TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Priidik Meelo Västrik

Muusika saate genereerimine tingimusliku vastandgeneratiivse närvivõrgu abil

Bakalaureusetöö (9 EAP)

Juhendaja: Anna Aljanaki, PhD

Tartu 2024

Muusika saate genereerimine tingimusliku vastandgeneratiivse närvivõrgu abil

Lühikokkuvõte:

Kvaliteetse muusika saate genereerimine on väga kasulik heliloojatele ja muusikaprodutsentidele. Generatiivsed tehisnärvivõrgud koguvad populaarsust ning muusikat genereerivaid mudeleid avalikustatakse üha enam. Käesoleva bakalaureusetöö eesmärk on genereerida muusikale saadet kasutades melspektrogrammide töötlust pilditöötlusmudeli Pix2Pix abil. Töö raames viiakse läbi eksperimente erinevate saadete genereerimiseks. Parimad tulemused tulid trummide genereerimisel. Tulemustest oli näha, et vastandgeneratiivsete närvivõrkudega töötades on väljundis sageli ebaloomulikud mustrid, mis kahjustavad tulemust. Selle ära hoidmiseks tuleb mudelit palju peenhäälestada.

Võtmesõnad:

Muusika, genereerimine, närvivõrgud, pilditöötlus

CERCS: P176 Tehisintellekt; P175 Informaatika, süsteemiteooria

Music Accompaniment Generation Using a Conditional Generative Adversarial Network

Abstract:

Generation of good quality music accompaniment is very useful for composers and music producers. Generative artificial neural networks are booming and there is recently an increasing amount of music generation models published. The aim of this Bachelor's thesis is to generate music accompaniment using spectrograms and an image translation model Pix2Pix. Experiments are conducted to generate different types of accompaniments. The best results are achieved when generating the drum stem. It can be seen from the results that generative adversarial networks' outputs contain unnatural artifacts that affect the results badly. Preventing this requires lots of finetuning.

Keywords:

Music, generation, neural networks, image translation

CERCS: P176 Artificial intelligence; P175 Informatics, systems theory

Sisukord

Sissejuhatus

Viimaste aastatega on tehisintellekti mudelite võimed arenenud märgatavalt ning seetõttu on neid kasutavaid rakendusi hakatud üha laialdasemalt kasutama. Peamiselt on levinud teksti- ja pildimudelid, mis suudavad väga loomulikult tekstivormis küsimustele vastata ja ülesandeid täita ning pilte genereerida ja muuta. Veidi vähem on levinud muusikatöötlus, kuid ka see kogub populaarsust ning muusika genereerimiseks on juba väga häid rakendusi nagu näiteks Suno[1], mis genereerib etteantud tekstile kvaliteetseid muusikapalu.

Käesolevas töös katsetatakse muusika osade genereerimist melspektrogrammide pilditöötluse abil. Eesmärk on muusika täieliku genereerimise asemel olemasolevat muusikat muuta läbi melspektrogrammide. See on kasulik näiteks muusikaprodutsentidele, kes saavad seeläbi proovida väikseid osi oma lugudest välja vahetada või leida inspiratsiooni uute lugude jaoks. Kui genereerida näiteks ainult loole vastavaid trummikäike, siis saab produtsendile tekitada suure valiku erinevatest käikudest, mis ülejäänud looga kokku sobiksid ning teha muusikatarkvarade kasutamise palju mugavamaks ja kiiremaks.

Kui treenida mudel ühe kindla žanri muusika põhjal ning sellele sisendina anda mõnest teisest žanrist puuduolevate osadega melspektrogramm, proovib mudel jätkata olemasolevaid mustreid treeningandmete žanri põhjal. See võimaldab olemasolevaid muusikapalu konverteerida teistele žanritele lähedasemaks ning heade tulemuste kohal saavad muusikud hakata selliseid mudeleid remikside tegemiseks kasutama, mis tooks juurde palju uut huvitavat muusikat.

Bakalaureusetöö raames analüüsitakse erinevaid muusika salvestusviise ning nende töötlemisvõimalusi. Seejärel kirjeldatakse töös kasutatud mudelit Pix2Pix ning muid tehnoloogilisi vahendeid. Kirjeldatakse antud töös kasutatud andmeid, nende töötlust ning muusika mudeli treenimist. Viimaks tehakse ülevaade heliklippidele trummide lisamise tulemustest.

[1] https://suno.com/

1. Mõisted ja terminid

Spektrogramm (ingl *spectrogram*) on akustilise või muu signaali muutuste graafiline esitus[2]. Seda eristab tavalisest kahemõõtmelisest graafikust kolmas telg, mis on välja toodud värvide abil. Muusika töötlemisel on spektrogramm kasulik, sest see võimaldab heli käsitleda kui pilti.

Melspektrogramm on spektrogramm, mille sageduse telg on viidud Mel skaalale [1].

Kõrgjõudlustöötlus (ingl *high-performance computing*) ehk supertöötlus on arvutivaldkonna haru, mis keskendub superarvutite kasutamisele ja rööptöötlusele[2]. Kõrgjõudlustöötluse klastrid koosnevad paljudest masinatest, mis sisaldavad võimsaid protsessoreid ja graafikakaarte. Sageli kasutatakse klastreid tehisintellekti mudelite kiiremini treenimiseks.

Neurovõrk (ingl *neural network*) ehk närvivõrk on tehisintellektis kasutatav masinõpe mudel, mis jäljendab bioloogilise neuronivõrgu omadusi[2]. Nende abil saab luua näiteks generatiivseid mudeleid, mis ennustavad muusikas järgmisi noote või klassifitseerivad muusikapala žanri järgi.

Heligruppide eraldamine (ingl *source separation*) on helisignaali eraldamine sarnaste helide gruppideks või voogudeks (ingl *stem*), näiteks saateks ja vokaaliks.

Muusika saade on muusikapala osa, mis toetab ja täiendab loo põhimeloodiat. Üldiselt kogu lugu peale vokaalide või muu meloodiat mängiva instrumendi.

[2] https://akit.cyber.ee/

2. Taustainfo

Selles peatükis antakse ülevaade erinevatest muusika struktuuri osadest ning nende töötlusest. Lisaks analüüsitakse teemaga seotud uurimustöid ning kirjeldatakse töös kasutatavat pilditöötluse mudelit.

2.1 Muusika osad

Muusikapalad koosnevad erinevatest komponentidest nagu tempo, meloodia, harmoonia ja tämber[3]. Need kõik koos kirjeldavad muusika olemust ehk tekstuuri. Tekstuur kirjeldab näiteks muusika tihedust ehk seda, kui palju erinevaid kihte lool on. Üks levinumaid traditsioonilisi muusika tekstuure on homofoonia[4]. Selles on põhiline meloodia, mida toetavad üks või mitu lisahäält. Muusikateadlane Allan Moore on 2012. aastal välja toonud, et traditsioonilised muusika tekstuuri vormid nagu homofoonia ja polüfoonia on suunatud klassikalise muusika kirjeldamisele ning pakkus välja funktsionaalsete kihtide süsteemi poppmuusika jaoks[5]. See koosneb selge rütmi kihist (ingl *explicit beat layer*), funktsionaalse bassi kihist (ingl *functional bass layer*), harmoonilise saade kihist (ingl *harmonic filler layer*) ja meloodilisest kihist.

Rütmi kihi moodustavad pillid, millel pole võimalik mängida kindlaid helikõrgusi nagu määratlemata helikõrgusega löökpillid. Kuna löökpille pole võimalik meloodiat mängima panna, tuleb need sobitada ülejäänud looga rütmi poolest. Trummikäikudega saab anda lugudele täiesti erinevaid meeleolusid ning seega on neil väga oluline roll.

Meloodia kihis on sageli juhtival kohal inimese hääl, kuid selle asemel saab olla ükskõik milline kontrollitava helikõrgusega pill (ingl *pitched instrument*). Moore'i sõnul on meloodia sageli inimestele kõige meeldejäävam osa loost ning vastavalt meloodia kihi instrumendile saab loo žanrit määrata.

[3] https://hellomusictheory.com/learn/texture/

[4] https://www.perennialmusicandarts.com/post/four-types-of-texture-in-music

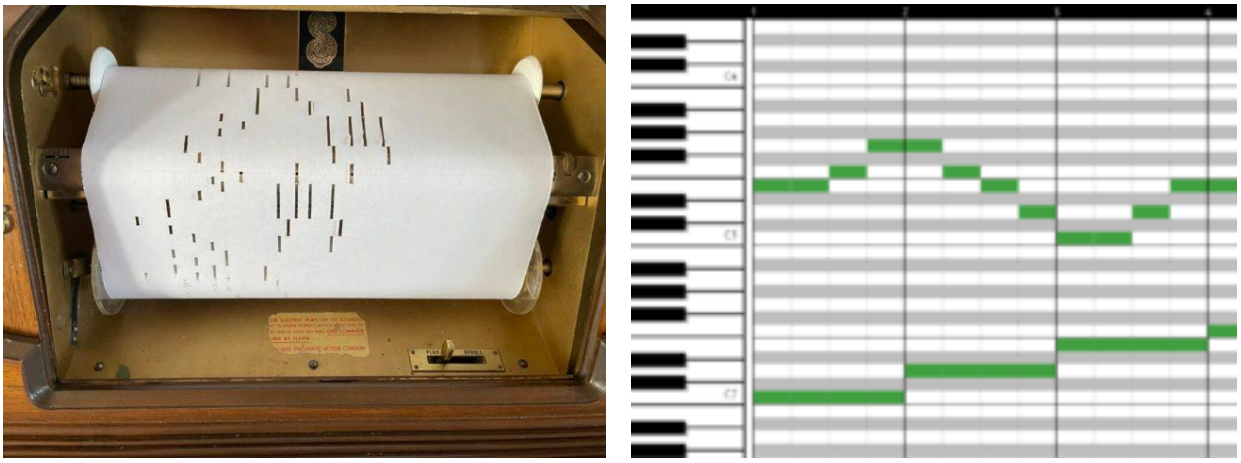
[5] https://viva.pressbooks.pub/openmusictheory/chapter/texture-in-pop-music/

Muusika töötlemiseks on vahel vaja need kihid tervikloost eraldada ehk jagada lugu voogudeks. Selle jaoks on levinud Pythonil põhinev tööriist Spleeter[6]. See jagab sisendloo vastavalt vajadusele kaheks kuni viieks vooks: vokaalid, trummi, bassi, klaveri ja ülejäänud helid. Kõige levinum on tavaline vokaali-saate eraldus, mis eraldab muusikapalast laulmise ja saatemuusika.

2.2 Muusika salvestusviisid

Muusika salvestusviise on kahte tüüpi: sümboolsed ja helisalvestised. Sümboolne muusika salvestus on justkui noodileht, mis sisaldab juhiseid loo mängimiseks. Helisalvestis on aga otsene salvestus, mis võimaldab levitada muusikat heli kujul.

Üks lihtsakoelisemaid sümboolseid muusika andmestruktuure on klaverirull (ingl *piano roll*). See on visuaalne esitus klaverist pealtvaates, kus iga klahvi mängimine on kujutatud kriipsuna, mille pikkus vastab heli pikkusele, selle klahvi reas. Masinõppes saab klaverirulli kujutada kui maatriksit, kus iga klahvi kohta on rida ning selle veerg on täidetud nendel hetkedel, kus klahvi on vajutatud ja hoitud. See formaat on küll parim klaverimuusika jaoks, kuid sellega saab kujutada ka teiste pillide muusikat. Siiski on tegu ainult nootide ja nende pikkuste suhtega, seega ei saaks selles formaadis aru, kas lugu on mängitud klaveri või kitarriga.



Joonis 1. Füüsiline[7] ja digitaalne[8] klaverirull

[6] https://research.deezer.com/projects/spleeter.html

[7] https://makezine.com/article/craft/music/the-quest-to-make-player-piano-rolls-using-lasers/

[8] https://en.wikipedia.org/wiki/Piano_roll

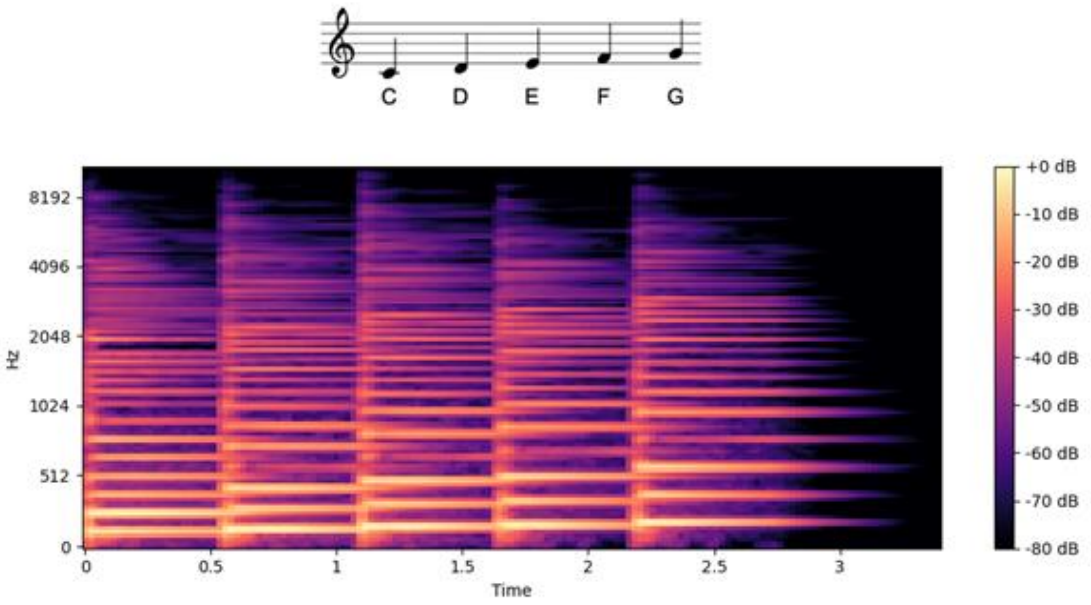
Klaverirulli formaat pärineb esimeste isemängivate klaverite – pianolade – sees olnud paberirullidest. Klaverirulle kasutati juba 20. sajandi alguses. Tegu on paberirulliga, millesse on tehtud augud samasuguse loogikaga nagu on noodid märgitud digitaalses klaverirulli formaadis. Seda kasutades mängitakse muusikat pianola abil nii, et inimene pumpab pedaalidega õhku läbi paberi ning vastavalt aukude asukohtadele liiguvad pianola haamrid nagu klaveril, mille tulemusena kõlavad noodid. Juba 1905. aastaks arendati seda edasi salvestamisklaveriks, mis suutis luua klaverirulle samal ajal kui inimene antud klaverit mängis[9]. Joonisel 1 on näha kõrvuti füüsilist ja digitaalset klaverirulli.

Üks levinumaid sümboolse muusika formaate on ka ABC-formaat[10], mis on justkui tavalise noodilehe digitaalne formaat. See sisaldab infot nootide kohta tähtsüsteemis, kus lisaks nootide tähtnimedele on võimalik märkida nootide pikkusi ning oktaveid. Lisaks on formaadis võimalik lisada infot taktimõõdu, tempo, helistiku ja metaandmete kohta. Seetõttu on see kasutusel näiteks Eesti Kirjandusmuuseumi rahvaviiside andmebaasis ning paljudes teistes andmebaasides, kus on vaja metaandmeid talletada. ABC-formaati kasutatakse sageli sümboolse muusika genereerimiseks, sest sellega saab kasutada tekstil põhinevaid mudeleid.

Lisaks sümboolsele muusikale on võimalik töödelda ka helisalvestusi. Helifaile saab Pythoni programmis sisse lugeda näiteks amplituudide aegreana ehk järjendina, kus on heli amplituudiväärtused vastavalt diskreetimissagedusele (ingl *sampling rate*) tüüpiliselt kas 22 050 või 44 100 väärtust sekundi kohta. Aegreana genereeritakse tekstist muusikat näiteks MusicLM-i [2] ning OpenAI Jukeboxi [3] mudelites. Kuna väärtuste sagedus on suur, kasutatakse nii Jukeboxis kui MusicLM-is efektiivseks töötlemiseks autokooderit (ingl *autoencoder*) või neurokodekit (ingl *neural codec*), mis pakib sisendjärjendi väiksemaks vektorite jadaks.

[9] https://www.allclassical.org/player-piano-rolls-listening-to-history/

[10] https://abcnotation.com/wiki/abc:standard



Joonis 2. Sama heliredel noodilehel[11] ning klaveriheli melspektrogrammil.

Alternatiiv aegridadele on spektrogramm. See on heliandmete graafiline kujutus – kahemõõtmeline graafik, mis sisaldab lisaks ka kolmandat dimensiooni värvides. Spektrogrammi teljed tähistavad aega ja signaali sagedust ning värv tähistab amplituudi. Seega vastavas punktis olev värv tähistab selle sagedusega signaali amplituudi vastaval ajahetkel. Selle infoga saab muusikapalast tehtud spektrogrammi ka tagasi samaks muusikapalaks konverteerida. Seetõttu on spektrogrammi pildina käsitledes võimalik kasutada pilditöötlusvahendeid, et genereerida heli. Käesolevas töös kasutatakse melspektrogramme, kus on sageduse telg viidud Mel skaalale [1], kuid struktuur on sama. Neid on inimestel parem jälgida, sest sellel skaalal on visuaalselt üksteisest sama kaugel olevad helid ka inimesele kuulates sama kaugel. Joonisel 2 on näha ühe heliredeli melspektrogrammi koos samale helile vastava noodilehega, mis illustreerib melspektrogrammide loogikat. Sellel joonisel tähendavad melspektrogrammis eredamad värvid suuremat amplituudi vastaval sagedusel. Kõrgematel helidel on kõrgem sagedus ning seega heliredeli korral on ka melspektrogrammil näha samamoodi heli kõrgenemist.

[11] https://www.flowkey.com/en/piano-guide/read-sheet-music

2.3 Seotud tööd

Muusika genereerimine arvutite abil ehk algoritmiline komponeerimine on alguse saanud juba eelmisel sajandil [4]. Siis töödeldi rohkem sümboolset muusikat, sest selle keerukus on väiksem. Ebcioglu lõi 1988. aastal reeglipõhise süsteemi CHORAL, mis harmoniseeris ning analüüsis J. S. Bachi stiilis koraale [5]. See võttis sisendiks tähestikul põhinevas kodeeringus koraali meloodia ning andis väljundiks selle meloodia harmoniseeringu graafiliselt noodilehena. Kuna tegu oli aga reeglipõhise harmoniseerimisega, siis mudeli väljundid olid piiratud selle autori teadmistega ning reeglitele erandeid lisades muutuvad sellised süsteemid liiga kompleksseteks [4].

Arvutite arvutuskiiruse tõusu ja närvivõrkude arenguga hakati genereerima sümboolse muusika asemel ka heli. DeepMindi 2016. aastal tehtud WaveNet [6] oli esimene mudel, mis genereeris muusikat helilainetena ning 2019. aastal avaldati WaveGAN [7], mis oli üks esimesi mudeleid, mis käsitles heli kui pilti. Selles töös uuriti nii helilainete kui ka spektrogrammina heli genereerimist. Heli pildina käsitlus ei pea olema vaid spektrogrammidega, vaid seda saab teha ka näiteks klaverirulliga nagu tegid 2023. aastal Min jt Polyffusioniga [8].

Spektrogrammi pildina käsitledes saab muusikat genereerida või uurida selle iseloomulikke omadusi kasutades pilditöötlusvahendeid. Üks genereerimise viise on kasutada pildi täitmise meetodit (ingl *image inpainting*) eemaldades spektrogrammilt mingi osa ning lastes mudelil pilt taastada treeningandmete põhjal. Selline lähenemine võimaldab muusika täieliku generatiivse loomise asemel olemasolevaid muusikateoseid muuta.

Piltide täitmisele on mitmeid lähenemisi. Üks lihtsamaid neist on paigapõhine meetod. Selle meetodiga on Nohiriko Kawai jt välja pakkunud viisi usutavalt peita piltidelt valitud objekte [9]. Huvipakkuva objekti ümbruse geomeetria põhjal laiendatakse tausta nii, et tulemus näiks loomulikuna. Kujundite ja mustrite põhjal on mudelitel võimalik ennustada, kuidas muster võiks edasi minna ning niimoodi objekte piltidel ära katta, mis tulemusena kaotab objekti pildilt jäljetult. Samasugust mustrite jätkamise loogikat on võimalik rakendada ka spektrogrammidel.

Üldkasutatavatele muusikaandmete formaatidele tehakse mudelite treenimiseks sageli väikseid muudatusi. Näiteks Hadjeresi jt loodud Bachi koraale genereeriva mudeli DeepBach [10] treenimiseks kasutati muusikaandmete salvestamiseks sümbolite jadasid. Nende esitusviisi tegi klaverirullist erinevaks lisasümboli „__“ kasutuselevõtt, mis tähistas noodi hoidmist. Seega neli lööki

pikk noot „D5“ on nende esitusviisis „D5, __, __, __“, kuid klaverirulli formaadis „D5, D5, D5, D5“. See teeb pika noodi kõrguse muutmise mudeli jaoks kordades lihtsamaks, sest muuta tuleb vaid ühte sümbolit. Autorite sõnul on see lisasümboli kasutuselevõtt ainuüksi põhjus, miks nad saavad valitud meetodiga nii häid tulemusi. Koraalid koosnesid enamasti neljast häälest, millest iga kohta oli järjend noodisümbolitest.

Guillen-Perezi projektis Vocals2Song[12] on kasutusel kodeeritud spektrogrammi formaat. Spektrogrammist on sageduse informatsioon salvestatud piksli punase värvi väärtustesse, amplituud rohelise värvi väärtustesse ja sinine on tühjaks jäetud. Pildid olid salvestatud BMP-formaadis, sest selle formaadi jaoks on TensorFlow-s olemas faili lugemise funktsioon. Selles formaadis kaob aga väärtuste täpsus, sest pikslivääртused on täisarvud, mis viis ebakvaliteetsema heliga tulemusteni. Siiski saadi isegi selle formaadiga autori sõnul paljulubavaid tulemusi. Projektis anti Pix2Pix mudelile ette mingi loo vokaalid ning loodeti saada täielik lugu koos saatega. Nende andmetega treenides jõuti selleni, et algne sisendheli oli tulemuses olemas, kuid saade oli obskuurne. Guillen-Perez toob välja, et paremate tulemuste jaoks võiks kasutada teist spektrogrammi salvestamise formaati nagu näiteks TIFF-formaat.

Vocals2Song töö on käesoleva tööga väga sarnane, aga selle leidsin alles projekti lõppfaasis ning avastasin, et olin jõudnud sarnase lahenduseni. Selleks ajaks kasutasin NPZ-formaati, mis lahendas juba ühe Guillen-Perezil tekkinud probleemi. Lisaks kasutan melspektrogramme, millel on graafiliselt paremini aru saada sellele vastavast helist, mis võiks aidata kaasa pilditöötlusmudeli treenimisele. Antud bakalaureusetöö raames on ka ligipääs suuremale arvutusvõimsusele, mis annab lootust parematele tulemustele.

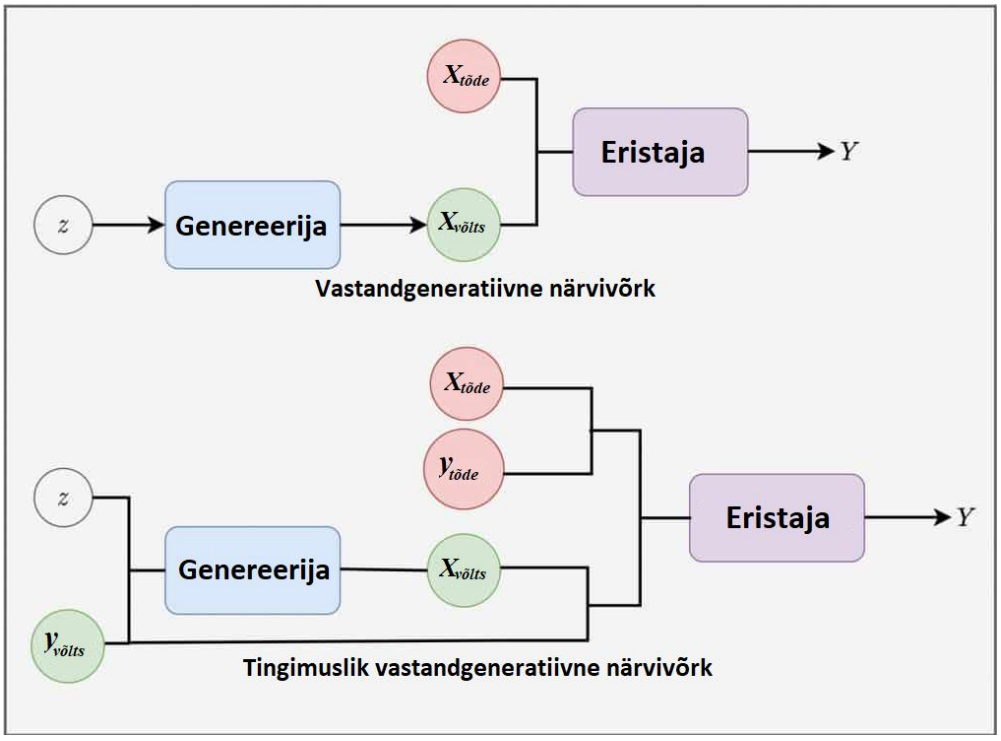
Rouard ja Hadjeres töötlesid välja meetodi CRASH (*Controllable Raw Audio Synthesis with High-resolution*), mis genereerib toore helisignaalina (amplituudi väärtuste aegrida) trummihelisid [11]. Helisignaaliga töötamine võimaldab võrreldes spektrogrammiga kõrgema kvaliteediga tulemusi saavutada, sest spektrogrammidest heliks tagasi konverteerimisel halveneb kvaliteet faasi informatsiooni puudumise tõttu. Lisaks on spektrogrammi sagedused ebatäpsed. CRASH kasutab hajuvusel (ingl *diffusion*) põhinevat mudelit, mida on lihtsam treenida kui näiteks

[12] https://github.com/AntonioAlgaida/Vocals2Song

vastandgeneratiivseid närvivõrke. Lisaks trummihelide tavalisele genereerimisele tegid nad ka auke täitva mudeli, mis võimaldab muusikaprodutsentidel olemasoleva trummiheli mingeid osi ära vahetada.

2.4 Pix2Pix

Käesolevas töös on melspektrogrammide genereerimiseks kasutusel pilditöötlusmudeli Pix2Pix arhitektuur [12]. See on pildist-pildiks mudel ehk võtab sisendiks pildi ning annab väljundiks sama suurusega pildi. Pilditöötluses on see peamiselt kasutusel näiteks visandite muutmisel maalideks või piltidelt defektide eemaldamisel (ingl *inpainting*). Tegu on vastandgeneratiivsel võrgul (ingl *generative adversarial network*), edaspidi GAN, põhineva arhitektuuriga. See tähendab, et mudel koosneb genereerijast (ingl *generaator*) ja eristajast (ingl *discriminator*). Treeningprotsessi käigus genereerija loob väljundi ning eristaja hindab kui hea antud väljund on. Pix2Pix on tingimuslik vastandgeneratiivne võrk ehk cGAN (ingl *Conditional GAN*). Seega selle väljundi genereerimist saab tingimustega mõjutada nagu näha joonisel 3. Tavalise GAN arhitektuuriga mudel genereerib väljundit lihtsalt treeningandmete põhjal suvaliselt. Käesolevas lõputöös on selleks tingimuseks melspektrogramm, mis suunab mudelit genereerima just sellele sarnaseid tulemusi.



Joonis 3. GAN ja cGAN arhitektuuride võrdlus

GAN-i genereerija on U-Neti struktuuriga ehk pilt sämplitakse väiksemaks ning siis uuesti suuremaks. See koosneb kodeerijatest ja dekodeerijatest – kõigepealt kodeerija töö käigus talletatakse pildi informatsiooni samal ajal, kui pildi suurust vähendatakse. Seeläbi saadakse kätte üha abstraktsemad pildi omadused. Dekodeerijate abil suurendatakse sämplit ning omaduste kogumi (ingl. *feature map*) abil pannakse kokku väljundpilt. Kodeerija ja dekodeerija kihtide vahel on ka otseühendused, mis aitavad üle kanda informatsiooni pildi algse suuruse kohta.

Eristaja eesmärk on teha vahet genereeritud piltidel ja baastõeks antud piltidel. See aitab GAN mudelitel genereerida väga realistlikke tulemusi, sest pidevalt hinnatakse nende autentsust ja selle põhjal ka õpitakse. Eristaja koosneb sarnastest kodeerimiskihtidest nagu generaator.

3. Kasutatud tehnoloogilised vahendid

Käesoleva peatükiga antakse ülevaade lõputöö raames kasutatud tehnoloogilistest vahenditest. Mudel on loodud programmeerimiskeeles Python Tensorflow raamistikuga ning treenitud kõrgjõudlustöötluse (ingl. *high performance computing*) , edaspidi HPC, keskkonnas kasutades Slurmi.

Python

Programmeerimiskeeleks on valitud Python[13], sest see on nii närvivõrkude loomisel kui muusikatöötlusel enim kasutatud keel.

Tensorflow

Masinõppes on kaks põhilist raamistikku – Tensorflow[14] ja PyTorch[15]. Mõlemal on omad eelised, kuid selles töös kasutatakse Tensorflow versiooni 23.04, sest pildimudel Pix2Pix, millel töö põhineb, on algselt tehtud selle raamistikuga. Tensorflow rakendusliides on tehtud Pythoni teegiks, mida on lihtne kasutada, et defineerida erinevaid masinõppemudelite struktuure, lugeda sisse andmeid ning mudeleid treenida.

Librosa

Pythoni teek librosa[16] on heli analüüsimise tööriist. See sisaldab hulgaliselt erinevaid meetodeid helist erinevate tunnuste eraldamiseks ning visualiseerimiseks. Käesolevas töös kasutatakse librosat muusikast melspektrogrammide tegemisel ning ka melspektrogrammide muusikaks konverteerimiseks.

Slurm

Slurm[17] (*Simple Linux Utility for Resource Management*) on tarkvara, mida kasutatakse laialdaselt teadusarvutuste keskustes tööde jooksutamiseks ja ressursihalduseks. HPC keskonnas programmi käivitamiseks on vaja kirjutada *batch script*, mis on sarnane dokkerfailile (ingl. *Dockerfile*) – see

[13] https://www.python.org/

[14] https://www.tensorflow.org/

[15] https://pytorch.org/

[16] https://librosa.org/doc/latest/index.html

[17] https://slurm.schedmd.com/overview.html

sisaldab infot ressursside kohta, mida programmi jaoks on vaja ja käske, kuidas keskkond valmis panna ning programm käivitada. Lisaks on failis Slurm-parameetritena märgitud näiteks maksimaalne tööaeg, väljundfailide asukohad ja muud parameetrid. See on vajalik sellepärast, et teadusarvuste keskustes käivad sama masina peal mitmed erinevate inimeste tööd samaaegselt ning kuna kasutajaid on palju, aga ressursid piiratud, on Slurmi põhiline eesmärk tagada võimalikult efektiivne tööde jaotus erinevate masinate vahel, et kasutajatel oleks võimalikult väiksed ooteajad.

4. Metoodika

Võtsin eesmärgiks genereerida muusika heli, mitte sümboolset formaati. Mudeli sisendiks on muusikaklipp ilma mingi osata (trummid, saade, vokaal) ning ülesandeks on puuduv osa lisada. Selleks kasutasin melspektrogrammide töötlust piltidel põhineva mudeliga Pix2Pix [12]. Mudel ise on piltpildiks, kuid kuna käesolevas töös on pildid melspektrogrammid, on mudeli sisendiks ja väljundiks heli. Viisin läbi kaks lugude täiendamise eksperimenti: trummide lisamine ülejäänud loo põhjal ning kogu loo saate genereerimine vokaalide järgi. Spektrogrammil väljenduvad trummilöögid pikkade vertikaalsete kriipsudena, seega on selle õppimine märgatavalt lihtsam kui kogu saate genereerimine.

Lõpptulemuseks on mudel, mis saab sisendiks loo melspektrogrammi ilma trummideta ning väljundina annab sama loo melspektrogrammi koos lisatud trummisaatega. Mudeli lihtsuse huvides on sisend ja väljund limiteeritud 256x256 piksli suuruseks ehk umbes 5 sekundi pikkuseks. Melspektrogrammide heliks ja heli melspektrogrammideks konverteerimine tehakse väljaspool mudelit vastavate programmidega. Lõputöö koodi ja tulemustega saab tutvuda projekti GitHubi lehel[18].

4.1 Andmebaasid

Töö käigus proovisin kasutada erinevaid andmestikke. Esimesteks katsetusteks kasutasin valimit music4all andmebaasist [13]. See andmebaas on loodud muusika andmeanalüütika (ingl. *music information retrieval*) valdkonna arenguks. See sisaldab iga loo kohta 30-sekundilist klippi, loo sõnu ja veel 16 erinevat atribuuti. Music4all koosneb rohkem kui 100 000-st loost ja 15 000-st anonüümsest kasutajast ja nende kuulamisajaloost. Käesoleva töö jaoks oli treenimiseks vaja vaid heliklippe.

Lisaks kasutasin Jamendot[19], muusikaplatvormi, mis on loodud muusika tasuta jagamiseks. 2004. aastal loodud lehekülg Jamendo.com on lehe loojate sõnul esimene muusikaplatvorm, mis jagab muusikat kõigilt Creative Commons litsentsi sõlminud artistidelt. Jamendol on ka avatud

[18] https://github.com/pvastrik/Pix2Pix-Music-Accompaniment-Generation

[19] https://www.jamendo.com/

rakendusliides, mille abil sain teha Pythoni programmi, mis laeb alla määratud koguse elektroonilise žanri muusikat.

Viimaks võtsin kasutusele MusDB18 andmestiku [14]. See on loodud muusikavoogude eraldamise treenimiseks ning koosneb juba eraldatud voogudest. MusDB18 koosneb 150-st täispikkuses loost ning nende eraldatud voogudest: trummid, bass, vokaalid ja muu. See on kokkupandud erinevatest andmebaasidest: 100 lugu on *Mixing Secrets For The Small Studio* andmestikust[20], 46 lugu MedleyDB andmebaasist [15], 2 lugu on Native Instrumentsi poolt ning 2 rokkbändi The Easton Ellises poolt. Kokku umbes 10 tundi muusikat.

4.2 Andmetöötlus

Music4allist ja Jamendost saadud lugudest pidi eraldama instrumentide vood vastavalt ülesandele. Selle käigus tuli olla tähelepanelik, et andmete seas ei oleks tühjasid vooge ja et saadud tulemused oleksid ka kvaliteetsed. Spleeteriga vooge eraldades olid tulemused vastavalt lugudele erineva kvaliteediga ning tundus, et see hakkas mudeli treenimist segama. Eriti elektroonilise muusika puhul oli märgata, et klassikaline voogude eraldamine polnud tõhus. Spleeteriga elektroonilise muusika voogusid eraldades jäi põhimõtteliselt kogu lugu *other* kategooriasse ehk kõik peale trummide, vokaalide, kitarri ja klaveri ning seega polnud eraldatud voogude melspektrogrammid treenimiseks piisavalt head.

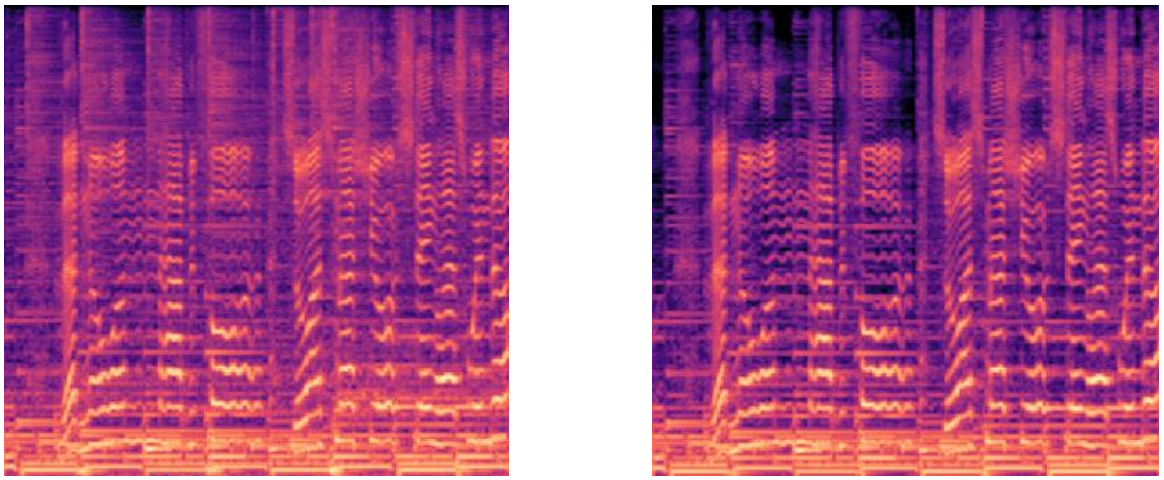
MusDB18 sisaldab faile *Native Instruments stems format*-is sufiksiga .stem.mp4. See on mitme helikanaliga failiformaat, kus on 5 stereo voogu. Erinevad vood on vastavalt järjekorrale: originaalne lugu, ainult trummid, ainult bass, kogu ülejäänud saade, vokaalid. STEM-failide töötlemiseks kasutasin Pythoni teeki stempeg[21]. Sellega kaasneb ka käsureatööriist, kuid kasutasin eraldamiseks lähtekoodi, mille abil lugesin STEM-failid sisse ning salvestasin eraldatud vood WAV-failidesse.

Kuna käesolevas töös on tegu helist-heliks mudeliga, kuid kasutan pilt-pildiks baasmudelit, siis on vaja treenimisele eelnevalt teisendada helifailid melspektrogrammideks, mida saab luua librosa meetodiga *librosa.feature.melspectrogram*. Melspektrogrammi pikslivääртused vastavad antud

[20] https://www.cambridge-mt.com/ms/mtk/

[21] https://pypi.org/project/stempeg/0.1.6/

ajal ja sagedusel oleva heli amplituudile. Seega on tegu põhimõtteliselt mustvalge pildiga, mille pikslivääртused võivad algselt olla vastavalt seadistusele väga erinevas vahemikus. Vahemiku normaliseerimiseks on librosa meetod *librosa.power_to_db*, mis viib amplituudi detsibell-skaalale ehk -80 ja 0 vahele. Andmete lihtsamaks grupeerimiseks ühendasin sisendi melspektrogrammi ja baastõe melspektrogrammi üheks pildiks ning andmeid sisse lugedes need eraldatakse programmaatiliselt. Joonisel 4 on näha näidet melspektrogrammide paarist, kus vasakpoolne on terviklik muusikaklipp, kuid parempoolsel on trummid eemaldatud.



Joonis 4. Trummi lisamise mudeli treeningandmete paar

Melspektrogrammi salvestasin algselt mustvalge pildina PNG-formaadis, kuid pärast esimesi katsetusi selgus, et see kaotab täpsust, sest pikslivääртused said olla vaid täisarvud vahemikus 0-255. See aga võtab melspektrogrammidega tegeledes täpsust hulganisti vähemaks. Seega võtsin kasutusele TIFF-formaadi, mis võimaldab salvestada ka ujukomaarve. Sellele formaadile polnud aga Tensorflow's sisselaadimise meetodit, seega tuli seda teha ise ning see osutus mitte eriti efektiivseks. Lõpuks kasutasin seetõttu hoopis NumPy poolt pakutavat NPZ-formaati[22], mis salvestab faili sisse lihtsalt NumPy järjendi vastavalt selle andmetüübile. NumPy järjendifaili on väga kerge sisse lugeda ning see kiirendas andmete sisselugemist märgatavalt.

[22] https://numpy.org/doc/stable/reference/generated/numpy.savez.html

Tensorflow lehel pakutud Pix2Pix mudeli lähtekoodis on mudelit kasutatud majafassaadide genereerimiseks lihtsa joonistuse põhjal[23]. Pildid on 256x256 pikslit suured ning kolmevärvilised. Seega mudel genereerib ja võtab sisendiks alati ainult kindla suurusega pilte. Seetõttu valisin ka spektrogrammidele sama suuruse, kuid tegu on ühedimensiooniliste piltidega.

Mudel salvestab väljundi melspektrogrammid NPZ-formaadis nagu sisendandmed. Melspektrogrammide heliks konverteerimine tuleb teha pärast treenimist eraldi Pythoni programmiga. Selle jaoks on librosas vajalikud meetodid olemas. Kõigepealt tuleb melspektrogramm viia detsibellskaalalt tagasi amplituudi skaalale meetodiga *librosa.db_to_power*. See meetod üldiselt ei tee midagi muud, kui suurendab helitugevust antud konstandi kordselt, seega saab selle konstandi abil tulemuse valjust mõjutada. Seejärel saab melspektrogrammist heli *librosa.feature.inverse.mel_to_audio* meetodiga, mille saab siis teegiga soundfile[24] helifailiks salvestada.

4.3 Treenimine

Kuna GAN-ide treenimine on väga ressursimahukas, sain mudeli treenimiseks ligipääsu Tartu Ülikooli teadusarvutuste keskuse klastrile Rocket[25] ning LUMI[26] klastrile. Kuna Rocketis tekkisid tarkvarade versioonidega konfliktid, viisin treenimised läbi LUMI keskkonnas. Kasutasin vaid ühte AMD MI250x graafikakaarti. Kokku treenisin mõlema eksperimendi jaoks 100 epohhi ning see võttis umbes 70 tundi.

Iga epohh koosnes 100 000 treeningsammust ning iga 50 000 sammu järel salvestas mudel ühe ja sama sisendiga tulemuse. Nende failide abil sai hiljem uurida, kuidas mudel arenes. Iga 5 epohhi järel salvestas ka mudel kontrollpunkti (ingl *checkpoint*), mille abil saab treenimist hiljem samast punktist alustada. Pärast 100 epohhi treenimist valis mudel testhulgast 50 sisendit ning genereeris ja salvestas sisendid, tulemused ja sisendiga kaasaskäiva testtulemuste kausta. See andis parema ülevaate mudeli hetkeseisust, sest treenimise ajal genereeriti tulemusi vaid ühe näitega.

[23] https://www.tensorflow.org/tutorials/generative/pix2pix

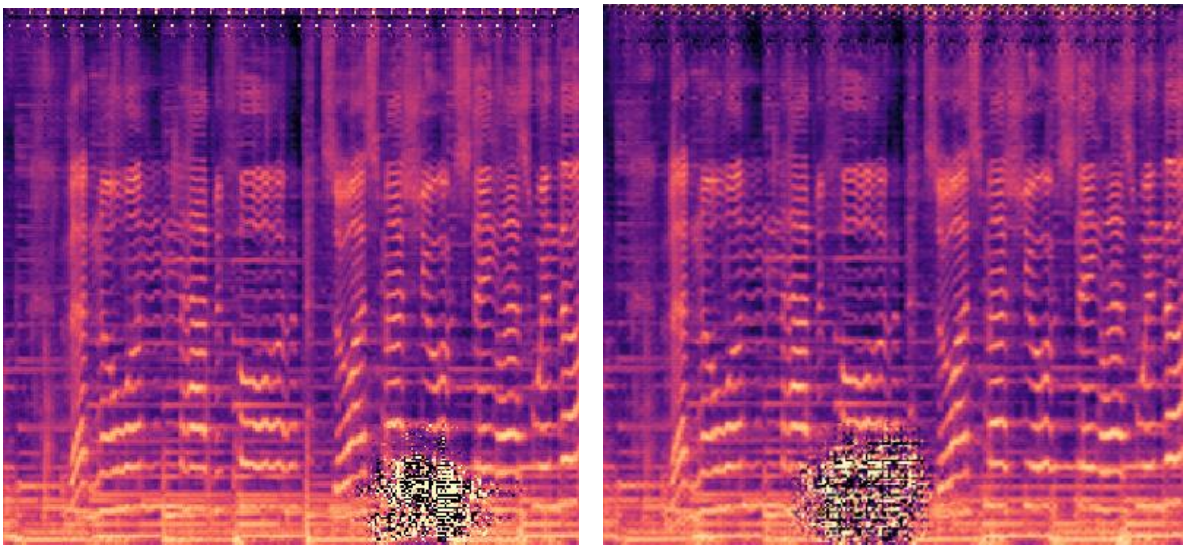
[24] https://pypi.org/project/soundfile/

[25] https://hpc.ut.ee/services/HPC-services/Rocket

[26] https://lumi-supercomputer.eu/

5. Tulemused

Mõlema katse korral oli märgata, et sisendheli oli väljundis kuulda juba peale üksikuid epohhe. Lisatav heli oli algselt vaid müra, kuid treenimise käigus eraldusid sellest äratuntavad helid. Arvestatavad tulemused tulid vaid trummide lisamisel, kogu saate genereerimise treenimistel tekkis tulemustesse liiga palju müra. Mõlemal treenimisel oli näha, et vastandgeneratiivse võrguga jäävad väljunditesse vahel visuaalsed tehismustrid (ingl *artifacts*). Selliste mustrite tekkimine genereerimise käigus on GAN närvivõrkudele iseloomulik. See on põhjustatud ülesdiskreetimise (ingl *upsampling*) kihtidest, mis teisendavad madala resolutsiooniga pildi suuremaks [16]. Treeningprotsesside käigus liikusid need mööda pilti ringi, kuid iga erineva ülesande raames tekkisid oma sarnased korduvad mustrid, mida on näha joonisel 5. Kõiki näiteid saab näha ja kuulata demo veebilehel[27].

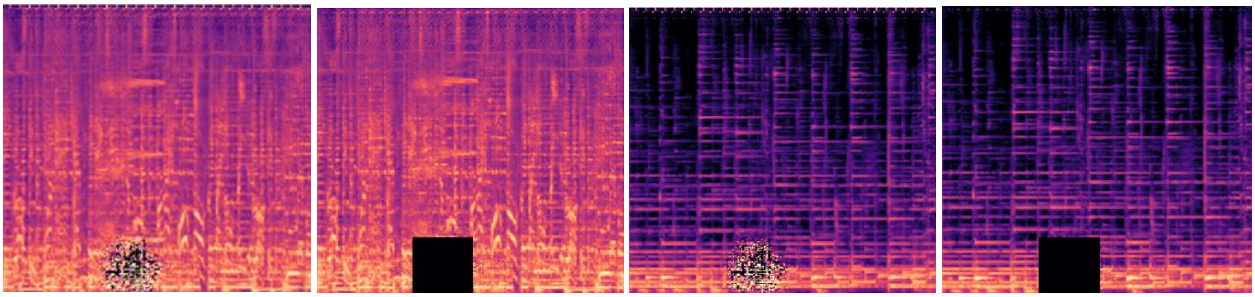


Joonis 5. Trummide genereerimise väljund sama sisendiga vastavalt 49. ja 72. epohhil

Trummide genereerimisel tekkis pärast 100 epohhi treenimist melspektrogrammide ülemisse äärde ühtlane muster, mis heliks konverteerides tähendas kõrget ühtlast pininat. Sarnase mustriga suuremad piirkonnad tekkisid ka piltide alumisse äärde. Neid on tugevamalt kuulda, kuid häirivad vähem, sest tegu on madalama sagedusega helidega. Samas pole need klipi jooksul pidevad, seega on nende olemasolu lihtsam märgata. Piltide järeltöötlusega saab sellist müra eemaldada näiteks

[27] https://pvastrik.github.io/Pix2Pix-Music-Accompaniment-Generation-Demo/

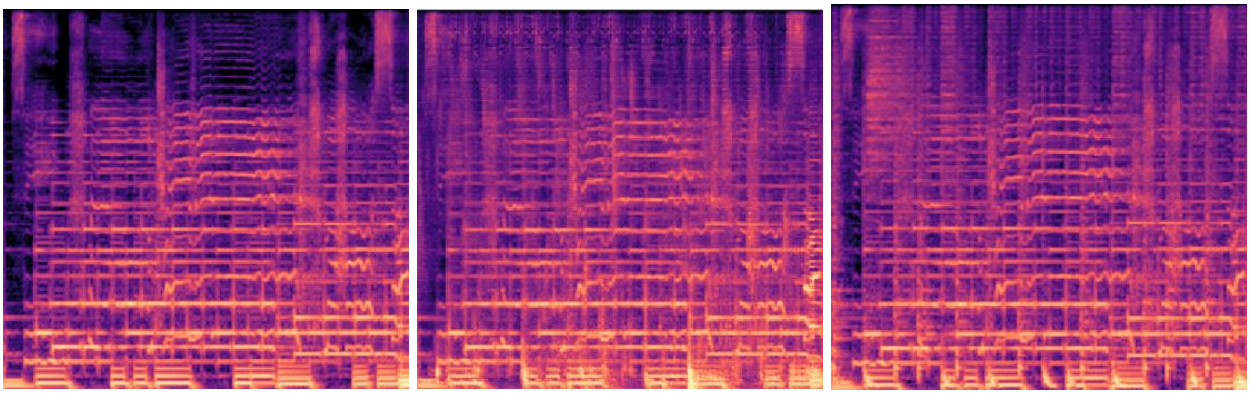
nende mustrite piirkondi välja lõigates. Selle tulemusena saab melspektrogrammi, millele vastaval klipil mürakohti pole, kuid augu koha peal on siiski kuulda, et madala sagedusega helid sel hetkel kaovad. Mudel genereerib samas treeningastmes olles samasuguste ebasobivate mustritega pilte ning seega saab kogu testhulgal lõigata välja sama piirkonna. Joonisel 6 on näha kahte näidet trummi genereerimise väljunditest pärast 100 epohhi treenimist ning kuidas need muutuvad, kui mürapiirkonnad välja lõigata. Nendele melspektrogrammidele vastavaid heliklippe saab samuti kuulata demolehel.



Joonis 6. Kaks näidet genereeritud müra melspektrogrammilt välja lõikamisest

Siiski, olenemata tekkinud ebaloomulikest mustritest, oli trummide genereerimisel näha häid tulemusi. Eriti rahulikumate või vaiksemate lugude puhul on kuulda lisatud trummihelisid, mis lähevad ka ülejäänud loo rütmiga kokku isegi siis, kui mõni piirkond oli väga mürarohke.

Parimad tulemused olid pärast 95. treeningepohhi – siis polnud tulemustes ühtegi läbivat mürapiirkonda. Võrreldes algsete lugudega, on lisatud trummikäigud tagasihoidlikumad ning seega jäävad tihedama kõlaga lugudel tagaplaanile. Levinumad on pehmemad kõristite või taldrikute helid, kuid mõnes näites on kuulda ka basstrummi ja plaksu heli.



Joonis 7. Vasakult paremale sisendi, väljundi ja baastõe kolmik trummi genereerimisel

Joonisel 7 on näha järjest trummi lisamise ülesande testhulga sisendit, väljundit ja oodatavat väljundit ühe treeninghulga klipiga. Kui melspektrogrammid heliks teha, on kuulda, et lisatud trummikäigud on erinevad. Seda on näha ka pildi pealt pikkade vertikaalsete joonte järgi. Oodataval väljundil on jooned tihedalt, mis näitab, et trummilööke oli rohkem ning see tuleb ka kuulates ilmsiks. Mudeli väljundi trummikäik sobib samuti sisendiga kokku, kuid kokkuvõttes on lool hoopis teine meeleolu. See illustreerib väga hästi trummikäigu olulisust loo juures ning võimalust selle mudeliga muusikat töödelda.

5.1 Tulemuste hindamine LPIPS ja MultiSSIM abil

Objektiivseks tulemuste hindamiseks võrdlesin genereeritud klippide melspektrogramme sisendi originaalklipi melspektrogrammiga kasutades piltide sarnasuse näitajaid. Selleks kasutasin Storia AI poolt tehtud Pythoni teeki image-eval[28], milles on erinevad piltide paarikaupa sarnasuse hindamise meetodid. LPIPS (ingl *Learned Perceptual Image Patch Similarity*) skoor on väärtus 0 ja 1 vahel, kus 0 tähendab, et kaks pilti on identsed [17]. MultiSSIM (ingl *Multiscale Structural Similarity*) [18] skoor on väärtus -1 ja 1 vahel, kus 1 tähendab perfektset sarnasust, 0 tähendab, et pole sarnasust ning -1 tähendab, et on perfektne anti-korrelatsioon. Tabelis 1 on näha mõlema treenimise sisendite ja väljundite keskmist sarnasust baastõega 50 testhulga näite puhul.

Tabel 1. Mudeli sisend ja väljund melspektrogrammide sarnasus kõiki voogusid sisalduva looga

	LPIPS	MultiSSIM
Saate genereerimise sisend	0.21875	0.705607
Saate genereerimise väljund	0.209149	0.665583
Trummi genereerimise sisend	0.0647562	0.938401
Trummi genereerimise väljund	0.100896	0.905285

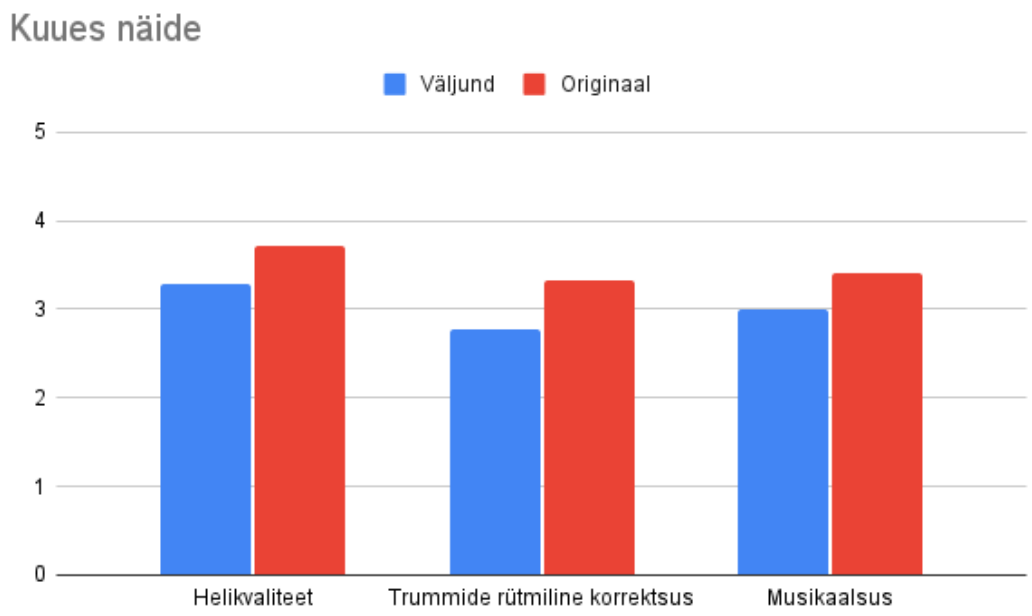
[28] https://github.com/Storia-AI/image-eval/

Kuna algse loo melspektrogrammis sisaldub ka sisendi melspektrogramm, siis oli loogiline, et sisendi ja baastõe sarnasus on päris suur, eriti trummide puhul. Siiski, sarnasuse püsimine ka väljundiga näitab, et mudel saab hästi hakkama vähemalt sisendis oleva info talletamisega.

Kuigi selline sarnasuse mõõtmine, annab aimduse, kui hästi mudel võis töötada, kuid tegelikkuses ei ole mudeli eesmärk genereerida baastõele sarnast, vaid lihtsalt kvaliteetset muusikat. Teatud piirini peab sarnasus jääma, kuid parimad tulemused on sellised, kus on lisatud midagi uut ja huvitavat. Seega on mõistlik lisaks hinnata tulemusi subjektiivselt kuulamise järgi.

5.2 Tulemuste subjektiivne hindamine

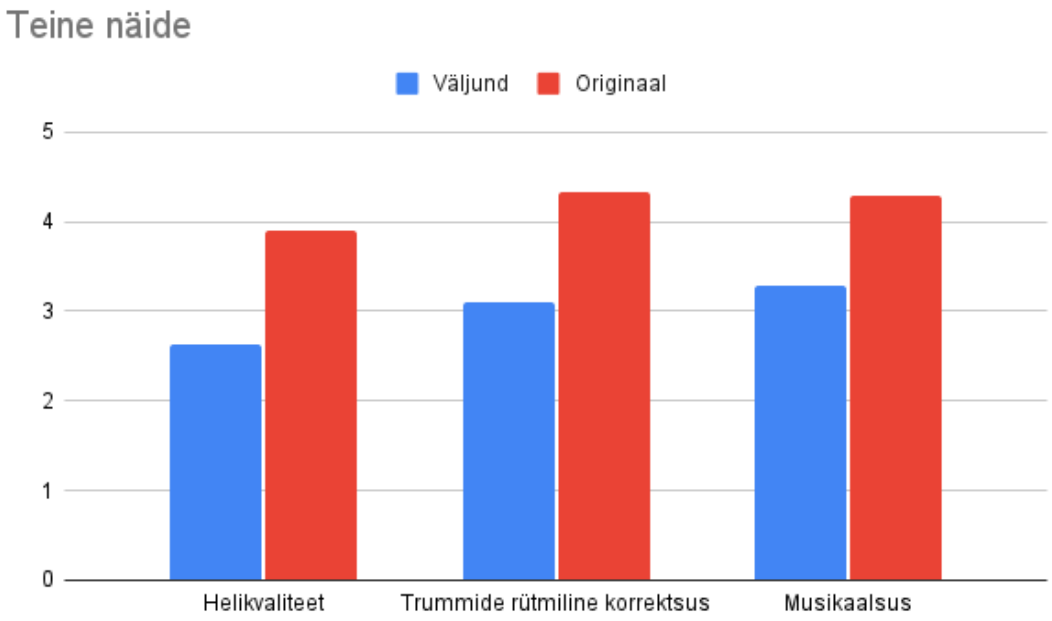
Koostasin demolehel olevate tulemuste analüüsimiseks küsitluse Google Forms keskkonnas ning palusin inimestel hinnata viie palli skaalal tulemuste kolme kriteeriumit: helikvaliteet, trummide rütmiline korrektsus ja musikaalsus. Küsitluse ajaks nimetasin mudeli väljundi esimeseks väljundiks ja baastõe teiseks väljundiks, seega vastajad ei teadnud, et kõik pole selle mudeli genereeritud tulemused. See tagas eelarvamusteta hindamise. Kokku oli küsitlusele vastajaid 22.



Joonis 8. Demolehe kuuenda näite väljundite hinnangud

Kõige sarnasemad hinnangud olid demolehe kuuendal näitel nagu näha Joonisel 8. Seda oli ka oodata, sest tegu on näitega, kus originaalis trumme pole ning mudeli väljundis on vaid väga

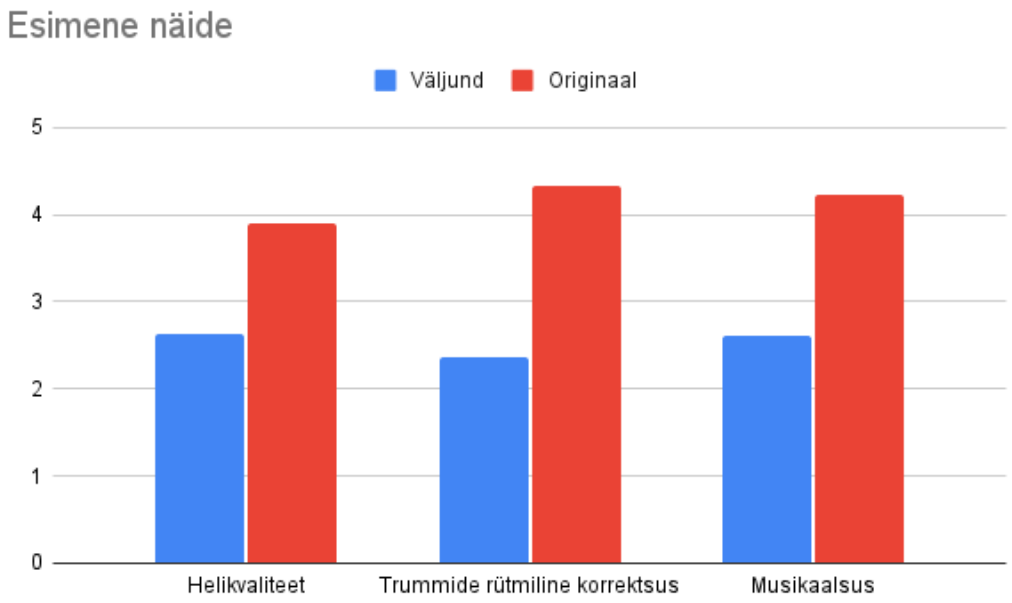
tagasihoidlikud trummilöögid taustal. See näitab, et mudel eristab lugusid nii, et rahulikumatele lugudele ei lisata suvaliselt agressiivseid trummikäike. Rahulikuma trummi lisamine tõi kaasa ka vähem müra ning seega on ka helikvaliteedi hinnang 3,3/5 selle näite väljundil parim.



Joonis 9. Demolehe teise näite väljundite hinnangud

Joonisel 9 on näha, et mudeli väljunditest kõrgeima hinnangu trummide rütmilisele korrektsusele ja musikaalsusele sai demolehe teine näide vastavalt keskmise hinnanguga 3,1 ja 3,3. See oli samuti aeglasem lugu, kuid sinna lisas mudel paremini kuuldava ja selgema rahuliku trummikäigu. Samuti oli sel korral ka originaalil trummisaade ning seega hinnati seda paremini.

Üllatavalt hinnati mudeli väljunditest kõige halvemini esimest näidet nagu on kujutatud joonisel 10. Trummide rütmiline korrektsus sai hindeks kõigest 2,37, kuigi pealtnäha seal otseselt valedel hetkedel trumm ei kõla. See on lihtsalt palju aeglasem ja tagasihoidlikum kui sama loo baastõde, mida hindajad pidasid teiseks väljundiks. See näide illustreerib trummivoo olulisust loo meeleolu tekitamises. Baastõel on palju kiirem ja täidetum trummivoog ning seega mõjub lugu palju rõõmsamalt, kui mudeli väljund. Siiski toodi kommentaarides välja, et selles näite mudeli väljundis on kuulda selliseid realistlikke trummihelisid, mida mujal ei ole.



Joonis 10. Demolehe esimese näite väljundite hinnangud

Ülejäänud kolm näidet olid kõik samast loost ning sarnaste väljunditega. Mudeli väljund suutis lisada loole kiirema, kuid seega ka mürarohkema trummivoo, milles oli vahel kuulda puhtaid trummikäike, kuid üldine ilme oli ebakvaliteetne. See põhjendab ka nende väljundite hinnangute keskmiseks jäämist.

Kokkuvõte

Muusika osade genereerimine on ressursimahukas ja keeruline ülesanne, millele on palju erinevaid lähenemisvõimalusi. Muusikat on pikalt genereeritud sümboolselt, kuid viimase aastakümne jooksul tänu tehisintellekti valdkonna kiirele arengule on hakatud muusikat genereerima helilainete või spektrogrammidena. Pilditöötlusmudeleid peenhäälestades on võimalik saavutada muusika genereerimisel sama häid tulemusi kui on suudetud piltidega.

Käesolevas lõputöös anti ülevaade muusika tekstuurist, erinevatest salvestusviisidest ning nende töötlemise võimalustest. Samuti analüüsiti varasemaid uurimusi, mis on seotud muusika või piltide generatiivse töötlusega. Anti ülevaade kasutatavast mudelist Pix2Pix ja vastandgeneratiivsetest närvivõrkudest.

Antud bakalaureusetöö eesmärk genereerida olemasolevatele lugudele uus saade kasutades melspektrogramme oli edukas vaid trummide genereerimisel. Tulemustes oli kuulda erineva iseloomuga trummivoogusid ning mõne näite puhul, kuhu trummid ei sobiks, mudel trumme põhimõtteliselt ei lisanudki. Melspektrogrammide hindamine nende visuaalse sarnasuse põhjal eriti informatiivne ei olnud, kuid inimeste subjektiivsete hinnangute põhjal oli näha, et arenguruumi veel on, kuid tulemused polnud ka halvad.

Viidatud kirjandus

[1] P. Pedersen, „The Mel Scale,“ *Journal of Music Theory,* pp. 295-308, 1965.

[2] A. Agostinelli, T. I. Denk, Z. Borsos, J. Engel, M. Verzetti, A. Caillon, Q. Huang, A. Jansen, A. Roberts, M. Tagliasacchi, M. Sharifi, N. Zeghidour ja C. Frank, MusicLM: Generating Music From Text, arXiv, 2023.

[3] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford ja I. Sutskever, Jukebox: A Generative Model for Music, arXiv, 2020.

[4] G. Papadopoulos ja G. Wiggi, „AI Methods for Algorithmic Composition: A Survey, a Critical view and Future Prospects,“ %1 *AISB Symposium on Musical Creativity*, 1999.

[5] K. Ebcioğlu, „An Expert System for Harmonizing Four-Part Chorales,“ *Computer Music Journal,* kd. 12, pp. 43-51, 1988.

[6] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior ja K. Kavukcuoglu, WaveNet: A Generative Model for Raw Audio, arXiv, 2016.

[7] C. Donahue, J. McAuley ja M. Puckette, „Adversarial Audio Synthesis,“ %1 *ICLR 2019*, 2019.

[8] L. Min, J. Jiang, G. Xia ja J. Zhao, „Polyffusion: A Diffusion Model for Polyphonic Score Generation with Internal and External Controls,“ %1 *24th Conference of the International Society for Music Information Retrieval (ISMIR 2023)*, Milan, Italy, 2023.

[9] N. Kawai, T. Sato ja N. Yokoya, „Diminished Reality Based on Image Inpainting Considering Background Geometry,“ *IEEE Transactions on Visualization and Computer Graphics,* kd. 22, nr 3, pp. 1236-1247, 2016.

[10] G. Hadjeres, F. Pachet ja F. Nielsen, „DeepBach: a Steerable Model for Bach Chorales Generation,“ %1 *The 34th International Conference on Machine Learning*, 2017.

[11] S. Rouard ja G. Hadjeres, „CRASH: Raw Audio Score-based Generative Modeling for Controllable,“ %1 *22nd ISMIR Conference*, 2021.

[12] P. Isola, J.-Y. Zhu, T. Zhou ja A. A. Efros, „Image-to-Image Translation with Conditional Adversarial Networks,“ %1 *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[13] I. A. P. Santana, F. Pinhelli, J. Donini, L. Catharin, R. B. Mangolin, Y. M. e. G. d. Costa, V. D. Feltrim ja M. A. Domingues, „Music4All: A New Music Database and Its Applications,“ %1 *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2020.

[14] F.-R. Stöter, A. Liutkus ja N. Ito, „The 2018 Signal Separation Evaluation Campaign,“ %1 *International Conference on Latent Variable Analysis and Signal Separation*, 2018.

[15] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam ja J. Bello, „MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research,“ %1 *15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, Taipei, Taiwan, 2014.

[16] X. Zhang, S. Karaman ja S.-F. Chang, „Detecting and Simulating Artifacts in GAN Fake Images,“ %1 *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2019.

[17] R. Zhang, P. Isola, A. A. Efros, E. Shechtman ja O. Wang, „The Unreasonable Effectiveness of Deep Features as a Perceptual Metric,“ %1 *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[18] Z. Wang, E. Simoncelli ja A. Bovik, „Multiscale structural similarity for image quality assessment,“ %1 *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Pacific Grove, CA, USA, 2003.

[19] Y. M. G. Costa, L. S. Oliveira, A. L. Koericb ja F. Gouyon, „Music genre recognition using spectrograms,“ %1 *2011 18th International Conference on Systems, Signals and Image Processing*, Sarajevo, Bosnia and Herzegovina, 2011.

Lisad

I. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Priidik Meelo Västrik,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose "Muusika saate genereerimine tingimusliku vastandgeneratiivse närvivõrgu abil", mille juhendaja on Anna Aljanaki reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 4.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Priidik Meelo Västrik

15.05.2024