams.



3 pav. Dirbtinis neuronas

Dirbtinio neurono pavyzdį galime matyti pavaizduotą 3-iame paveikslėlyje.

3.2. Aktyvavimo funkcijos

Aktyvavimo funkcija yra būtina, jei norima, kad neuroninis tinklas gebėtų išmokti netiesines funkcijas. Aktyvavimo funkcijos gali būti skirtingos - darbe mes naudosime „ReLU" bei „softmax".

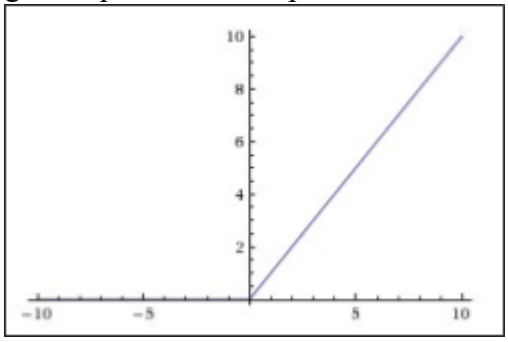
3.2.1. ReLU

ReLU (angl. „Rectified Linear Unit") yra dažniausiai naudojama aktyvavimo funkcija giliojo mokymosi modelių konstravimo uždaviniuose. Bendruoju atveju ReLU funkcija grąžina 0, jei gauna bet kokį neigiamą įvestį, o teigiamai įvesčiai *x* - ją ir grąžina. ReLU galime pateikti lygybe:

$$f(x) = max(0, x), \quad (2)$$

kur *x* yra gaunama įvesties vertė (pvz. - iš kito neurono).

Grafiškai ReLU veikimą galime pavaizduoti taip:



4 pav. Aktyvavimo funkcijos „ReLU" pavaizdavimas

ReLU funkciją naudosime kaip aktyvavimo funkciją visuose sluoksniuose išskyrus išvesties.

3.2.2. Softmax

Angl. „Softmax” - minkštojo maksimumo funkcija yra atsakinga už visų išeities komponenčių pavertimą į tikimybes, sunormuojant išėjimo vektorių į 1 (bendra visų reikšmių suma turi būti lygi 1). Softmax funkciją galime išreikšti lygybe:

$$\varphi(x)_i = \frac{e^{x_i}}{\sum_n e^{x_n}} \tag{3}$$

Ši funkcija yra naudinga klasifikuojant - kiekviena išėjimo vektoriaus komponentė atitinka gali atitikti tikrą klasę. Kiekvienai gautai komponentei $\varphi(x)_i$ mes priskiriame klasės tikimybę, arba kiek „įsitikinęs” yra neuroninis tinklas, jog nagrinėjami duomenys priklauso vienai ar kitai klasei. Galime įsivaizduoti, jog nagrinėjama užduotis yra priskirti gautam garsui garso šaltinio klasę - gatvė, mašina, šurmulys. Atitinkamai pagal gautą įvestį, pavyzdinė išvestis pagal softmax funkciją galėtų būti [0.3, 0.55, 0.15] - pagal tai galime suprasti, kad mūsų neuroninis tinklas spėja, jog garso šaltinis yra mašina.

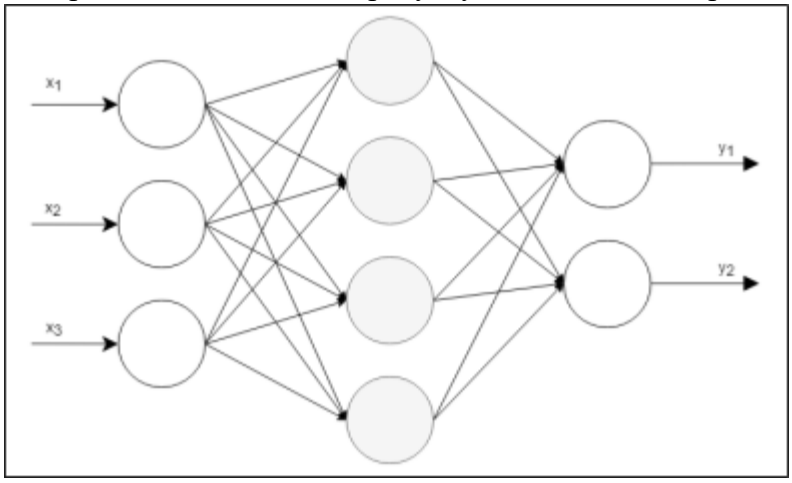
Softmax funkciją naudosime dirbtiniuose neuroniniuose tinkluose paskutiniame lygmenyje, norint klasifikuoti prieš tai buvusių sluoksnių neuronų jungtis į išeities klases.

3.3. Dirbtiniai neuroniniai tinklai

DNT yra sudaryti iš tarpusavyje sujungtų dirbtinių neuronų. Egzistuoja skirtingos DNT rūšys, tačiau visi DNT turi įeities bei išvesties sluoksnius, kurie yra tarpusavyje sujungti su tarp jų dažniausiai esančiu paslėptuoju sluoksniu. Paslėptieji sluoksniai skiriasi priklausomai nuo pritaikomos DNT architektūros.

3.3.1. Gilusis dirbtinis neuroninis tinklas

Gilusis dirbtinis neuroninis tinklas (angl. „Deep Neural Network”, toliau - DNN). Šis tinklas susidaro iš įvesties sluoksnio, išvesties sluoksnio, o tarp jų - vieno ar daugiau paslėptųjų sluoksnių, kurie yra skirti tik apskaičiavimams. DNN pavyzdys matomas 5-ame paveikslėlyje:



5 pav. Gilus dirbtinis neuroninis tinklas

Iš šio 5-o paveikslėlio matome, jog gilusis dirbtinis neuroninis tinklas turi 3 sluoksnius:

- Įeities (kairiausią), į kurį patenka apdoroti duomenys.
- Paslėptą (vidurinis, pažymėtą pilka spalva) kuris atlieka skaičiavimus ir keičia svorius tarp vidurinio ir kitų sluoksnių jungčių.

- Išeities (dešiniausią), kuris atsakingas už tinklo rezultato išvedimą.

Taip pat galime matyti sujungimus tarp šių skirtingų sluoksnių. Šiuo atveju, visi vieno sluoksnio neuronai yra sujungti su visais kito sluoksnio neuronais.

3.3.2. Konvoliucinis dirbtinis neuroninis tinklas

Konvoliucinis dirbtinis neuroninis tinklas (angl. „Convolutional Neural Network", toliau - CNN) yra giliojo mokymosi algoritmas, kuris įvesčiai gauna paveikslėlį ir priskiria svorius (išmokstamus parametrus bei nuokrypius) įvairiems įvesties objektams, kad vėliau galėtų sugebėti tarp jų diferencijuoti. CNN modeliai yra artimiausi lyginant su žmogaus smegenimis, tiksliau - regimosios žievės. Individualūs neuronai atsako į stimulus tik atsižvelgiant tik į aktualius požymius, taip sugeba prafiltruoti ne tiek svarbius požymius. CNN modelių tikslas yra atrasti paslėptąjį šabloną ir konvoliucijų metodais „supaprastinti" gautą vaizdą tik iki tam tikros formos, kurią sudaro pagrindiniai požymiai. Supaprastinus tokias įvestis mes galime paspartinti atpažinimo procesus.

Konvoliucinių dirbtinių neuroninių tinklų modelius sudaro įvairūs lygmenys, kurie yra atsakingi už atskirus charakteristikų atpažinimus.

Nors mūsų darbe mūsų įvestis ir pagrindinė užduotis yra atskirti skirtingus garsinius pavyzdžius ir juos klasifikuoti, o CNN modeliai yra dažniausiai naudojami vaizdų atpažinimui, mes savo CNN pateiksime garsinio pavyzdžio Melo dažnių kepstrinius koeficientus kaip įvestį.

CNN modeliuose pirmas sluoksnis visada būna konvoliucinis sluoksnis (angl. - „Convolutional layer"). Pats pirmas lygmuo yra atsakingas už pagrindinius (ryškiausius) įvesties charakteristikų išskyrimus. Konvoliucijos sluoksnio parametrai yra sudaryti iš mokomųjų filtrų rinkinio. Filtro pozicijos keitimo metu yra sukuriamas dviejų dimensijų dydžio aktyvacijos žemėlapis. Šis sluoksnis grąžina požymių žemėlapių rinkinį, kurio dėka neuroninis tinklas yra mokamas atpažinti tam tikrus dominuojančias savybes, kaip briaunas, spalvą. Šalia konvoliucinių sluoksnių taip pat yra naudojama telkimo funkcija. Funkcijos dažniausiai būna dviejų tipų:

- Maksimalaus telkimo
- Vidutinio telkimo

Telkimo operacijos metu yra mažinama mūsų gauta įvesties reprezentacija, išlaikant pagrindinius reprezentacijos požymius.

Po konvoliucijos sluoksnio bei telkimo operacijos seka pilnai sujungtas sluoksnis - neuronai šiame sluoksnyje yra visi tarpusavyje lokaliai sujungti.

4. Garso charakteristikos

Pirmas žingsnis, prieš leidžiant garsui būti naudojamam atpažinimo uždaviniuose, jį reikia atitinkamai paruošti – reikia suteikti šias garso bangas kompiuteriui suprantamu formatu. Priešingai negu kad paveikslėliai, garsas yra perduodamas bangomis, tad reikia šiuos duomenis paversti į skaičius, kuriuos suprastų kompiuteris. Garso bangos yra vienos dimensijos, ir bet kuriuo laiko momentu turi kažkokią reikšmę atitinkamai nuo taško aukščio toje bangoje. Paprasčiausias garso paruošimas būtų gauti mūsų taškų aukštį kas tam tikrą laiko tarpą, verčiant nepertraukiamo laiko garso signalą į tam diskretaus laiko garso signalą. Tai vadinama imties dažnių atranka (angl. - „Sampling"). Dabartiniai kompaktiniai diskai įrašo garsą su imties dažniu lygiu 44.1kHz (44100 perskaitymų per sekundę), tačiau klasifikavimo uždaviniams užtenka ir 16kHz atrankos normos įvairiems garso atpažinimo uždaviniams (faktiškai dėl to, jog pats žmogus gali girdėti vidutiniškai iki tokio dažnio) [RH11].

Kaip minėta [SP14], neapdorotas garso signalas, sudarytas iš daug garso bangų, nėra tinkamas tiesioginiam naudojimui klasifikavimo uždaviniuose dėl aukšto dimensiškumo bei dėl to, kad mums suvokiami panašūs garso signalai (vektoriai) vektorių erdvėje nebūtinai būtų vienas šalia kito. Dėl šios priežasties gaunami audio duomenys yra konvertuojami į spektrogramines jų reprezentacijas prieš naudojimą, ir vėliau iš šių spektrogramų (ar panašių garsinio signalo reprezentacijų) mes gauname reikalingas charakteristikas klasifikavimo uždavinių naudojimui.

Spektrograma – garso dažnių spektro vaizdavimas laiko atžvilgiu. Norint gauti garso signalo spektrogramą, mes atliekam Furjė transformaciją kiekvienam laiko momentui, norint gauti dažnius, iš kurių susidaro mūsų nagrinėjamas laiko momentas garso signale. Tai mums grąžina amplitudinį bei fazinį spektrą. Norint nagrinėti garso signalo charakteristikas, kurios priklauso nuo laiko, mes atliekame Furje transformaciją persidengiantiems segmentams [PP08].

Furjė transformacija - procesas, kurio metu gauta muzikinė funkciją (bangą/dažnį) yra transformuojama į jos sudedamąsias. Pavyzdys būtų muzikinio akordo transformacija į jį sudedančias natas.

Toliau aptarsime pagrindines charakteristikas (ZCR, MFCC, Chroma, Mel), kurias vėliau naudosime apmokant neuroninius tinklus.

4.1. Perėjimas per nulį

Angl. - „Zero-Crossing rate” (toliau - ZCR) matuoja, kiek kartų nagrinėjamas signalo ženklas pasikeičia duotajame kadrų tarpe. ZCR suteikia informacijos apie spektrą nenaudojant daug resursų skirtinguose šios charakteristikos pritaikymo atvejuose. Paprastiems signalams, (pvz. sinusoidei), ZCR tiesiogiai susijęs su fundamentaliuoju/pagrindiniu tonu (fundamentalu - dominuojančiu tonu signale) - periodinės bangos žemiausias dažnis. Sudėtingesniems signalams, ZCR koreliuoja su vidutiniu signalo dažniu.

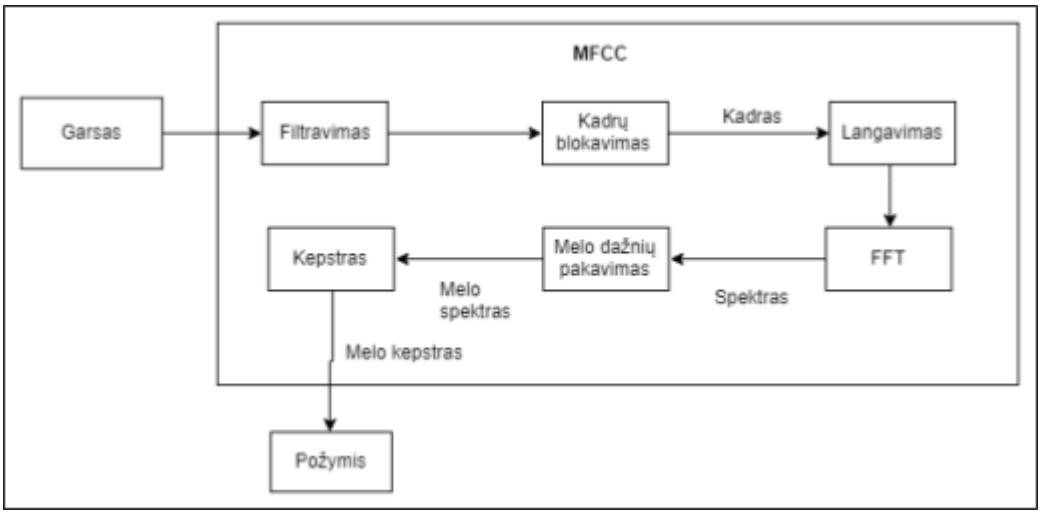
ZCR galime aprašyti pagal formulę:

$$ZCR = \frac{1}{2(N-1)} \sum_{n=2}^{N} |sgn(s(n)) - sgn(s(n-1))|, \quad (4)$$

kur n yra diskretus laiko indeksas, N yra nagrinėjamo kadro dydis, $s(n)$ yra signalo reikšmė nagrinėjamame laiko indekse n, o $sgn(x)$ grąžinamos reikšmės būna 1, kai $s(x) \geq 0$ ir -1, kai $s(x) < 0$.

4.2. MFCC

Angl. - „Mel-Frequency Cepstral Coefficients” - Melų dažnių skalės kepstro koeficientai (toliau - MFCC), kurie sudaro Melo dažnių kepstrą. MFCC bando atkartoti žmogaus garso suvokimą, tad jis yra itin geras indikatorius padedantis garso klasifikacijos uždaviniuose naudojant dirbtinį intelektą. MFCC apskaičiavimą galime pavaizduoti šiuo grafiku, pasiremtu procesu, aprašytu [Ang18]:



7 pav. MFCC procesas

Taigi, algoritmas, norint apskaičiuoti įvesties garso MFCC būtų:

1. Filtravimas - šio žingsnio tikslas yra išgauti lygesnę spektrinę formą iš mūsų gaunamo signalo dažnių - tokiu būdu, mes galime išfiltruoti dalį pašalinių garsų perduodame signale.
2. Kadrų blokavimas - šio proceso metu garsinis signalas yra segmentuojamas į daug persidengiančių kadrų, kad nebūtų prarandami individualūs signalai. Mūsų nagrinėjamas garsinis signalas susideda iš *x[n]* kadrų. Vienas kadras turi *N* duomenų ėminių (angl. - "sample"). Tarp vieno ir sekančio persidengiančio kadro yra *M* duomenų ėminių, kad *M* < *N,* kai *N = 2*M*.
3. Langavimas yra procesas kurio metu yra bandoma sumažinti pertraukimus patiriamus tarp skirtingų kadrų.
4. Greitoji Furjė transformacija (FFT) - funkcija, kurios metu yra konvertuojamas kadras, praėjęs langavimo procesą į jo dažnių spektrą. FFT yra greitas Diskrečiosios Furjė Transformacijas (DFT) algoritmas, o tai yra naudinga norint konvertuoti kiekvieną kadrą į *N* ėminių iš laiko į jo dažnių domeną.
5. Melo dažnių pakavimas į logaritminę skalę (Melo pasiskirstymo filtro banko skaičiavimas) - norint atkartoti žmogaus ausies garso suvokimo percepciją, reikia žinoti, jog žmogus garsų dažnių nejaučia tiesiškai, tačiau yra atvaizduojama Melo skale - dažniai, kurie yra mažesni negu 1000 Hz yra atvaizduojami tiesiškai, o didesni negu 1000 Hz - logaritmiškai. Tam naudojame Melo skalės funkciją:
$$Mel = 2595 * log_{10}(1 + \frac{f}{700}), \quad (5)$$
kur *f* yra tuo metu nagrinėjamas dažnis.
6. Kepstras - gaunamą Melo spektrą mes konvertuojame atžvilgiu į laiką naudodami Diskrečiąją Kosinuso Transformaciją (toliau - DCT). Gaunamas rezultatas ir yra mūsų norimas Melo-dažnių skalės kepstro koeficientas (MFCC).

4.3. Chromagrama

Chromagrama yra būdas atvaizduoti garsus, kai bendras garsų spektras yra skaidomas į 12 skyrių, kurie atvaizduoja 12 atskirų pustonių. Ši charakteristika yra dažnai naudojama muzikos atpažinime, nes žinant chromų pasiskirstymą tačiau neturint informacijos apie naudotą oktavą, yra įmanoma išgauti naudingos informacijos apie patį signalą, o tai naudoti klasifikavimui (pavyzdžiui automatiniam dainų grupavimui pagal žanrus) [CU07].

4.4. Melo spektrograma (angl. „Mel spectrogram”)

Melo spektrograma stengiasi atvaizduoti akustinę laiko-dažnio garso reprezentaciją Melo skalėje.

5. Dirbtinių neuroninių tinklų implementavimas

DNN, CNN ir RNN implementavau naudodamas Python 3.7.3 programavimo kalbą Google Colaboratory aplinkoje. Apdorojant duomenis bei programuojant tinklus naudojau šias bibliotekas:

- librosa - duomenų charakteristikų (ZCR, MFCC, Chroma bei Melo spektrogramų) gavimui;
- numpy - veiksmams su matricomis (pvz.: kuriamos charakteristikos matricos požymių nustatymui, formatavimui);
- keras - neuroninių tinklų konstravimui naudojantis Tensorflow API;
- matplotlib, seaborn - grafiniam duomenų atvaizdavimui;
- pickle - charakteristikų saugojimui tarpiniams rezultatams;

5.1. Google Colaboratory

Garsinių failų apdorojimą bei neuroninių tinklų apmokymą vykdžiau Google Colaboratory aplinkoje. Ši aplinka leidžia nemokamai naudotis jų suteikiamomis GPU paslaugomis iki 12 valandų vienu kartu. Tai itin patogu, nes suteikiamas virtualus serveris yra galingas, kad būtų galima juo naudotis įvairioms mašinos mokymosi užduotims neapkraunant asmeninio kompiuterio, o implementuojant įvairius tarpinius rezultatų išsaugojimus 12 valandų limitas gali būti apeinamas.

Tolimesnis darbas ir eksperimentai buvo visi atlikti naudojantis šia aplinka su GPU akseleravimu. Toliau pateikta lentelė, pavaizduojanti suteikiamos aplinkos parametrus:

Procesorius (CPU)	Xeon Processors @2.3Ghz
Vaizdinė plokštė (GPU)	Tesla K80, 2496 CUDA cores, compute 3.7, 12GB GDDR5 VRAM
Operatyvioji atmintis (RAM)	13 GB

1 lentelė. Google Colaboratory aplinkos parametrai

5.2. Duomenų bazė „UrbanSound8K”

Mūsų nagrinėjami pavyzdžiai bus kasdieniniai miesto garsai, paimti iš UrbanSound8K duomenų bazės, kurią sudaro 8732 mažesnių nei 4 sekundžių ištraukų iš įvairių miesto garsų šaltinių (oro kondicionierius, automobilio įspėjamasis signalas, žaidžiantys vaikai, lojantis šuo, gręžimas, automobilio variklio veikimas, pneumatinė drėlė, sirena, gatvės muzika) yra suskirstyti į 10 atskirų aplankų mokymui, mokymo validacijai ir testavimui. Garso pavyzdžiai buvo paimti iš „Freesound” archyvo. Pasirinkti garso šaltiniai buvo paimti iš taksonomijos, parinktos pagal skundus gautus dėl triukšmo 2010 metais Niujorke. Šiuo atveju, nagrinėjami garsai nėra sukurti skaitmeniniu būdu, o įrašyti, tad yra galimybė, kad vienoje ištraukoje gali būti girdimas daugiau

nei vienas garso šaltinis. Visos ištraukos buvo anotuotos ranka su šaltinio identifikavimo numeriu bei nuosprendžiu, ar garso šaltinis yra pirminis ar ne.

Duomenų pasiskirstymą per suteiktus aplankus galime pavaizduoti šia lentele:

	index	jackhammer	children_playing	drilling	air_conditioner	street_music	dog_bark	engine_idling	siren	car_horn	gun_shot
0	fold1	120	100	100	100	100	100	96	86	36	35
1	fold2	120	100	100	100	100	100	100	91	42	35
2	fold3	120	100	100	100	100	100	107	119	43	36
3	fold4	120	100	100	100	100	100	107	166	59	38
4	fold5	120	100	100	100	100	100	107	71	98	40
5	fold6	68	100	100	100	100	100	107	74	28	46
6	fold7	76	100	100	100	100	100	106	77	28	51
7	fold8	78	100	100	100	100	100	88	80	30	30
8	fold9	82	100	100	100	100	100	89	82	32	31
9	fold10	96	100	100	100	100	100	93	83	33	32

2 lentelė. Duomenų paskirstymas per aplankus duomenų bazėje UrbanSound8K.

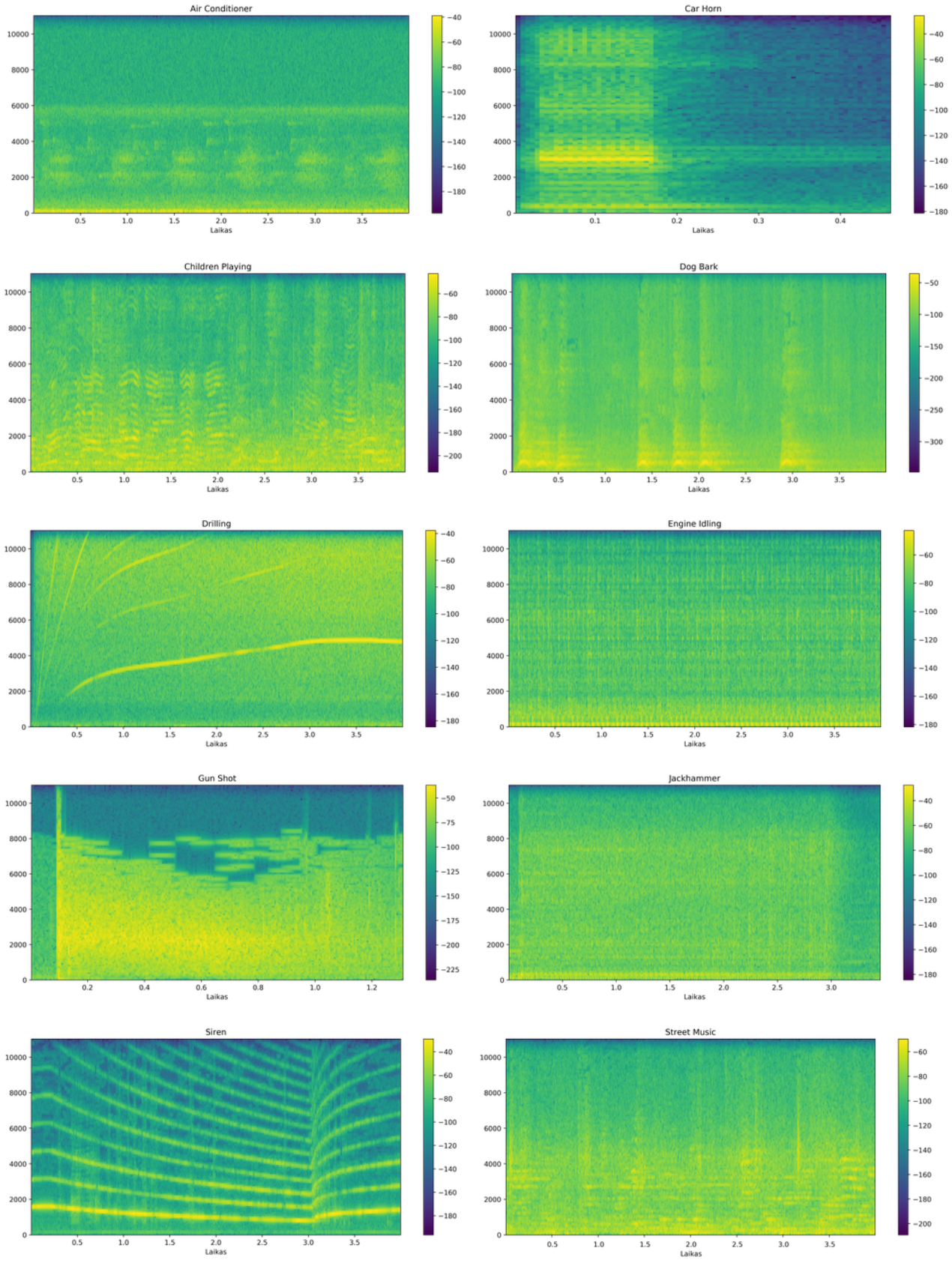
Žinant mūsų turimus duomenis, galime rasti skirtingų pavyzdžių bendrą pasiskirstymą:

Žymės indeksas	Pavyzdžio žymės pavadinimas	Bendras pasiskirstymas
0	dog_bark	0.114521
1	children_playing	0.114521
2	street_music	0.114521
3	jackhammer	0.114521
4	drilling	0.114521
5	engine_idling	0.114521
6	air_conditioner	0.114521
7	siren	0.106390
8	car_horn	0.049130
9	gun_shot	0.042831

3 lentelė. Duomenų bazės UrbanSound8K pavyzdžių pasiskirstymas.

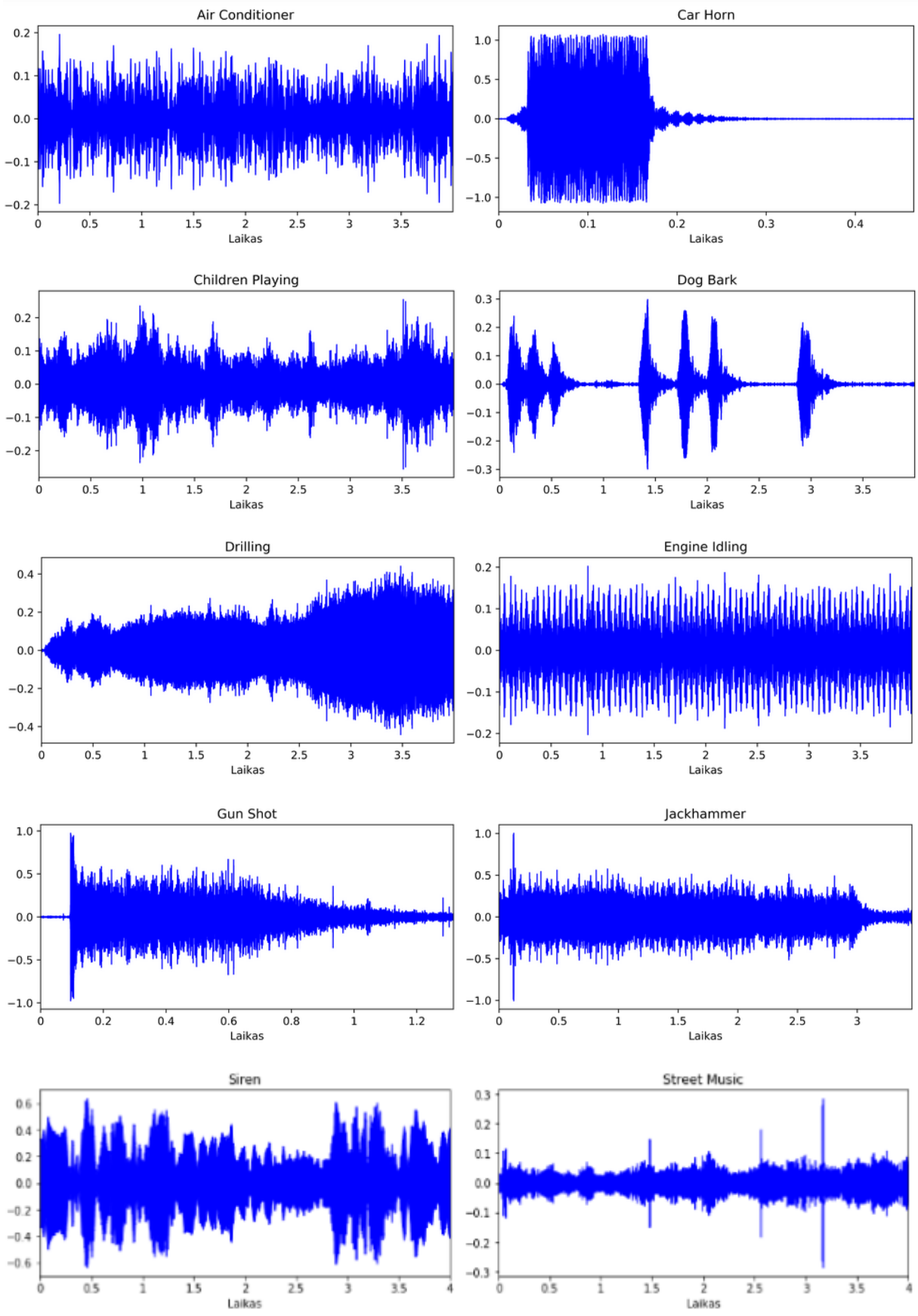
Matome, jog šautuvo šūvių bei mašinos įspėjamojo signalo ėminių yra per pus mažiau negu kitų - po 4-5 %, kai visų likusių pavyzdžių yra po 11%.

Galime pavaizduoti kiekvienos klasės spektrogramas:



8 pav. Miesto garsų klasių spektrogramos.

Bei kiekvienos klasės pavyzdinius amplitudinius grafikus:



9 pav. Miesto garsų klasių amplitudiniai grafikai.

Analizuodami gautas spektrogramas bei amplitudinius grafikus mūsų nagrinėjamiems

duomenims mes matome, jog išsiskiria kai kurie bruožai – garsams kaip gatvės muzika, oro kondicionierius, gręžimas pastebime monotoniškumą, taip pat ir pastebime momentą, kai sirena iš naujo pradeda skleisti garsą. Matome aiškius atskirus momentus, kai šuo pradeda loti ar yra iššaunamas šūvis, kada pradeda veikti mašinos įspėjamasis signalas. Tačiau vien tik analizuojant spektrogramas mes negalime pastebėti ryškių skirtumų tarp mašinos variklio bei gręžimo garso, o tam mes pasitelksime neuroninius tinklus bei kitas charakteristikas, kurios nėra vien tik matomos spektrogramose.

Šalia 10 aplinkų, kuriuos sudaro garsiniai pavyzdžiai .wav formatu, yra pateiktas .csv formato failas, kuriame yra kiekvieno .wav pavyzdžio žymuo.

Mes gautus duomenis naudosime kaip nurodyta [SJB14] - 9 aplankus naudosime neuroninio tinklo apmokymui, dalį šių duomenų iš 9 aplankų (20% iš šių 9 aplankų) validavimui, o likusį aplanką naudosime testavimui. Tai atliksime 10 kartų ("10-fold cross validation"), o galutinis rezultatas (teisingo klasifikavimo procentas) bus bendras tinklo įvertinimas - kiek teisingai (procentaliai) gali mūsų tinklas klasifikuoti gautą garsą į 10 klasių. Šalia bendro tinklo įvertinimo taip pat matysime minimalius ir maksimalius mokymo bei testavimo metu pasiektus klasifikavimo tikslumus.

5.3. Duomenų paruošimas

5.3.1. Charakteristikų gavimas ir jų išsaugojimas

Dirbant Google Colaboratory aplinkoje, visi garsiniai pavyzdžiai buvo sukelti į Google Drive aplanką. Autorizavus Google Colaboratory prieigą prie asmeninės Google Drive paskyros, per šiuos failus buvo iteruojama ir išgaunamos reikalingos charakteristikos, besinaudojant librosa bibliotekos funkcijomis. Buvo sukurti 2 stekai - *fold<numeris>_feat* (tam tikro aplanko charakteristikų) bei *fold<numeris>_label* (tam tikro aplanko atskirų pavyzdžių žymių). Tam, kad nereikėtų kas kartą iš naujo iteruoti pro failus (nes tai užtrukdavo iki 4 valandų) renkant jų charakteristikas, atskiros charakteristikos bei žymės buvo saugomos .npy formatu pačiame Google Drive aplanke. Norėdami mokyti dirbtinio neuroninio tinklo modelį, paprasčiausiai užkrauname šias išsaugotas charakteristikas naudodamiesi 'pickle' biblioteka.

5.3.2. Žymių paruošimas naudojant „One-hot encoding"

Surinkus informaciją apie atitinkamus duomenų pavyzdžius ir jų žymes, reikėjo šiuos duomenis užkoduoti prieš leidžiant jį naudoti DNT mokymui. Šiam tikslui buvo naudojamas „One-hot encoding" (toliau - OHE), norint pakeisti kategorines žymes (0-9, kur 0 = *dog_bark*, 1 = *children_playing* ir t.t.) į numerines žymes.

Tai galime suprasti geriau nagrinėdami mūsų duomenis - be OHE, mūsų žymes galime suprasti kaip vienmatę matricą, kur kiekvienas indeksas yra atitinkantis nagrinėjamą pavyzdį, o skaičius šiame indekse atvaizduoja minėtą klasę, kuriai priklauso pavyzdys. Tai galime pavaizduoti 1 × N dydžio matrica:

```
[2 7 1 ... 2 0 2];
```

Atlikus OHE operaciją šiam masyvui, mūsų rezultatas būtų 10 × N matrica:

```
[[0. 0. 1. ... 0. 0. 0.]
 [0. 0. 0. ... 1. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 ...
 [0. 0. 1. ... 0. 0. 0.]
 [1. 0. 0. ... 0. 0. 0.]
 [0. 0. 1. ... 0. 0. 0.]];
```

6. Tyrimas

Šiame skyriuje aptarsime kuriamų dirbtinių neuroninių tinklų parametrus bei rezultatus testuojant su mūsų minėta duomenų baze. Toliau minimi sutrumpinimai, kaip DNN-256-256 leidžia mums atskirti vieną neuroninio tinklo modelį nuo kito. DNN-256-256 pažymi, jog tai yra gilusis dirbtinis neuroninis tinklas, kurį sudaro 2 paslėptieji lygmenys po 256 neuronus.

Modelio įvertinimą sudarys jo maksimalus, vidutinis bei minimalus mokymosi ir atskirai - testavimo metu pasiektas atpažinimo tikslumas. Kaip ir minėta, bus atliekama 10 aplankų validacija - kiekviena iteracija bus paimami 9 aplankai su duomenimis mokymui, iš šių 9 aplankų 20% atsitiktinai atrinktų duomenų bus naudojami tarp epochų validavimo tikrinimui, o testavimas vyks ant 10 aplanko. Iteruosime per visus 10 aplankų, ir grąžinsime gautus rezultatus.

Šalia gautų tikslumo rezultatų bus taip pat pateikiama normalizuota maišos matrica, sugeneruota tuo atveju, kai testavimo rezultatas yra artimiausias vidutiniam testavimo tikslumo rezultatui. Maišos matrica - būdas pamatyti, kokius spėjimus daro mūsų modelis ir kiek tiksliai jis sugeba atskirti pažymėtus duomenis, bei su kuo dažniausiai sumaišo spėjimus.

6.1. Gilieji dirbtiniai neuroniniai tinklai

Visuose toliau aptariamuose DNN modeliuose įvesties sluoksnis susidarė iš 181 neuronų, dėl to, kad mūsų gautos garso charakteristikos buvo 181 × 7895 dimensijų masyvas. Mūsų garso charakteristikos bendras vektorius buvo sudarytas iš dydžių: 40 MFCC, 1 ZCR, 12 Chromogramos, 128 Melo spektrogramos.

Paslėptieji sluoksniai susidarė abu iš 256 neuronų, o tarp jų buvo išmetimo santykis lygus (angl. „dropout rate”) 40%. Išvesties sluoksnis buvo sudarytas iš 10 neuronų, nes mūsų užduotis buvo klasifikuoti garsus į 10 klasių. Tarp giliųjų sluoksnių buvo naudojama „ReLU” aktyvavimo funkcija, o išvesties aktyvavimo funkcija buvo „Softmax”. Taip pat kompiliavimo metu buvo optimizuojama naudojant „adam” algoritmą. Iš viso mūsų tinklas turėjo 78 858 kintančius parametrus.

Lyginant modelius tarpusavyje, mūsų įeities ir išeities modelių neuronų kiekiai nesiskirs, nes skirtingiems modeliams bus suteikiami tie patys duomenys.

Modeliai turėjo nustatytas veikimo ribas - arba kol pasieks 500 epochų, arba iki kol sustojo mokytis dėl prastėjančių validavimo rezultatų (persimokymo). Persimokymas - procesas, kurio metu mašinų mokymosi algoritmas pernelyg pripranta prie nagrinėjamų duomenų, ir vietoj to, jog ieškotų kažkokių koreliacijų ar paslėptų šablonų, galima sakyti, jog jis „įsimena” teisingus atsakymus. Tai galima patikrinti naudojant validavimo duomenis, kurie, jei pradeda kilti „loss” funkcijos reikšmė (ką galime suprasti kaip algoritmo „pasitikėjimą” darant spėjimą, tai šis algoritmas pradeda persimokyti. Tai buvo daroma naudojant Keras „EarlyStopping monitor” funkcija, kuri, jei modelis tam tikrą epochų skaičių (mes pasirinkome 10) nesumažina validavimo ‘loss’ funkcijos reikšmės, nutraukiamas mokymas.

Dirbtinio neuroninio tinklo apmokymas buvo vykdomas naudojant ~8000 pavyzdžių, o validuojama ir testuojama su likusiais ~800-ais pavyzdžių atliekant 10 aplankų validavimą.

6.1.1. DNN-256-256

Pirmąjį mūsų modelį sudarė 2 paslėptieji sluoksniai, turintys po 256 neuronus.

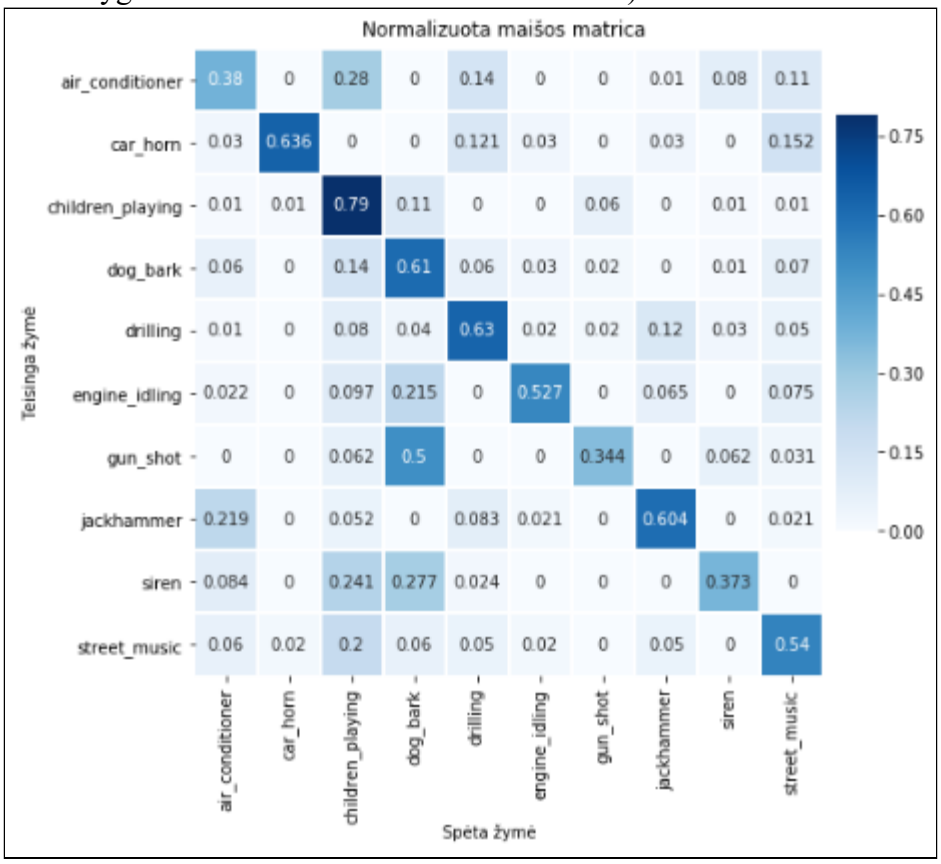
Atlikus 10 aplankų validavimą, buvo gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 82.94%
- Minimalus mokymo metu pasiektas tikslumas buvo 75.81%
- Vidutinis mokymo metu pasiektas tikslumas buvo 79.16%

- Maksimalus testavimo metu pasiektas tikslumas buvo 62.60%
- Minimalus testavimo metu pasiektas tikslumas buvo 46.05%
- Vidutinis testavimo metu pasiektas tikslumas buvo 57.14%
- Viso testo trukmė - 22 minutės.

Nenaudojant išankstinių tinklo mokymosi nutraukimų ir paliekant tinklą mokytis ilgiau (50 ar daugiau epochų), tinklas grąžina mums prastesnius validavimo, bet geresnius mokymo metu pasiekiamus rezultatus, ką galime suprasti kaip tinklo persimokymą.

Galime matyti sugeneruotą klasifikavimo maišos matricą testavimo atvejams, o parinktas buvo tas k-fold validavimo scenarijus, kai validavimo tikslumas buvo 55% (arčiausiai pasikartojęs tikslumas lyginant su vidutiniu validavimo tikslumu):



10 pav. DNN-256-256 maišos matrica

Galime nesunkiai pastebėti, jog mūsų sukonfigūruotas giliajam neuroniniam tinklui daugiausiai bėdų sukėlė bandymas atskirti ginklo šūvio garsą nuo šuns lojimo - ir apmokymo, ir testavimo metu galime matyti pasiekiamą mažesnį tikslumą.

6.1.2. DNN-256-256-256

Šis tinklas buvo sukurtas pažiūrėti, ar pridedant papildomą paslėptąjį sluoksnį su tiek pat neuronų padėtų pagerinti prieš tai peržiūrėto neuroninio tinklo atpažinimo tikslumą. Tarp sluoksnių išlieka 40% išmetimo santykis.

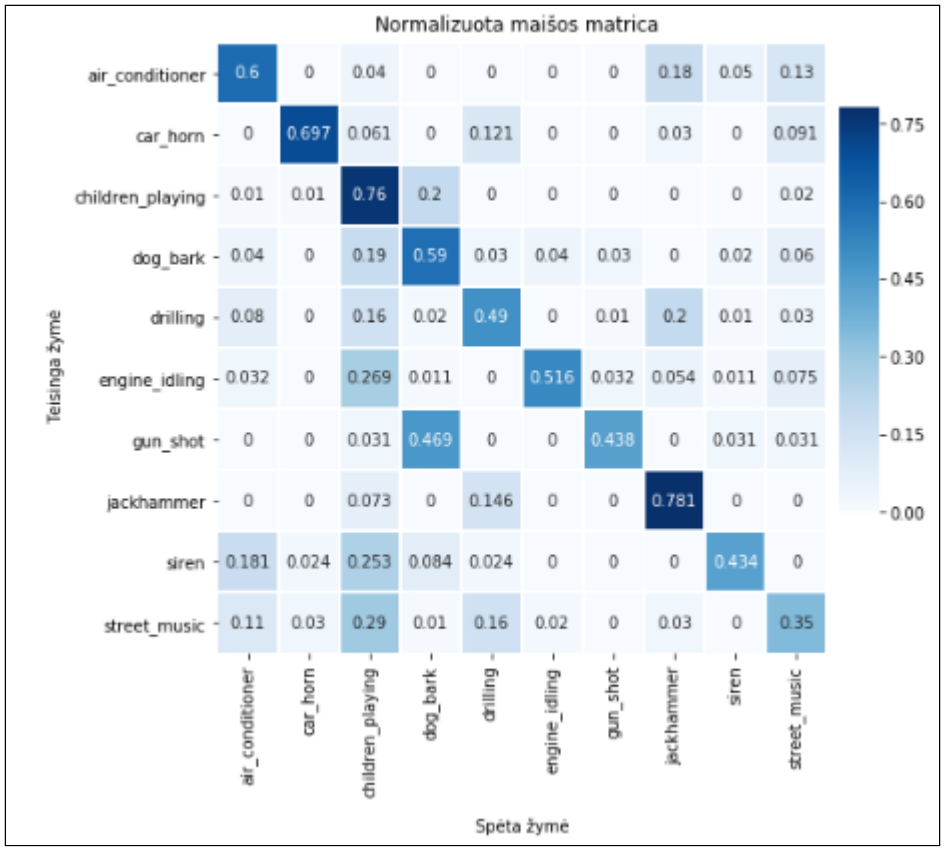
Atlikus 10 aplankų validavimą, buvo gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 74.89%
- Minimalus mokymo metu pasiektas tikslumas buvo 69.32%
- Vidutinis mokymo metu pasiektas tikslumas buvo 72.62%
- Maksimalus testavimo metu pasiektas tikslumas buvo 56.43%
- Minimalus testavimo metu pasiektas tikslumas buvo 47.21%
- Vidutinis testavimo metu pasiektas tikslumas buvo 54.75%

- Viso testo trukmė - 14 minučių

Lyginant su prieš tai konstruotu modeliu, matome itin panašius atpažinimo rezultatus ir mokymo, ir testavimo metu.

Gauta maišos matrica:



11 pav. DNN-256-256-256 maišos matrica

Matome, jog didelių nuokrypių nuo prieš tai sukonstruoto modelio nematome, ir modelis susiduria su prieš tai minėtomis bėdomis bandant atpažinti panašių struktūrų garsus (gatvės muzika, gręžimas), taip pat maišo ginklo šūvio garsus su šuns lojimu.

6.1.3. DNN-256-512-256

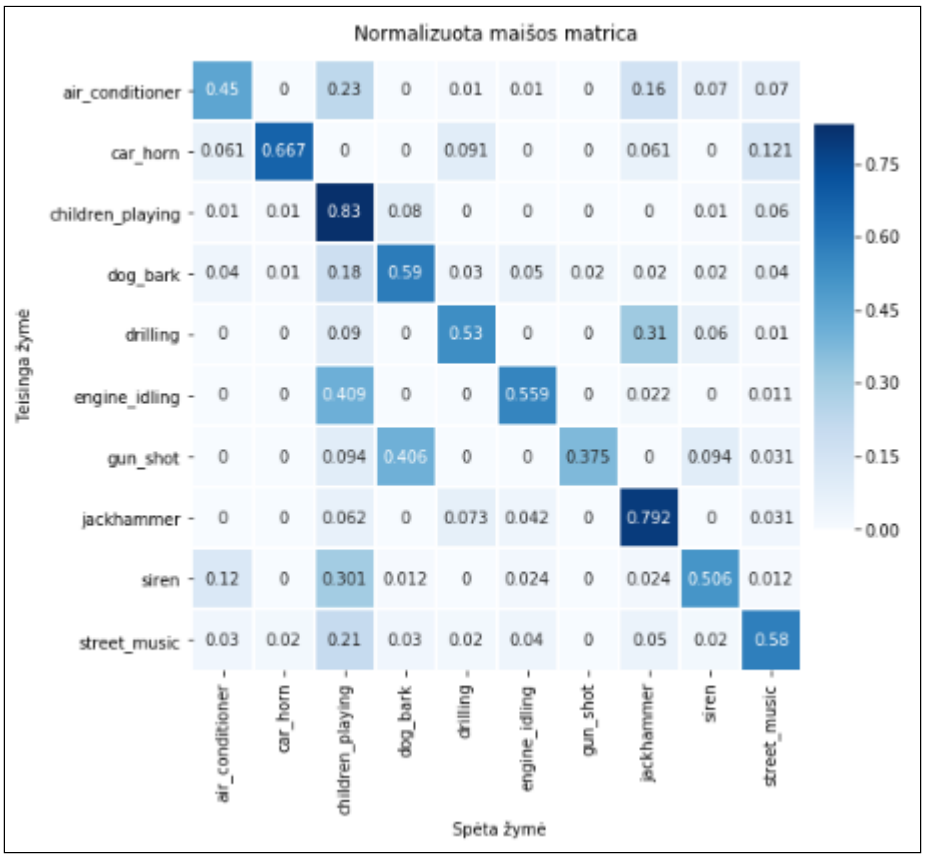
Šis modelis buvo konstruojamas norint vidurinį paslėptąjį sluoksnį turėti dvigubai daugiau neuronų, negu turi kaimynai. Tarp sluoksnių išlieka 40% išmetimo santykis.

Buvo gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 83.28%
- Minimalus mokymo metu pasiektas tikslumas buvo 71.96%
- Vidutinis mokymo metu pasiektas tikslumas buvo 76.28%
- Maksimalus testavimo metu pasiektas tikslumas buvo 66.62%
- Minimalus testavimo metu pasiektas tikslumas buvo 57.82%
- Vidutinis testavimo metu pasiektas tikslumas buvo 63.11%
- Viso testo trukmė - 32 minutės

Kaip ir prieš tai konstruoti modeliai, matome itin panašius mokymosi metu atliekamus testų rezultatus, tačiau matome bendrą modelio validavimo metu pagerėjimą atpažinimo atžvilgiu - bendru atveju, šis modelis pasiekia 5% geresnį atpažinimo tikslumą.

Gauta maišos matrica:



12 pav. DNN-256-512-256 maišos matrica

Tyrinėjant maišos matricą, toliau matome panašius atvejus, kur mūsų modelis neteisingai atspėja garso žymę. Šiuo atveju, modeliui prasčiau sekėsi atpažinti ginklo šūvio bei oro kondicionieriaus garsus.

6.1.4. DNN-512-1024-512

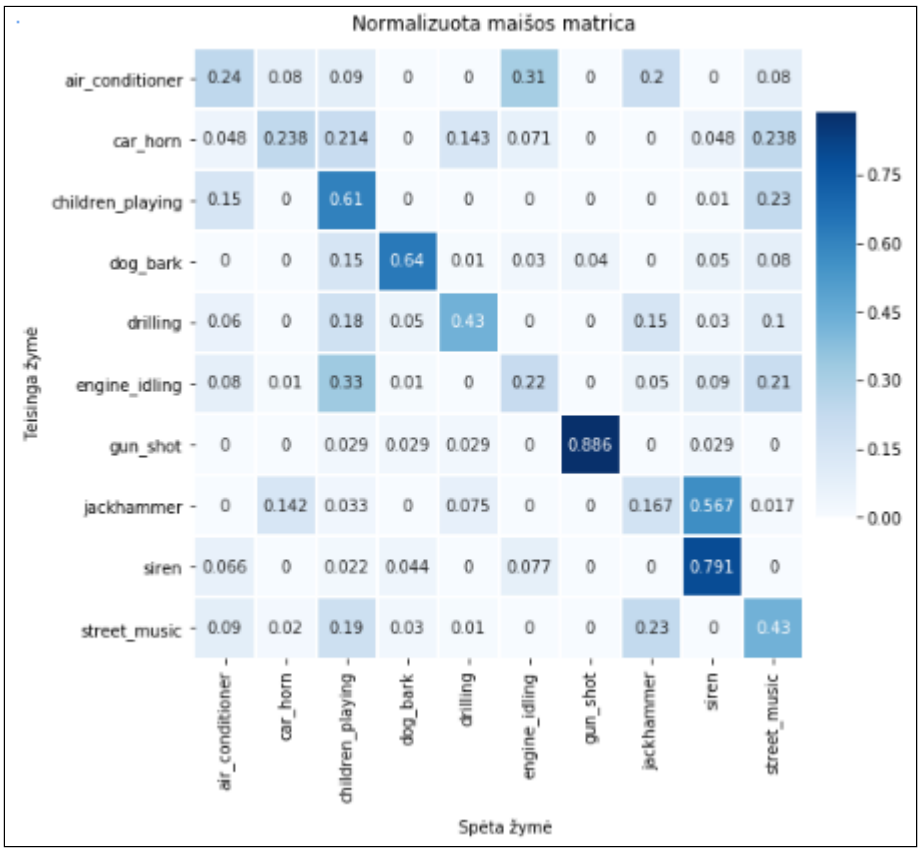
Šiuo atveju, padidinome kiekvieno sluoksnio neuronų kiekį dvigubai, tikėdamiesi, jog tai padės gauti geresnį atpažinimą. Tarp sluoksnių išlieka 40% išmetimo santykis.

Buvo gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 89.28%
- Minimalus mokymo metu pasiektas tikslumas buvo 32.43%
- Vidutinis mokymo metu pasiektas tikslumas buvo 67.99%
- Maksimalus testavimo metu pasiektas tikslumas buvo 55.86%
- Minimalus testavimo metu pasiektas tikslumas buvo 14.32%
- Vidutinis testavimo metu pasiektas tikslumas buvo 44.65%
- Viso testo trukmė - 40 minučių

Matome, jog nepasiteisino bandymas sulaukti geresnių rezultatų didinant neuronų kiekį kiekviename sluoksnyje. Įdomu, jog testuojant naudojant aplankus 1, 4 buvo gauti itin prasti rezultatai - 14.32% ir 24.55%. Galime daryti spėjimą, jog šiais atvejais tinklas “įsimindavo” atsakymus mokymosi fazėje, o testavimas parodė, jog įvyko persimokymas.

Gauta maišos matrica:



13 pav. DNN-512-1024-512 maišos matrica

Įdomu pastebėti tai, jog šiuo atveju, tinklas (lyginant su prieš tai sukurtais modeliais) gerai sugeba atpažinti ginklo šūvio bei sirenos garsus, kur prieš tai konstruoti modeliai prastai susidorodavo su šia užduotimi. Tačiau matome, jog kartojasi ir prieš tai minėti sunkumai atpažinti panašių struktūrų „monotoninius" garsus - mašinos variklio bei pneumatinės drėlės garsus.

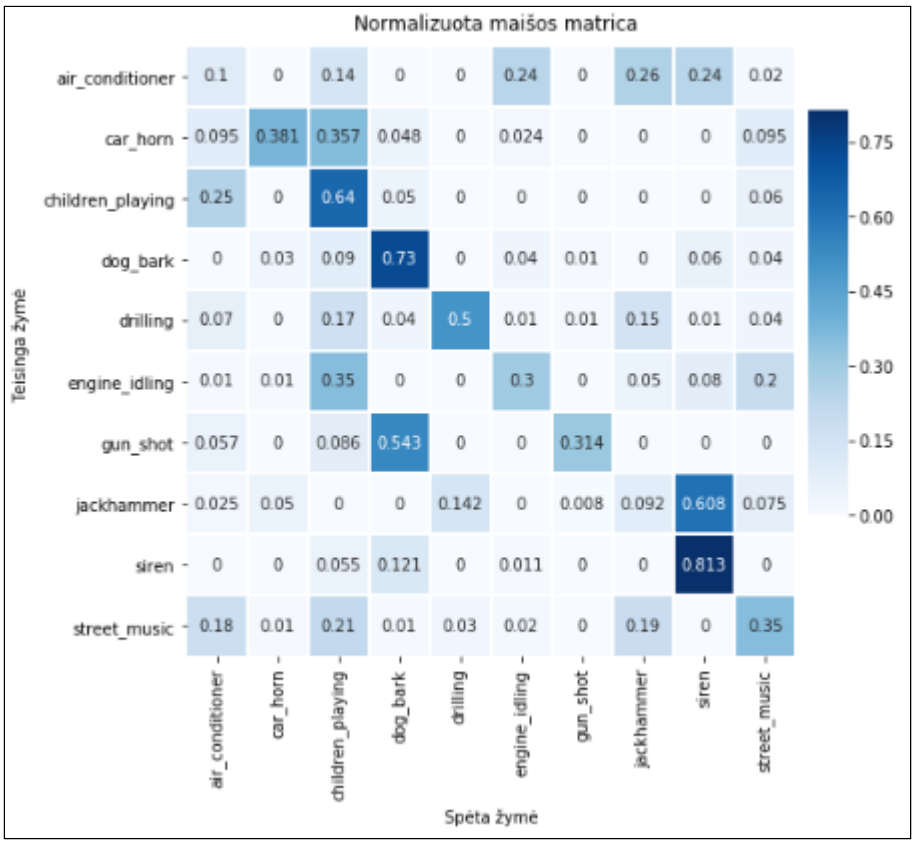
6.1.5. DNN-400-200

Šis modelis yra sudarytas iš 2 paslėptųjų sluoksnių, kuriuos sudaro 400 ir 200 neuronų. Tarp sluoksnių yra 40% išmetimo santykis.

Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 74.48%
- Minimalus mokymo metu pasiektas tikslumas buvo 68.24%
- Vidutinis mokymo metu pasiektas tikslumas buvo 72.41%
- Maksimalus testavimo metu pasiektas tikslumas buvo 60.21%
- Minimalus testavimo metu pasiektas tikslumas buvo 49.33%
- Vidutinis testavimo metu pasiektas tikslumas buvo 54.88%
- Viso testo trukmė - 11 minučių

Gauta maišos matrica:



14 pav. DNN-400-200 maišos matrica

Modelio architektūra nepasiteisino - matome pasikartojančią bėdą bandant atskirti monotoniškus garsus kaip oro konicionierius, mašinos variklio garsas, pneumatinė drėlė. Tačiau kaip ir DNN-512-1024-512, geriau sugeba atskirti garsus kaip sirena, šuns lojimas, tačiau ginklo šūvio garsą sumaišo su šuns lojimu.

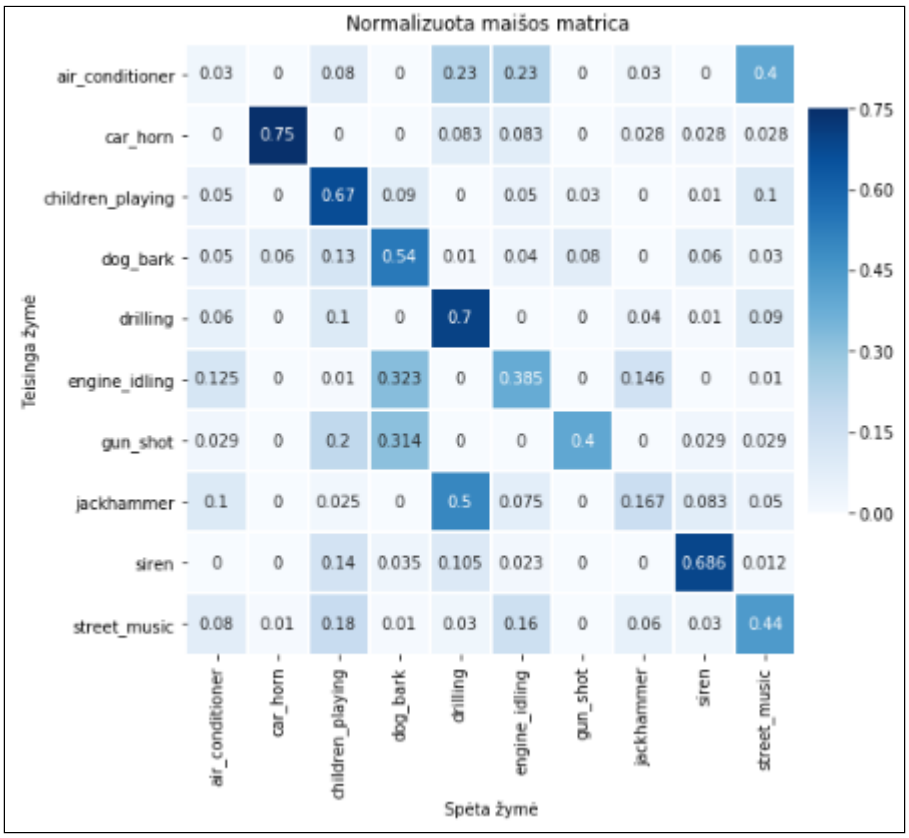
6.1.6. DNN-128-256-512-256-128

Atsižvelgiant į tai, jog kol kas sėkmingiausias tinklas vidutiniu testavimo atveju buvo DNN-256-512-25, šalia įeities ir išeities sluoksnių pridėjome po 128 neuronus turinčius sluoksnius. Tarp sluoksnių išlieka 40% išmetimo santykis.

Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 72.17%
- Minimalus mokymo metu pasiektas tikslumas buvo 63.18%
- Vidutinis mokymo metu pasiektas tikslumas buvo 68.08%
- Maksimalus testavimo metu pasiektas tikslumas buvo 55.75%
- Minimalus testavimo metu pasiektas tikslumas buvo 42.60%
- Vidutinis testavimo metu pasiektas tikslumas buvo 44.28%
- Viso testo trukmė - 63 minutės

Gauta maišos matrica:

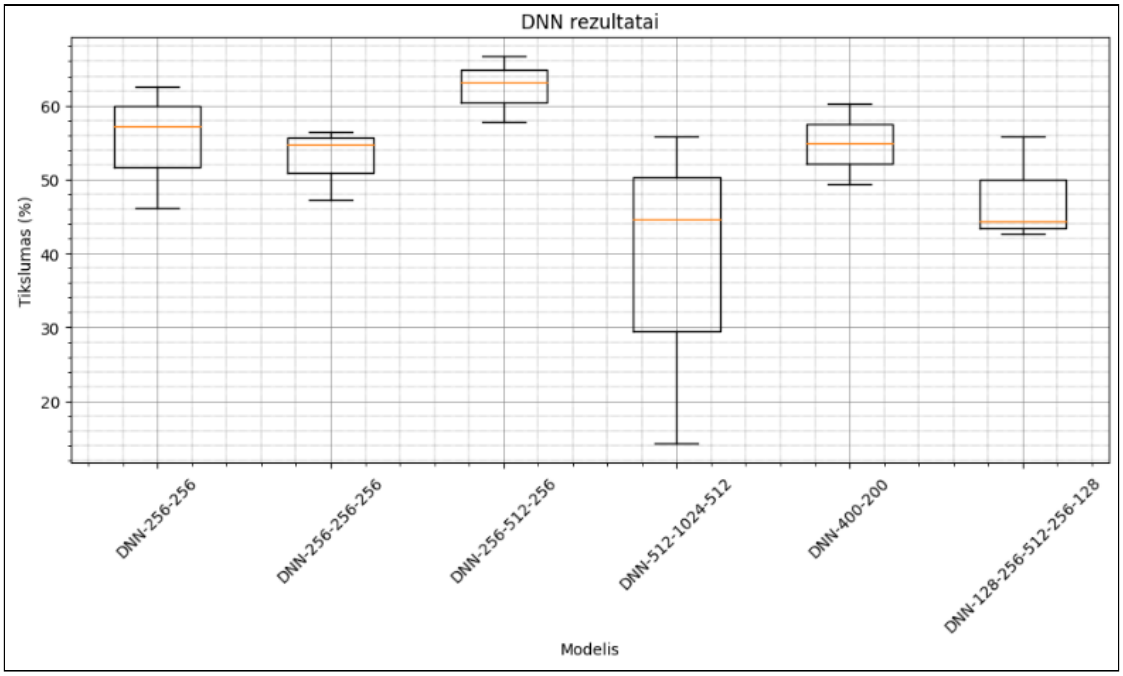


15 pav. DNN-128-256-512-256-128 maišos matrica.

Matome, jog papildomų sluoksnių pridėjimas į modelį nepadeda gauti geresnių rezultatų, o kaip tik trukdo tinklui korektiškai klasifikuoti testinius duomenis.

6.1.7. Tarpinė DNN išvada

Iš 6 mūsų sukonstruotų giliųjų dirbtinių neuroninių tinklų, atsižvelgiant į testavimo vidutinį tikslumą, geriausiai pasirodė DNN-256-512-256 modelis su vidutiniu tikslumu lygiu 63.11%.



16 pav. DNN modelių stačiakampinė diagrama.

Iš šios stačiakampinės diagramos (angl. - “boxplot”), pavaizduotos 16 pav. taip pat matome, jog minėtas DNN-256-512-256 taip pat ir turėjo didžiausią maksimalų testavimo vidurkį, lygų 66.12%.

Taip pat buvo bandytos ir kitos DNN konfigūracijos, kaip DNN-1024-1024-1024, DNN-2048-2048, tačiau jų tikslumas buvo mažiau nei 20%, tad buvo nutarta itin didelius neuronų kiekius turinčius paslėptuose sluoksniuose tinklų toliau nenagrinėti.

6.2. Konvoliuciniai dirbtiniai neuroniniai tinklai

Mūsų toliau nagrinėjami CNN įvesčiai gauna MFCC reprezentaciją. Pirmui lygmeniui suteikiama (40, 174, 1) dimensijų masyvas, kur 40 - MFCC dydis, 174 - nagrinėjamo lango dydis, 1 - požymis, kuris nusako, jog nagrinėjamas garsas yra monofoninis.

Tarp visų sluoksnių yra naudojamos ReLU aktyvavimo funkcijos, 40% išmetimo santykis. Perduodamo paketo dydis (angl. - „Batch size”) - 256.

Po kiekvieno konvoliucinio sluoksnio buvo atliekamas maksimalus telkimas (telkimo dydis = 2×2).

CNN turėjo apribojimus - dėl ilgesnio mokymosi laiko, mokymosi laikas buvo apribotas iki 100 epochų, arba iki prastėjančių validavimo rezultatų - jei modeliui nepavyksta sumažinti validavimo ‘loss’ funkcijos per 5 epochas, nutraukiamas mokymas.

Kaip ir su DNN, CNN buvo apmokytas naudojant 8000 pavyzdžių, iš kurių buvo imami 20% pavyzdžiai validavimui, o testuojamas galutinis modelis su likusiais 800 pavyzdžių 10 kartų, rotuojant pro skirtingus aplankus.

Toliau yra aprašyti mūsų konstruoti modeliai, keičiant filtrų kiekį konvoliuciniuose sluoksniuose norint gauti geresnius atpažinimo rezultatus.

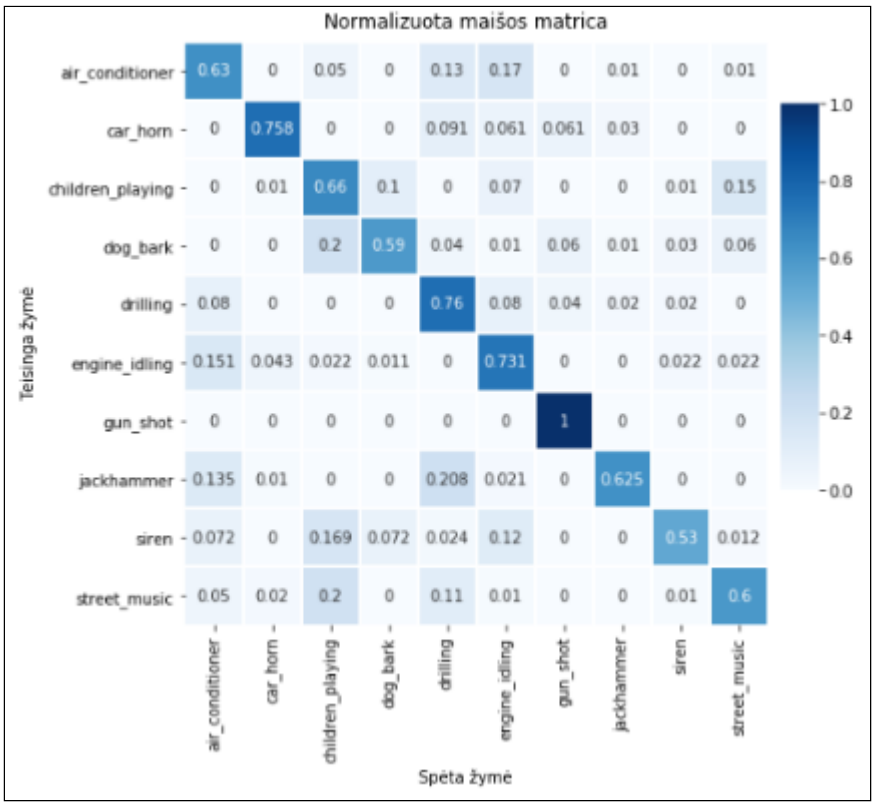
6.2.1. CNN-16-32-64-32

Konvoliucijos sluoksniai buvo sudaryti iš 16, 32, 64, 32 filtrų.

Buvo gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 80.17%
- Minimalus mokymo metu pasiektas tikslumas buvo 72.43%
- Vidutinis mokymo metu pasiektas tikslumas buvo 76.22%
- Maksimalus testavimo metu pasiektas tikslumas buvo 67.17%
- Minimalus testavimo metu pasiektas tikslumas buvo 62.23%
- Vidutinis testavimo metu pasiektas tikslumas buvo 64.31%
- Viso testo trukmė - 42 minutės

Vidutinio atvejo maišos matricos pavyzdys (šio tinklo atpažinimo tikslumas testuojant buvo 66.3%):



17 pav. CNN-16-32-64-32 maišos matrica.

Kaip matome, mūsų CNN modelis puikiai sugeba atskirti ginklų šūvio garsą - buvo paduoti 32 skirtingi pavyzdžiai, ir tinklas sugebėjo kiekvieną kartą atspėti šio garso klasę.

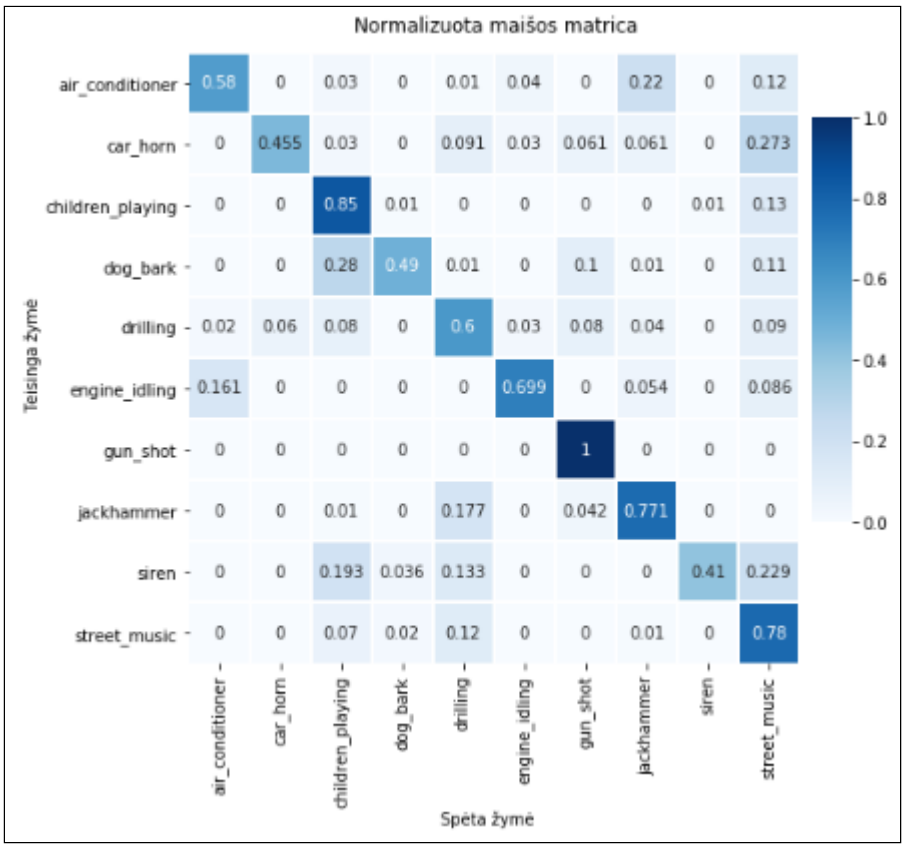
6.2.2. CNN-32-32

Norint toliau nagrinėti modelius šiai užduočiai, pasirinkome naudoti po 32 filtrus 2 konvoliucijos sluoksniuose.

Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 74.17%
- Minimalus mokymo metu pasiektas tikslumas buvo 64.43%
- Vidutinis mokymo metu pasiektas tikslumas buvo 67.71%
- Maksimalus testavimo metu pasiektas tikslumas buvo 69.17%
- Minimalus testavimo metu pasiektas tikslumas buvo 63.04%
- Vidutinis testavimo metu pasiektas tikslumas buvo 65.71%
- Viso testo trukmė - 23 minutės

Gauta maišos matrica:



18 pav. CNN-32-32 maišos matrica

Kaip ir praeitas CNN modelis, 18-ame paveikslėlyje pavaizduota maišos matrica mums parodo, jog mūsų sukonstruotas modelis puikiai sugeba atpažinti ginklo šūvių garsus. Tačiau ir monotoniškesnius - gatvės muziką, pneumatinę drėlę, variklio garsą taip pat gali atpažinti geriau, negu prieš tai naudotas modelis.

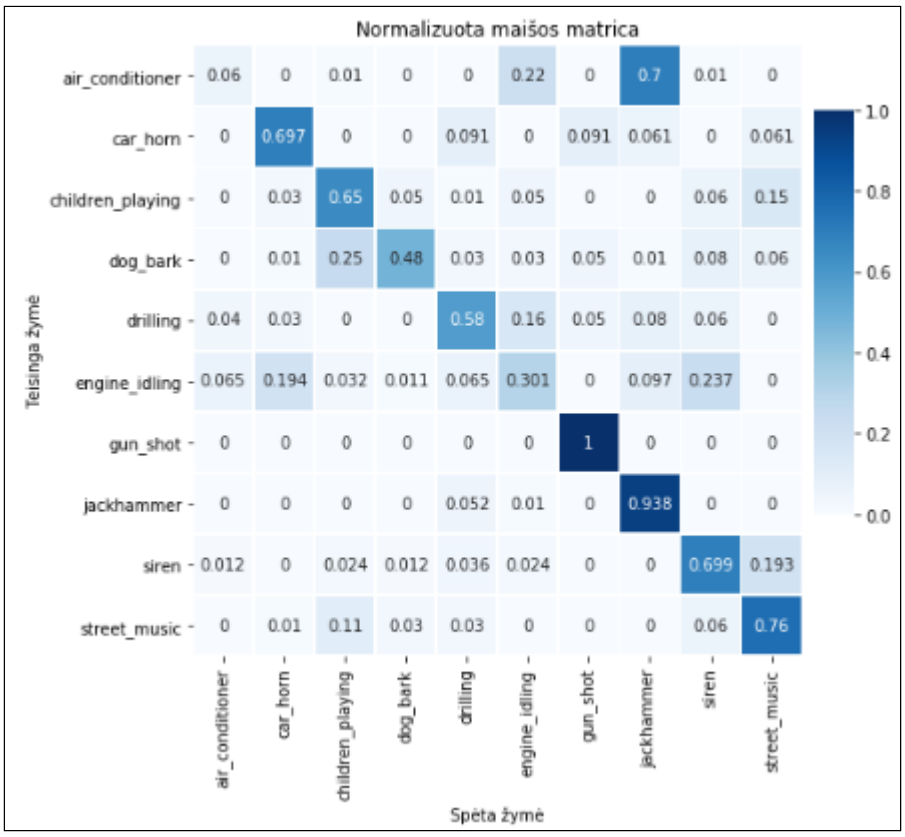
6.2.3. CNN-32-64-32

Šiuo atveju stengėmės įterpti dar vieną konvoliucijos sluoksnį, kurį sudaro 2 kartus daugiau filtrų, palyginus su aplinkiniais sluoksniais.

Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 79.54%
- Minimalus mokymo metu pasiektas tikslumas buvo 73.00%
- Vidutinis mokymo metu pasiektas tikslumas buvo 77.54%
- Maksimalus testavimo metu pasiektas tikslumas buvo 65.17%
- Minimalus testavimo metu pasiektas tikslumas buvo 63.21%
- Vidutinis testavimo metu pasiektas tikslumas buvo 60.43%
- Viso testo trukmė - 24 minutės

Gauta maišos matrica:



19 pav. CNN-32-64-32 maišos matrica

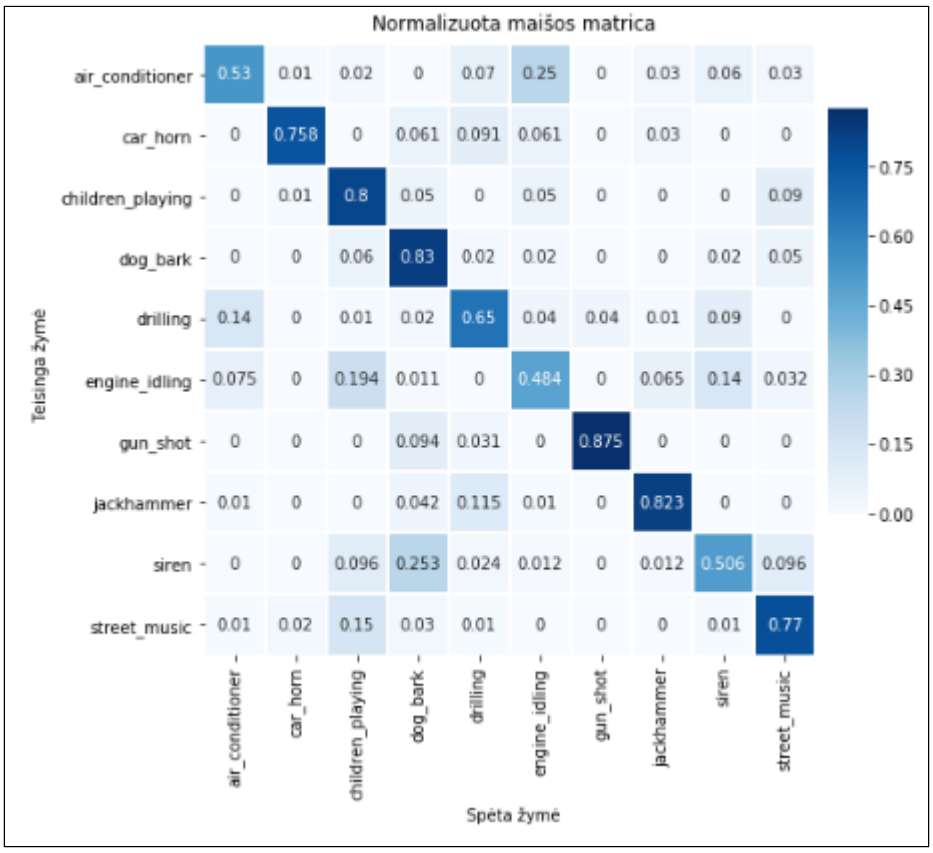
Deja, rezultatai testavimo metu yra prastesni - nors tinklas ir pasiekia didesnį mokymo metu tikslumą, testavimo metu rezultatai prastesni už prieš tai 2 sukonstruotus modelius. Maišos matrica parodo, jog šiuo atveju, pritaikyti filtrai priverčia modelį sumaišyti oro kondicionieriaus garsą su pneumatine drėle.

6.2.4. CNN-64-128-256

Bandėme pakeisti konvoliucijos sluoksnius, jog juos sudarytų 64, 128, 256 filtrai.
Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 74.03%
- Minimalus mokymo metu pasiektas tikslumas buvo 68.71%
- Vidutinis mokymo metu pasiektas tikslumas buvo 70.61%
- Maksimalus testavimo metu pasiektas tikslumas buvo 72.44%
- Minimalus testavimo metu pasiektas tikslumas buvo 57.95%
- Vidutinis testavimo metu pasiektas tikslumas buvo 69.49%
- Viso testo trukmė - 31 minutė

Gauta maišos matrica:



20 pav. CNN-64-128-256 maišos matrica

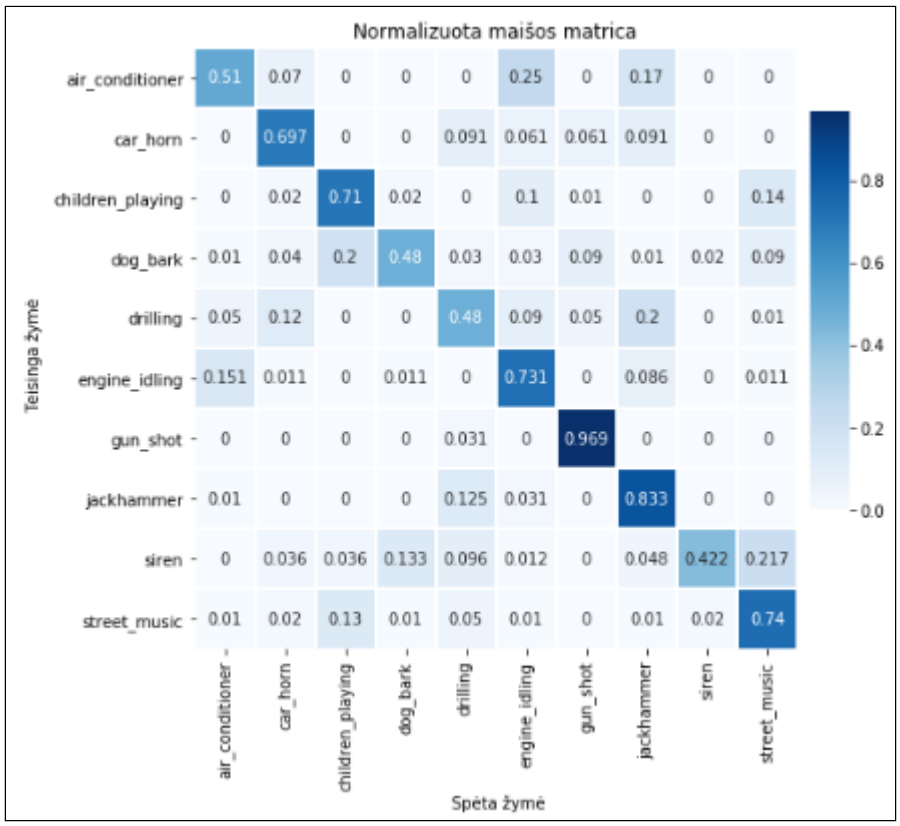
Iš 20-o paveikslėlio matome, jog mūsų sukonstruotas modelis gerai sugeba atpažinti daugumą garsų, o su tais garsais, kurių atpažinimo tikslumas nėra toks aukštas, kaip sirena, variklio garas ar oro kondicionierius - nėra itin pastebimos maišos, t.y. yra gan panašus atpažinimo metu garsų susimaišymo pasiskirstymas.

6.2.5. CNN-128-256-512

Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 85.60%
- Minimalus mokymo metu pasiektas tikslumas buvo 78.54%
- Vidutinis mokymo metu pasiektas tikslumas buvo 84.32%
- Maksimalus testavimo metu pasiektas tikslumas buvo 69.91%
- Minimalus testavimo metu pasiektas tikslumas buvo 65.35%
- Vidutinis testavimo metu pasiektas tikslumas buvo 68.94%
- Viso testo trukmė - 35 minutės

Gauta maišos matrica:



21 pav. CNN-128-256-512 maišos matrica

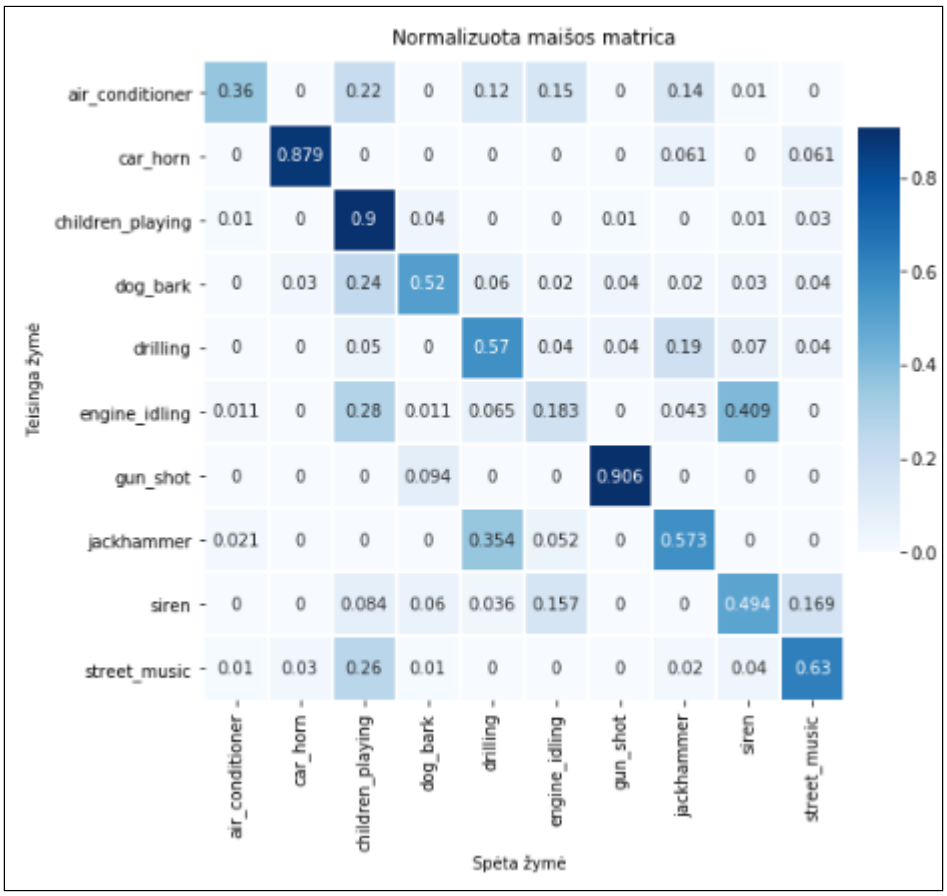
Maišos matricoje, pavaizduotoje 21 pav. matome, jog sukonstruotas modelis neblogai atpažįsta įvairaus tipo garsus - nėra atvejų, kai tikslumas mažesnis nei 40%.

6.2.6. CNN-128-256-512-1024

Gauti rezultatai:

- Maksimalus mokymo metu pasiektas tikslumas buvo 84.51%
- Minimalus mokymo metu pasiektas tikslumas buvo 79.53%
- Vidutinis mokymo metu pasiektas tikslumas buvo 82.77%
- Maksimalus testavimo metu pasiektas tikslumas buvo 60.22%
- Minimalus testavimo metu pasiektas tikslumas buvo 54.36%
- Vidutinis testavimo metu pasiektas tikslumas buvo 58.94%
- Viso testo trukmė - 56 minutės

Gauta maišos matrica:

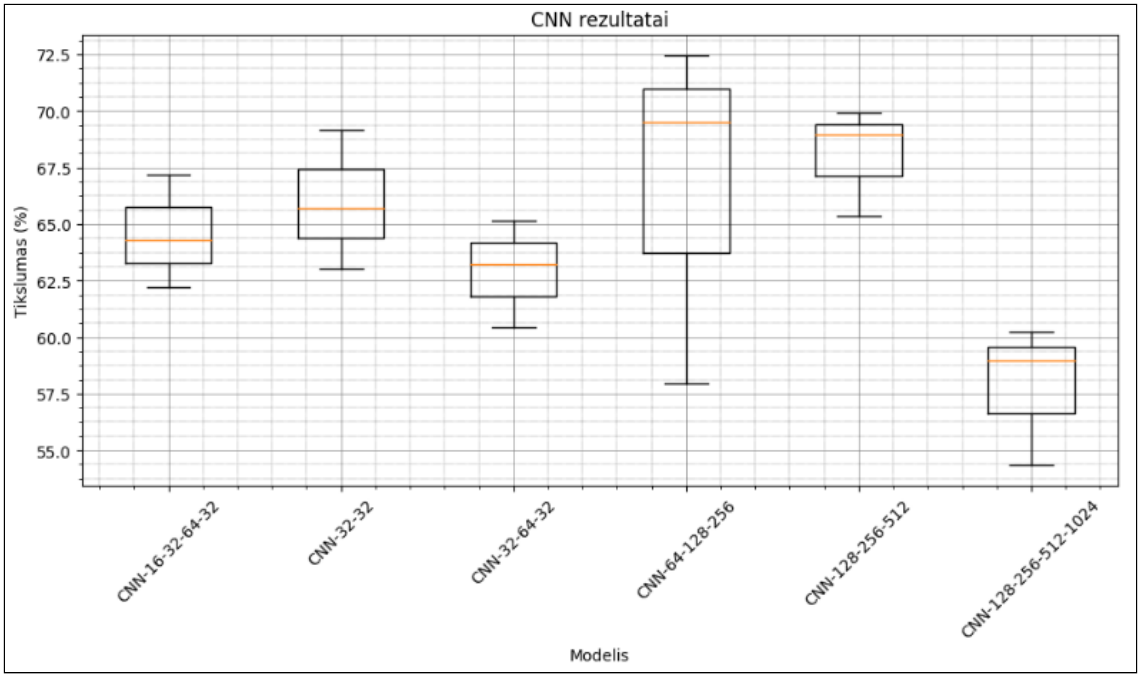


22 pav. CNN-128-256-512-1024 maišos matrica

Stengiantis toliau tobulinti prieš tai sukonstruotą modelį, buvo bandoma pridėti dar vieną konvoliucinį sluoksnį su 1024 filtrais. Deja, tai nepadėjo sulaukti tikslesnių rezultatų, o matoma, jog labiau įvyko persimokymas - mokymosi rezultatai beveik identiški su prieš tai nagrinėto modelio, tačiau testavimo metu šio modelio rezultatai prastesni.

6.2.7. Tarpinė CNN išvada

Mūsų sukonstruoti CNN modeliai leido pasiekti aukščiausią vidutinį mokymosi tikslumą 84.32% ir aukščiausią vidutinį testavimo tikslumą lygų 69.49%.



23 pav. CNN rezultatų stačiakampinė diagrama

23 paveikslėlyje mes matome CNN modelių testavimo tikslumus. CNN-64-128-256 pasiekė geriausius rezultatus, tačiau aiškiai matyti, jog taip pat turi ir daugiausiai svyravimų - atliekant 10 aplankų validavimą, tikslumas kito nuo 57 iki 72 %.

Be aukščiau aprašytų modelių, buvo bandoma tobulinti rezultatus keičiant konvoliucijos kernelio dydį, išmetimo santykį, tačiau didelių pokyčių nebuvo pastebėta - mūsų aprašytas CNN-64-128-256 modelis visais atvejais grąžino geriausius vidutinio atvejo testavimo tikslumo rezultatus - po 10 aplankų validacijos, tikslumas siekia 69.49%.

6.3. Rezultatai

Toliau pateikiama bendra nagrinėtų modelių rezultatų lentelė, kur tikslumas yra vidutinis modelio atpažinimo procentas teisingai klasifikuojant testinius duomenis.

Modelis	Tikslumas
DNN-256-256	57.14%
DNN-256-256-256	54.75%
DNN-256-512-256	63.11%
DNN-512-1024-512	44.65%
DNN-400-200	54.88%
DNN-128-256-512-256-128	44.28%
CNN-16-32-64-32	64.31%
CNN-32-32	65.71%

CNN-32-64-32	60.43%
CNN-64-128-256	**69.49%**
CNN-128-256-512	68.94%
CNN-128-256-512-1024	58.94%

4 lentelė. DNT modelių tikslumo rezultatai

Palyginus mūsų sukurtus 12 dirbtinių neuroninių tinklų modelių (6 giliuosius ir 6 konvoliucinius), galime spręsti, jog CNN geriau susitvarko su urbanistinių garsų klasifikavimo užduotimi. Lyginant abiejų tipų geriausius atvejus, testavimo metu mūsų sukurti CNN modeliai ~6 procentais tiksliau korektiškai atpažįsta nagrinėjamą garsą, lyginant su sukurtais DNN modeliais.

Tačiau mums nepavyko sukurti tokių modelių, kurie pralenktų rezultatus, kurie minimi [ZXCZ18] - jų pateiktas modelis pasiekia 83.7% tikslumą. Šiuo atveju mums pavyko sukurti modelius, kurie konkuruoja su baziniu atpažinimo tikslumu minėtu [Pic15] - mūsų 69.49% su CNN-64-128-256 modeliu, o [Pic15] - 72.7%.

Išvada

Bakalaurinio darbo metu buvo iškeltas tikslas palyginti skirtingų dirbtinių neuroninių tinklų - konvoliucinių bei giliųjų neuroninių tinklų atpažinimo efektyvumus urbanistinių garsų kontekste. Tam, kad pasiekti šį tikslą, buvo išanalizuota teorinė dalis, susijusi su dirbtiniu intelektu, mašinų mokymusi, neuroniniais tinklais - buvo išanalizuoti du neuroninių tinklų tipai - gilieji ir konvoliuciniai. Taip pat buvo išanalizuota tai, kaip išgaunamas garsas, kaip jis paruošiamas, kokios buvo naudojamos skirtingos charakteristikos tolimesniam neuroninių tinklų lyginimui, bei kokios buvo garso apdorojimo technikos naudojamos tinkamai paruošti duomenims.

Šiame darbe taip pat matome implementuotas dirbtinių neuroninių tinklų architektūras Google Colaboratory aplinkoje naudojant Python programavimo kalbą. Buvo naudota UrbanSound8K duomenų bazė, kurią sudaro 9.7 valandos įvairių pažymėtų garsų, kuriuos kas dieną girdime.

Iš viso buvo nagrinėta 12 tinklų: 6 gilieji ir 6 konvoliuciniai dirbtiniai neuroniniai tinklai. Taip pat buvo konstruojami ir kiti modeliai su įvairiais kintamųjų pakeitimais, tačiau jų rezultatai nebuvo geresni, negu prieš tai aprašyti modeliai.

Mes sugebėjome sukurti CNN modelį, kuris pasiekia vidutiniu atveju 69.49% tikslumą.

Tyrimas pavyko - lyginant DNN ir CNN tikslumo rezultatus, galime spręsti, jog šiuo atveju CNN naudoti urbanistinių garsų klasifikavimui yra geriau, negu DNN - lyginant geriausius sukonstruotus DNN ir CNN modelius, CNN 6% tiksliau korektiškai atpažįsta garsą.

Summary

In this Bachelor's thesis, the main goal was to compare the different architectures of a few artificial neural networks (deep neural networks and convolutional neural networks) in the context of urban environmental sound classification. To reach this goal, I have analyzed the theoretical side of artificial intelligence, machine learning, artificial neural networks. There were 2 artificial neural network types that were analyzed - deep as well as convolutional neural networks. Also, I've investigated as to how is sound processed and used in the context of sound classification - what features are commonly used, what pre-processing methods are implemented to prepare the data for use in the neural networks.

In this thesis we can see the results of our implemented neural network architectures. The environment to achieve this was Google Colaboratory as well as by using the Python programming language mainly by using the libraries mentioned in the work itself. The dataset for neural network evaluations was UrbanSound8K, which consists of 9.7 hours worth of individual sound slices, which we ourselves can hear each day.

There were 12 main networks that were analysed: 6 deep and 6 convolutional artificial neural networks. There were also other models of DNN and CNN that were constructed with different parameter changes (e.g. dropout rates, different activation functions), although they did not yield better results than the 12 networks that are discussed above.

We were able to create a CNN model, which, on average, would be able to correctly identify 69.49% of provided urban sounds.

We can conclude that the experiment was successful - after comparing the created DNN and CNN model results, we can conclude that the use of CNN (as opposed to DNN) in the context of urban sound classification would be more appropriate - when comparing the best cases of our created models, CNN is able to correctly identify 6% more of the provided sound data.

Literatūros sąrašas

[Ang18] D. Anggraeni "*The Implementation of Speech Recognition using Mel-Frequency Cepstrum Coefficients (MFCC) and Support Vector Machine (SVM) method based on Python to Control Robot Arm*", 2018 , pp. 2-4

[Bar15] D. Barchiesi "*Acoustic scene classification: Classifying environments from the sounds they produce.*" IEEE Signal Processing Magazine 32.3. 2015, pp. 16-34

[Bis15] M. J. Bishop "*History and philosophy of neural networks.*", 2015, p. 9

[CNK09] S. Chu, S. Narayanan and C. -. J. Kuo, "*Environmental Sound Recognition With Time–Frequency Audio Features.*", 2009, pp. 1142-1158

[CU07] Columbia University "*Chroma Feature Analysis and Synthesis*". URL: https://labrosa.ee.columbia.edu/matlab/chroma-ansyn/ , 12.04 KB, 2007

[Cyr08] V. Čyras "*Dirbtinis intelektas*", 2008, pp. 5-6

[MA16] Y. Munshi, A. Afshar. "*Five Stage Dynamic Time Warping Algorithm for Speaker Dependent Isolated Word Recognition in Speech.*", 2016, pp. 112-115

[PP08] B. Prasad, S. R. M. Prasanna "*Speech, Audio, Image and Biomedical Signal Processing using Neural Networks*", 2008

[Pic15] K. J. Piczak, "*ESC: Dataset for Environmental Sound Classification*", 2015, pp.

[RH11] S. Rosen, P. Howell, "*Signals and Systems for Speech and Hearing (2nd ed.)*", 2011

[SJB14] J. Salamon, C. Jacoby, J. P. Bello, "*A Dataset and Taxonomy for Urban Sound Research*", 2014

[SP14] D. Stowell, M. D. Plumbley, "*Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning*", 2014, pp. 488

[ZXCZ18] Z. Zhang, S. Xu, S. Cao, S. Zhang, "*Deep Convolutional Neural Network with Mixup for Environmental Sound Classification*", 2018, p. 7