

Nuottikirjoituksen tunnistus graafien ja konvoluutioverkkojen avulla

TURUN YLIOPISTO
Tietotekniikan laitos
LuK-tutkielma
Tietojenkäsittelytiede
Maaliskuu 2024
Sami Salo

TURUN YLIOPISTO
Tietotekniikan laitos

SAMI SALO: Nuottikirjoituksen tunnistus graafien ja konvoluutioverkkojen avulla

LuK-tutkielma, 20 s.
Tietojenkäsittelytiede
Maaliskuu 2024

Nuottikirjoituksen tunnistuksella tarkoitetaan sen piirteiden erottelamista ja muuntamista koneelle luettavaan muotoon. Piirteiden erottelamista voi lähestyä ongelmana monesta eri näkökulmasta, mutta yleisesti graafien ja konvoluutioverkkojen on todettu olevan hyödyllisiä työkaluja. Nuottikirjoituksen koneellista tunnistamista varten tulee kuitenkin huomioida syötteen laatu, ja tunnistamisen jälkeen vaaditut toimenpiteet tiedon tallentamiseksi koneelle ymmärrettävään muotoon. Nuotinluvun tunnistamista, esi- ja loppukäsittelyä yhdistää OMR:n, eli Optical Music Recognitionin, tutkimusala.

Ennen nuottikirjoituksen varsinaista tunnistamista voidaan syötekuvaa esikäsitellä tarpeellisten menetelmien avulla, joista yleisimpiä ovat binärisaatio, vinoumankorjaus, kohinanvaimennus ja tarkennus. Esikäsitelytoimenpiteiden tarkoituksena on kohentaa syötteen laatua, jotta syötettä käsittelevät algoritmit voisivat antaa mahdollisimman tarkkoja ennusteita. Vasta esikäsitelytoimenpiteiden jälkeen voidaan aloittaa merkintöjen tunnistus.

Nuottikirjoituksen merkintöjen tunnistus voidaan jakaa kahteen vaiheeseen: nuottiviivaston tunnistus ja symbolien tunnistus. Nuottiviivaston tunnistaminen toimii pohjana yksittäisten merkintöjen tunnistamiseen, koska monelle merkinnälle on tärkeää erottaa suhteellinen sijainti nuottiviivastoa kohden. Graafeja voidaan hyödyntää tehokkaasti tässä vaiheessa. Yksittäisten merkintöjen tunnistaminen puolestaan tapahtuu konvoluutioverkoilla kuten muissakin kuvantunnistusalgoritmeissa.

Merkintöjen tunnistamisen jälkeen tulee kuitenkin vielä huomioida nuottikirjoituksen semanttiset piirteet ennen tiedon lopullista tallentamista. Sovellettuja kielioppimalleja käyttämällä tunnistettujen symbolien piirteet ja laajemmat kokonaisuudet voidaan yhdistää merkityksellisesti, jolloin tiedostoon voidaan sisällyttää myös monet korkeamman tason nuottikirjoituksen ominaisuudet, joita syötteen tekijä on halunnut soittajan huomioivan. Valittu tiedon tallennusmuoto voi kuitenkin vaikuttaa lopulliseen tulokseen, sillä kaikki tarkoitukseen soveltuvat vaihtoehdot eivät kuitenkaan ole yhtä laadukkaita.

Asiasanat: OMR, nuottikirjoitus, neuroverkot, konvoluutioverkot, graafit

Sisällys

1 Johdanto	1
2 Neuroverkot ja graafit	4
2.1 Neuroverkot	4
2.1.1 Perseptronit	5
2.1.2 Konvoluutioverkot	6
2.2 Graafit	8
3 Nuotinluku kuvantunnistuksen avulla	9
3.1 Kuvan esikäsittely	10
3.2 Sisällön tunnistus	10
3.2.1 Nuottiviivaston erottaminen graafien avulla	12
3.2.2 Symbolien tunnistaminen konvoluutioverkkojen avulla	13
3.3 Tiedon rekonstruktio	15
3.4 Tiedon tallennus	17
4 Yhteenveto	19
Lähdeluettelo	21

1 Johdanto

Koneellistettuja menetelmiä nuotinlukuun tarvitaan tulkitsemaan jo vuosisatoja säilöttyjä kokoelmia käsinkirjoitettua musiikkia. Nuottien kääntäminen koneelle ymmärrettävään muotoon on jo itsessään hyvin työlästä, vanhemman notaation tulkinnasta puhumattakaan, ja kulttuurillisesti merkittävien säilöttyjen nuottien määrä on liian suuri käsin käännettäväksi. Nuottikirjoituksen koneellista tunnistamista (engl. Optical Music Recognition, OMR) on tutkittu jo vuosikymmeniä, mutta järjestelmällisen tutkimuksen puutteen takia on edistys ollut hajanaista ja johtanut usein saman tutkimuksen toistamiseen. Käsinkirjoitetun nuottikirjoituksen koneellinen tunnistaminen (engl. Handwritten music recognition, HMR) on myös eritelty omaksi tutkimuskohteeksi OMR:n alla, sillä käsinkirjoitetut nuotit eroavat selkeästi laadultaan koneellisesti tuotetuista. [1] [2]

Nuottikirjoitusta käytetään tapana merkitä musiikkia soittajalle luettavaan ja esitettävään muotoon. Länsimaisen yleisnotaation (engl. Common Western Music Notation, CWMN) ominaisia piirteitä on viisiviivainen nuottiviivasto, jonka avulla voidaan kuvata nuottien sävelkorkeutta annetulla asteikolla, tahtiviivat osoittamassa tahtilajin mukaista etenemistä, sekä nuottien ja niiden välisten taukojen kestojen kuvaaminen erilaisilla symboleilla. Nämä piirteet eivät kuitenkaan riitä kuvaamaan aina kaikkia haluttuja piirteitä, joten apuna käytetään usein myös semanttisia piirteitä, kuten äänenvoimakkuutta, intonaatiota tai rytmitystä kuvaavia symboleja joko yksittäisten nuottien tai laajempien kokonaisuuksien kuvaamisessa. Monien

muiden piirteiden kuvaaminen onkin riippuvaista notaatiosta, minkä takia kirjoitusmenetelmiä tulee käsitellä koneellisen nuotinluvun yhteydessä usein erikseen. Tämän tutkielman näkökulma rajoittuukin siksi vain länsimaiseen yleisnotaatioon. Muita notaatioita olisi esimerkiksi usein kielisoittimille kuten kitaralle tai bassolle suunnattu tabulatuurinotaatio (engl. tablature notation), jossa viivasto kuvastaa soittimen kieliä ja sille asetettavat numerot otelaudan tai -kaulan nauhakohtia, ja yleisnotaatiota edeltänyt mensuraalinotaatio (engl. mensural notation). Mensuraalinotaation suurimpia vajavaisuuksia yleisnotaatioon verrattuna on tulkinnanvaraisuus nuottien kestossa etenkin kolmijakoisessa tahtilajissa.

Nuottikirjoituksen luettavuuteen vaikuttaa moni asia merkintöjen selvydestä niiden aseteluun. Nuottien luettavuus on olennaista huomioitavaa niin soittajan kuin myös koneen ymmärryksen kannalta, ja luettavuus korostuu etenkin käsinkirjoitetussa nuottikirjoituksessa. Merkittävä luettavuuteen vaikuttava tekijä onkin joustavuus merkintätapojen suhteen, yleisnotaatio noudattaa ennalta sovittuja merkintätapoja joista kirjoittaja voi joustaa nuottien luettavuuden kustannuksella. Tahti- tai apuviivojen vinous, symbolien epäselvyys tai vaihtelu nuottien kestopuhteellisessa sijoittelussa ovat kaikki esimerkkejä luettavuuteen negatiivisesti vaikuttavista tekijöistä, joita tulee ottaa huomioon nuotteja tunnistaessa. [3]

Vaikka osa OMR:n algoritmeista kykeneekin tulkitsemaan käsinkirjoitettuja nuotteja ei tätä kuitenkaan voi kaikilta algoritmeilta realistisesti odottaa, minkä takia usein täsmennetäänkin onko kehitetty algoritmi suunniteltu OMR:n vai HMR:n tutkimukseen. Luettavuuden aiheuttamat ongelmat eivät rajoitu kuitenkaan vain käsinkirjoitettuun notaatioon, vaan myös koneellisesti tuotettu kirjoitus voi olla hankalasti luettavaa vaikka käytetyt symbolit olisivatkin selvästi kirjoitettuja. Lukuisten etumerkintöjen käyttö nuoteissa asteikon merkinnän sijaan tai usean apuviivan käyttö oktaavimerkinnän 8^{va} / 8^{vb} sijaan ovat esimerkkejä haasteista jotka vaikuttavat luettavuuteen käsialasta riippumatta. [3]

Tässä työssä käsitellään nuottikirjoituksen muuntamista koneelle luettavaan muotoon graafi- ja konvoluutioverkkopohjaisten menetelmien avulla. Nuotinluvun tunnistamisen tutkimuksessa näitä menetelmiä käytetään suhteellisen hyvin tuloksin OMR-prosessin pääasiallisessa tunnistusvaiheessa, joka tapahtuu esikäsittelyn jälkeen ja ennen muuntamista koneelliseen muotoon.

Tutkielmassa pyritään vastaamaan kysymyksiin: “miten nuottikirjoitusta voi muuntaa koneelle luettavaan muotoon” ja “miten kuvantunnistusta voi käyttää nuottikirjoituksen tunnistuksessa”.

Tiedonhaussa on hyödynnetty Google Scholar, Volter ja IEEE Xplore -kirjallisuuskantoja. Tiedonhaussa on käytetty materiaalin etsimisessä hakutermejä “optical music recognition” OR “OMR”, “staff line” AND (“recognition” OR “detection”) ja “music symbol” AND (“recognition” OR “detection”). Lisäksi tietoa neuro- ja konvoluutioverkkojen yleisestä toiminnasta sekä graafeista haettiin hakutermeillä “neural network”, “convolutional neural network” OR “CNN” ja “graph data structure”.

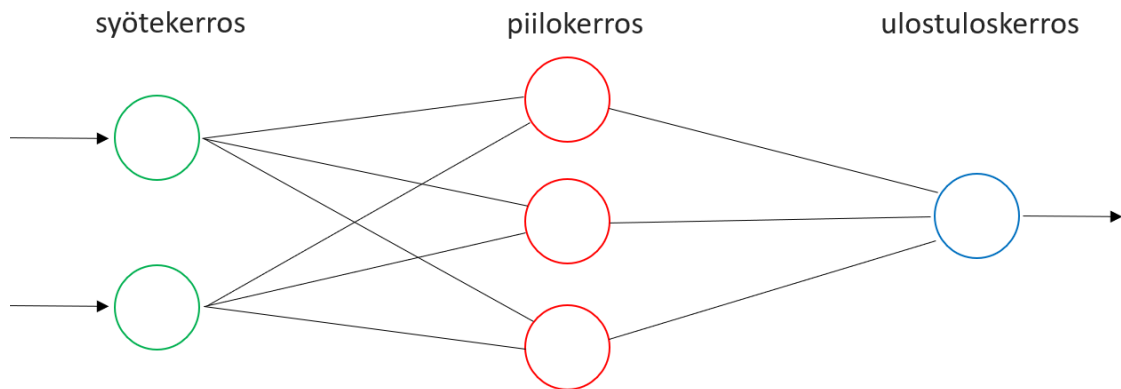
Tutkielmassa käsitellään ensin neuroverkot ja perseptronit ennen syventymistä konvoluutioverkkoihin, ja toisena graafit yleisellä tasolla. Taustakappaleiden jälkeen syvennyttään OMR:n yleisesti sovellettaviin vaiheisiin luvussa 3, ja lopuksi esitetään tutkielman yhteenveto.

2 Neuroverkot ja graafit

2.1 Neuroverkot

Neuroverkot ovat laskennallisista neuroneista koostuvia järjestelmiä, jotka prosessoivat vastaanottamiaan signaaleja keinotekoisen ajattelun tavoin [4]. Neuronit yhdistettynä syötekerrokseen muodostaa yksinkertaisen neuroverkon eli perseptronin, joka pystyy ratkaisemaan lineaarisia ongelmia, ja yhdistelemällä perseptroneita toisiinsa peräkkäin voidaan muodostaa monimutkaisempiin ongelmiin kykeneviä kytkettyjä neuroverkkoja. [5]

Neuroverkoilla on kolme pääasiallista kerrosta: syötekerros (engl. input layer), piilokerros (engl. intermediate/hidden layer) ja ulostulokerros (engl. output layer). Syöte- ja ulostulokerroksia on molempia vain yksi, kun taas piilokerroksia voi tehtävistä ja sen monimutkaisuudesta riippuen olla yhdestä lukemattomiin. Neuroverkon tyypistä riippumatta syöte- ja ulostulokerrosten toiminta pysyy pitkälti samana, ja merkittävimmät eroavaisuudet verkkojen välillä tapahtuu piilokerroksissa. [5] Neuroverkkoja on lukuisia eri tyyppisiä, kuten takaisinkytkettyjä verkkoja (engl. recurrent neural network), mutta tälle työlle oleellista on vain konvoluutioverkot niiden ollessa pääasiallinen kuvantunnistukseen käytetty verkkotyyppi.



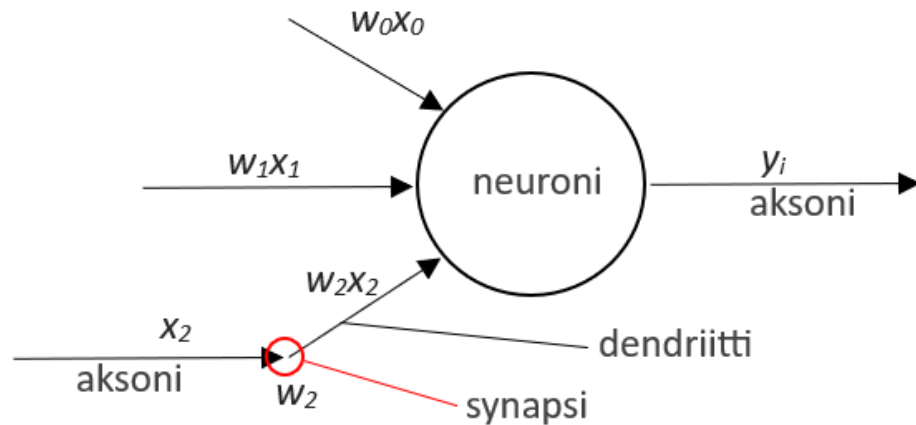
Kuva 2.1: Esimerkki yksinkertaisesta neuroverkosta.

2.1.1 Perseptronit

Perseptronit vastaanottavat tietoa dendriiteillä, käsittelevät sen neuronissa ja lähettävät sen eteenpäin aksoneilla. Aksonit puolestaan välittävät käsitellyn tiedon seuraavan perseptronin dendriitteihin synapsien välityksellä, muodostaen isommalla mittapuulla neuroverkon kokonaisuuden. Tiedon x_i välittyessä synapsin kautta kerrotaan se painokertoimella w_i , jolloin binäärinen tieto voidaan halutessaan muuttaa muiksi lukuarvoiksi. [6]

Perseptronin ja suuremmalla mittakaavalla neuroverkon koulutus tapahtuu hienosäätämällä neuronien välisiä painokertoimia. Painokertoimet alustetaan satunnaisesti, mutta testikierroksia suorittamalla ja hienosäätämällä painokertoimia algoritmi ”oppii”, eli konvergoituu kohti parempia ennusteita. [6]

Neuronien toiminta voidaan pohjustaa painotetun summan ja askelfunktion käyttöön. Painotetun summan neuronin saa summaamalla dendriittien kautta kerätyt arvot summafunktiolla $u = f\left(\sum_i w_i x_i + \theta\right)$, jossa kynnyisarvo (engl. threshold/bias) θ ohjaa summaa johonkin ennaltamäärättyyn suuntaan. Sijoittamalla summafunktio u askelfunktiolla $f(u) = \begin{cases} 1 & \text{jos } u \geq 0 \\ 0 & \text{jos } u < 0 \end{cases}$ saadaan eteenpäin lähetettävä arvo. [6]

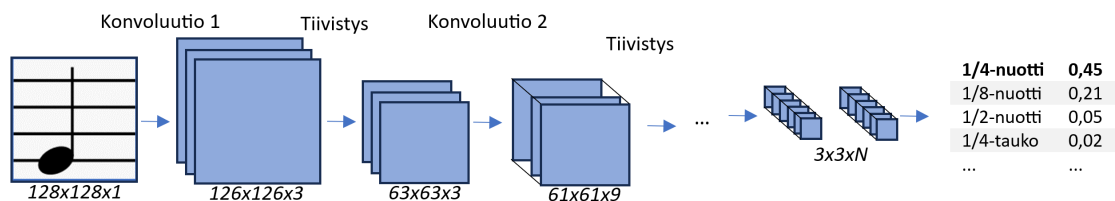


Kuva 2.2: Perseptronin rakenne.

2.1.2 Konvoluutioverkot

Konvoluutioverkko on neuroverkon tyyppi, joka hyödyntää konvoluutiokerroksia luodakseen aina korkeampilaatuisen abstraktion syötteestä syvemälle verkkoon edessä. [5] Konvoluutioverkot soveltuvat erityisesti korkean abstraktiotason tehtäviin, kuten kuvan- tai äänentunnistukseen ja robotiikkaan, mutta tässä työssä on oleellista perehtyä konvoluutioverkkoihin vain yksittäisten nuottikirjoituksen merkintöjen tunnistuksen kannalta.

Konvoluutiokerrokset muodostavat syötteestä piirrekarttoja (engl. feature map, FMap) ja poimivat yksilöiviä piirteitä konvoluutiosuodattimien (engl. filter) avulla, säilyttäen syötteen ominaiset piirteet alhaiselle abstraktiotasolle asti. Konvoluutiokerrosten operaatioiden jälkeen piirrekartat siirtyvät koontikerrosten (engl. fully-connected layers) läpi luokittelua varten ennen lopullisen ennusteen antoa ulostulokerroksesta. Konvoluutioverkossa on tarkoituksesta riippuen viidestä tuhanteen

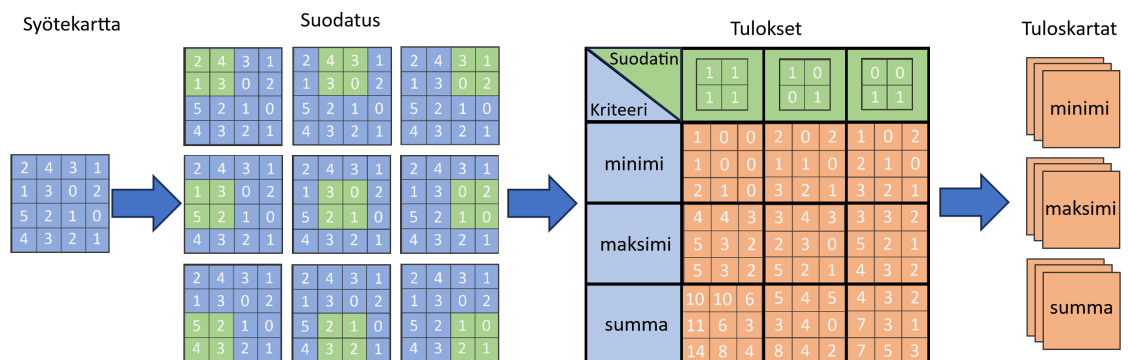


Kuva 2.3: Konvoluutioverkon toiminta.

konvoluutiokerrosta ja yhdestä kolmeen koontikerrosta.[5]

Suodatuksen yhteydessä syötteen ulottuvuudet kaventuvat suodattimen ulottuvuuksien perusteella, suurempi suodatin yhdistää laajempia alueita syötteestä kun taas pienempi suppeampia. Suodatusta varten on suodattimen koon lisäksi määriteltävä myös sen askelväli (engl. stride), jonka avulla tiedetään suodattimen liikumaetaisyys syötteen ulottuvuuksissa. Myös askelväli vaikuttaa tulokartan (engl. output FMap) ulottuvuuksiin, sillä suuremmalla askelvälillä tulokarttaan yhdistetään harvempia alueita syötekartoista (engl. input FMap).[5]

Konvoluution lisäksi voidaan käyttää myös tiivistystä (engl. pooling) kaventamaan syötteen ulottuvuuksia ja siten tehostamaan suurikokoisen syötteen käsittelyä. Tiivistys kaventaa syötteen ulottuvuuksia yhdistämällä käsittelykentän (engl. receptive field) arvot määritellyn valintakriteerin perusteella. Valintakriteerillä voidaan priorisoida esimerkiksi alueen pienin tai suurin arvo, tai arvojen laskettu keskimäärä. Useimmiten tiivistykseen annetut mitat vastaavat askelmittaa, että tiivistettäviä osia ei jäisi välistä tai tiivistettäisi useampaan osioon tulokarttaa. [5]



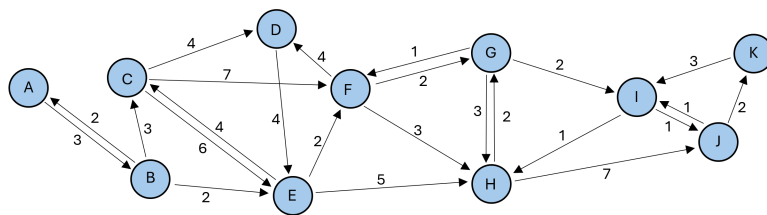
Kuva 2.4: Konvoluutio. Kuvassa mallinnettu 3 erilaista suodatinta ja valintakriteeriä (minimi, maksimi ja summa), ja näiden koostetut tulokartat.

2.2 Graafit

Graafi on tietorakenne, jossa jokaisella solmulla on arvo ja tieto yhteydestä muihin solmuihin. Toisiinsa yhteydessä olevia solmuja yhdistetään kaarilla. Toisin kuin puu-tietorakenteissa graafeilla ei ole topologista juuri-lehti-hierarkiaa, joka lyhyesti tarkoittaa että kaikki solmut voivat yhtä lailla olla suorassa yhteydessä toisiinsa. [7] Graafit voidaan jakaa suunnattuihin ja suuntaamattomiin graafeihin, mutta myöhemmin tässä työssä käsitellyille menetelmille on oleellista esitellä vain suunnatut. Osana graafien käsittelyä myöhempiä menetelmiä varten esitellään myös Dijkstran algoritmi, jolla pystytään hakemaan lyhin reitti kahden eri solmun välille.

Suunnatuille graafeille on ominaista, että jokainen solmujen välinen yhteys on suunnattu alkavan toisesta solmusta ja päättyvän toiseen solmuun, jolloin yksittäisten solmujen voi kuvitella olevan toistensa alku- ja päätöspisteitä. Suunnatuilla graafeilla on mahdollista kulkea solmujen välillä molempiin suuntiin, mutta tulee kuitenkin huomioida solmujen välisten kaarten olevan mahdollisesti eri painoisia. [7]

Dijkstran algoritmi on hakualgoritmi, joka löytää alhaisimman kustannuksen reitin kahden graafin solmun välillä ajassa $O(n^2)$, olettaen että kohdesolmun sijainti ja solmujen väliset painot ovat tiedossa. Algoritmi toimii tarkastelemalla sen hetkisen solmun tavoitetta lähempänä olevia naapurisolmuja, ja valitsee seuraavaksi solmukseen sen, jonka kanssa se jakaa pienimmän painon kaaren. Algoritmia voi siis kutsua “ahneeksi”, sillä valitsee parhaimmalta vaikuttavan reitin ilman varmaa tietoa muista poluista. [7]



Kuva 2.5: Esimerkki suunnatusta, painotetusta graafista.

3 Nuotinluku kuvantunnistuksen avulla

Nuotinluku koneellisin menetelmin (engl. Optical Music Recognition, OMR) tutkii tapoja muuntaa nuottikirjoitusta koneelle käsiteltävään muotoon laskennallisoin keinoin. Yhteinen määritelmä on kuitenkin kiistanalainen, sillä aihealueen tutkimus on luottanut intuitiiviseen yhteisymmärrykseen käsitteen tarkoituksesta. Tarkkojen raamien määrittelemättä jättäminen on näin johtanut aihepiirin tutkimuksen hajanaisuuteen, ja suurempien kokonaisuuksien määrittelyn tai ratkaisun sijaan keskittyminen kohdistuu yksittäisiin haasteisiin osana laajempaa kokonaisuutta. [1] [2] Tästä syystä seuraavissa kappaleissa esitetäänkin vain eri lähteiden ehdottamia ratkaisuja OMR:n yleisesti käytetyn rungon eri vaiheisiin.

Nuotinluku voidaan soittajan näkökulmasta jakaa vaiheisiin luku, tekniikka ja tulkinta. Lukuvaiheessa soittaja käy läpi näkemänsä symbolit, tekniikkavaiheessa johtaa luettujen symbolien vuorovaikutukset ja tulkintavaiheessa päättää miten nämä tulisi ilmaista. Ensimmäiset kaksi vaihetta ovat lähes kiistattomia tarkan symboliikan vuoksi, mutta kolmannen vaiheen ollessa liian soittajakohaista keskitytään OMR:n kohdalla vain kahteen ensimmäiseen vaiheeseen eli symboliikan tallentamiseen ja merkitykselliseen yhdistämiseen. [2] Nämä vaiheet voidaan OMR:n avulla ilmaista neliosaisena jatkumona: kuvan esikäsittely, sisällön tunnistus, tiedon rekonstruktio ja tiedon tallennus [8].

3.1 Kuvan esikäsittely

Esikäsittelytoimenpiteitä hyödynnetään monissa kuvantunnistusalgoritmeissa tarkkuuden parantamiseksi, ja OMR:ssa yleisimmiten käytettyjä esikäsittelymenetelmiä ovat vinoumankorjaus, kohinanvaimennus, tarkennus ja binäärisaatio, mutta on olemassa muitakin laatuun ja piirteisiin vaikuttavia toimenpiteitä, kuten syötteen morfologinen muuntelu. [8] Esikäsittely onkin vakiintunut käyttöön monilla OMR:iin kehitetyillä algoritmeilla, sillä esimerkiksi jo kahden asteen vinoutuma syötteessä voi vaikuttaa saatuihin tuloksiin [9]. Myöhempien vaiheiden menetelmät olettavat syötteen olevan esikäsitelty seuraavaksi selitettävien menetelmien mukaisesti.

Vinoumankorjauksella pyritään saamaan syöte muotoon, jossa nuottiviivasto olisi mahdollisimman vaakasuorassa. Kohinanvaimennuksella tarkoitetaan syötteen värien ja kirkkauden aiheuttamien vääristymien korjaamista nuottikirjoituksessa, ja tarkennuksella puolestaan kontrastin, kirkkauden tai muun ominaisuuden muuttamista suodattimien avulla tarkemman kuvan saamiseksi. Binärisaatiolla tarkoitetaan OMR:n kontekstissa syötteen kaksiarvoistamista, eli kaiken värjäämistä joko mustaksi tai valkoiseksi ilman välisävyjä. Binärisaation pääasiallinen tarkoitus on helpottaa yksittäisten merkintöjen tunnistamista, sillä muuten mahdolliset sävyt voivat algoritmin tulkinnasta riippuen kuulua osaksi jotain symbolia, tai olla vääristymiä joko esimerkiksi valaistuksen tai fyysisen syötteen luonnollisen kuluman takia. Binärisaatiota voidaankin tämän takia suorittaa alueellisesti yleisen käsittelyn lisäksi. [8]

3.2 Sisällön tunnistus

Monet OMR-algoritmit perustuvat ensin nuottiviivaston erottamiseen syötteestä joko eristämällä tai poistamalla se kokonaan, josta jatketaan muiden merkintöjen tunnistamiseen nuottiviivastoista saatujen raamien perusteella. Nuottiviivasto ero-

tetaan usein syötteestä ennen muiden symbolien tutkimista, koska sen avulla voidaan tarkentaa symbolien tunnistusta etenkin käsinkirjoitettujen nuottien lukemisessa, joissa viivastot eivät välttämättä ole kohtisuorassa toisiaan kohden. Viivastot saattavat olla myös kohdittain vinoutuneita, mikä hankaloittaa selkeiden linjojen tunnistamista entisestään, puhumattakaan muiden symbolien päällekkäisyyden tai käsinkirjoitetun paperin kohdalla normaalikulumisen vaikutuksesta esikäsitellynkin syötteen luettavuuteen. Laajasta tutkimuksesta huolimatta nuottiviivastojen tunnistukselle ei ole vielääkään saatu kehitettyä kuin vain tyydyttävästi toimivia keinoja viivastojen laadun ja muodon vaihtelun vuoksi. [8] Tässä työssä otetaan nuottiviivastojen tunnistamisen menetelmistä esille vakaimman reitin menetelmä (engl. stable path approach), joka on graafeihin ja dijkstran algoritmiin perustuva ratkaisu suhteellisen hyvällä tehokkuudella.

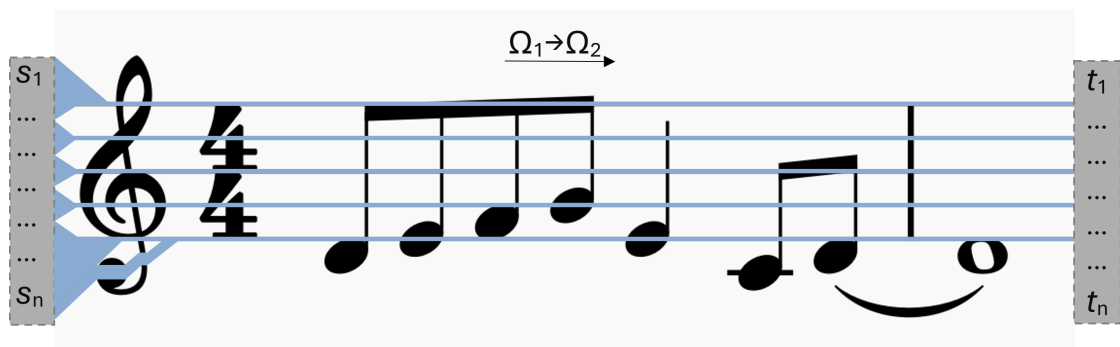
Nuottiviivastojen erottaminen on kuitenkin vasta ensimmäinen askel symbolien tunnistamisessa, ja vaikka monet algoritmit yhdistävätkin viivastojen ja muiden symbolien tunnistuksen yhteen vaiheeseen esimerkiksi Markovin mallien (engl. Markov Model) avulla, voidaan tätä vaihetta tarvittaessa jakaa osiin tutkimustyön helpottamiseksi. Muiden symbolien tunnistus nojautuu nuottiviivaston tiedon tuntemiseen, esimerkiksi nuottien sävelkorkeus voi vääristyä jos ei voida olla varmoja sen suhteellisesta sijainnista nuottiviivastoa kohden. Symbolien tunnistukseen vaikuttaa myös nuottiviivaston tunnistamisen tarkkuus, sillä etenkin menetelmissä missä nuottiviivasto poistetaan syötteestä käsittelyvaiheessa voi muiden symbolien tunnistamiseen kriittiset piirteet kärsiä tai kadota kokonaan. Joka tapauksessa hyvinkin toteutettu nuottiviivaston tunnistaminen vaikuttaa muiden symbolien tunnistamiseen, sillä monet symbolit ovat päällekkäin viivastojen kanssa ja pelkkä viivaston tunnistaminen jättää algoritmin arvioitavaksi, onko syötteen osa pelkkää symbolia, viivastoa vai molempia. Viivaston poistaminen puolestaan jakaa symbolit osiin, vaikeuttaen yksittäiseen symboliin kuuluvien osuuksien yhdistämistä toisiinsa. [8]

muodostaminen vastakkaiselle puolelle. Polun päätöspisteen ei tarvitse olla erillinen jokaiselle iteraatiolle, joten nuottiviivastojen tunnistamisen käyttötapauksessa polut konvergoituvat optimaalisimmalle reitille, eli nuottiviivan päälle. Varsinaisen polun tehokkuus voidaan laskea hyödyntämällä funktiota $F_{\Omega_1 \rightarrow \Omega_2}()$, jossa polun $P_{s,t}$ toisen reunan piste $s \in \Omega_1$ ja vastakkaisen reunan piste $t \in \Omega_2$, josta voidaan johtaa funktio $F_{\Omega_1 \rightarrow \Omega_2}(F_{\Omega_2 \rightarrow \Omega_1}(s)) = s$ silloin, jos polku $P_{s,t}$ on alhaisimman kustannuksen polku. [10]

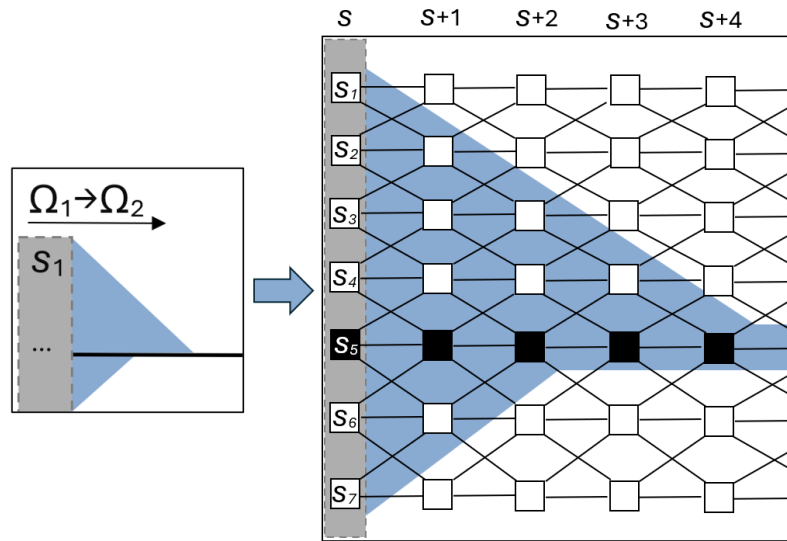
Nuottiviivaston etsimisen tapauksessa voidaan olettaa, ettei graafissa etsitä suoraan ylös, alas tai tulosuuntaan haarautuvia yhteyksiä, sillä tavoitteena on löytää lyhin vaakasuunnassa oleva kevyiden solmujen yhteys, eli syötettä voidaan tarvittaessa pitää suunnattuna graafina toisesta laidasta toiseen menetelmän vaiheesta riippuen. [10] Nuottiviivan tunnistaminen tapahtuu siis etsimällä tehokkain reitti syötteen reunasta toiseen, ja viivoja ollessa viisi huomioidaan sen lisäksi myös neljä sen jälkeen alhaisimman kustannuksen reittiä suhteellinen etäisyys huomioiden.

3.2.2 Symbolien tunnistaminen konvoluutioverkkojen avulla

Konvoluutioverkot ovat kehittyneet viime vuosien aikana yhdeksi tehokkaimmista menetelmistä OMR:n tutkimuksessa ja ovatkin nopeasti ohittaneet jo yli 90% tarkkuuden yksittäisten nuottien tunnistamisessa. Hyödyntämällä monta, tuloksiaan



Kuva 3.2: Osagraafin Ω_1 pisteistä $s_1 \dots s_n$ lähtevät polut konvergoituvat nuottiviivastolle.



Kuva 3.3: Tarkennus pisteistä lähtevien polkujen konvergoitumisesta.

vertaavaa konvoluutioverkkoa yhdessä ReLU-aktivaatiofunktion (Rectified Linear Unit) $f(x) = \max(0, x)$ kanssa voidaan saavuttaa jopa yli 95% tarkkuus käsinkirjoitettujen symbolien tunnistuksessa. Nuottikirjoituksen ollessa OMR:n esikäsittelyvaiheen jälkeen binärisoitua voidaan konvoluutioverkkojen toimintaa myös tehostaa vähentämällä konvoluution aikana luotavia piirrekarttoja vain yhteen, vähentäen muuten eksponentiaalisesti kasvavaa suoritustehovaatimusta. [11] Kuten siis muissakin kuvantunnistuksen algoritmeissa, niin myös nuottikirjoituksessakin esiintyviä symboleita voidaan siis tunnistaa rakentamalla tarpeelliset verkot ja kouluttamalla niille oleellinen tieto.

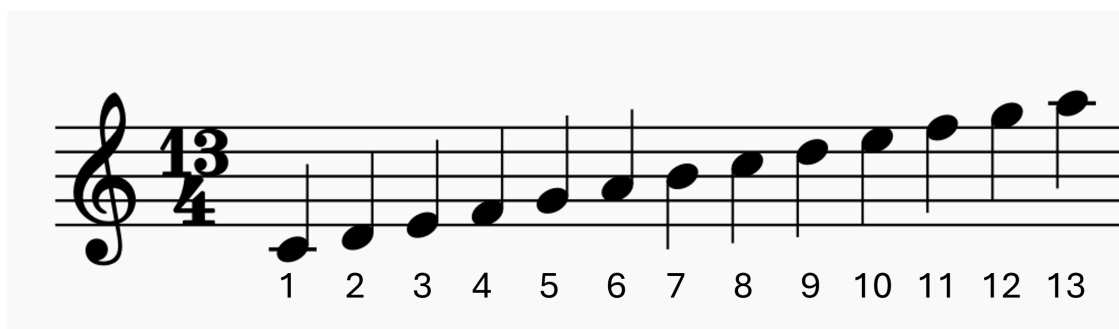
Symbolien varsinaisen tunnistuksen jälkeen jää kuitenkin vielä tunnistusvaiheessa kerättävän semanttisen tiedon tunnistaminen, joista nuottien sävelkorkeuden määrittäminen on tärkein lopullisen tuloksen tarkkuuteen vaikuttava muuttuja. Yleisnohtaatioissa nuotin kesto voidaan ilmaista nupin sisällön, nuotin varren tai sen palkkien avulla, mutta nupin pystysuuntainen sijainti on ainoa suhteellista sävelkorkeutta määrittävä tekijä. [12] Nuotteja tunnistessa on siis mahdollista tehostaa neurotai konvoluutioverkon toimintaa opettamalla yksittäisten symbolien tunnistamisen lisäksi myös sävelkorkeuden tunnistamista erikseen esimerkiksi suhteuttamalla nuot-

tien nupit taustan nuottiviivastoon, tai vaihtoehtoisesti sijaintiin syötteessä, mutta tässä työssä sävelkorkeuden tunnistamista tarkastellaan vain nuottiviivastoon suhteutettuna.

Nuotit voidaan jakaa nupin sijainnin mukaan 13 eri sävelkorkeuteen lähes kahden oktaavin alalle, jossa matalin on G-avaimella $c(n)$ ja korkein $a(n+1)$, jossa n merkitsee oktaavia. Jotta konvoluutioverkko saisi tarpeeksi koulutusmateriaalia luotettavien lopputulosten saamiseksi, tulee sille antaa riittävästi esimerkkejä jokaisesta 13 luokasta eri konteksteissa, kuten vinoutuneena, eri kohdissa syötettä tai yhdessä muiden alle neljäsosanuotin pituisten nuottien kanssa, jolloin nuotteja yhdistetään palkilla. [12] Tulee myös huomioida, että tarvittun koulutusmateriaalin määrä luotettavia tuloksia varten käsinkirjoitettujen nuottien tunnistamiseen on suurempi kuin digitaalisten, sillä käsinkirjoitetut nuotit ovat huomattavasti vaihtelevimpia käsialaltaan.

3.3 Tiedon rekonstruktio

Varsinaisten symbolien tunnistuksen ja erittelyn jälkeen tulee nämä vielä yhdistää toisiinsa merkittävällä, koneelle ymmärrettävällä tavalla. Ideaali OMR:n menetelmä hyödyntää lopullista tallennusmuotoa, jossa voidaan antaa erityistä painoarvoa merkintöjen sijainnille, sillä toisin kuin esimerkiksi tavallisen kirjoituksen tunnistuk-

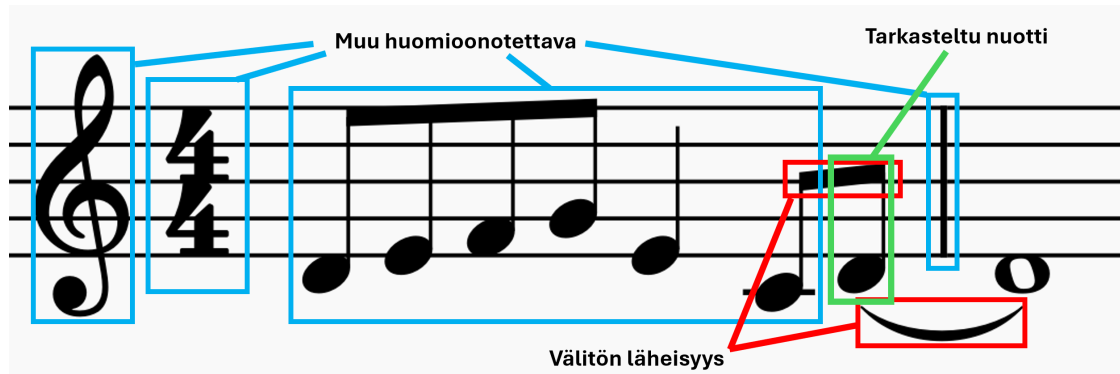


Kuva 3.4: Sävelkorkeuksien numerointi tunnistamista varten.

sessä (engl. Optical Character Recognition, OCR) tulee nuottikirjoituksen symbolleissa ottaa kirjoitusrivistä huomioon merkintöjen vaakaulotteisen sijainnin lisäksi myös pystyulotteinen, kuin myös monet muut laajempaa kokonaisuutta määräävät tekijät kuten legato- tai oktaavimerkinnät. Monet tutkimukset ovatkin siksi kehittäneet tähän tarkoitukseen kielioppimalleja (engl. grammar), jotta monitulkintaiset ja suurien kokonaisuuksien laajuisten symbolien vaikutukset voidaan ottaa huomioon tehokkaasti lopputulokseen. [13]

Kielioppimallien tehtävänä on määritellä algoritmille rekonstruktioleolliset raamit, eli miten saatu tieto tulee käsitellä, mitä nuottikirjoitukselle on oleellista ollakseen oikein kirjoitettua ja miten nuotit sekä muut symbolit tulee toisistaan eritellä. Hyvin toteutetut kielioppimallit siis tehostavat lopullisen algoritmin suorittamista. Kielioppimallit ovat kuitenkin sellaisenaan vain välittömän läheisyyden huomioonottavia menetelmiä tiedon käsittelyssä, joten ilman muita keinoja laajempien kokonaisuuksien huomioimiseen voi oleellinen tieto yksittäisille merkinnöille laajemmassa kokonaisuudessa jäädä ottamatta huomioon. Monet OMR-algoritmit hyödyntävätkin tästä syystä kielioppimallien lisäksi jäsentelijää (engl. parser) kaksivaiheisessa prosessissa kokonaisuuden hahmoittamiseksi. [13]

Kielioppimallin jäsentelijän tehtävänä on tuoda musiikillisen kontekstin semanttinen tieto, kuten sävellaji ja kertausmerkinnät, mukaan symbolien graafisten kuvausten käsittelyyn. Näin voidaan kohentaa annettujen tulosten tarkkuutta verrattuna vain symbolien suoraan, pelkkään kuvauspohjaan perustuvaan tunnistukseen. Jäsentelijä onkin olennainen osa korkeamman tasoista kielioppimallia kuten määritellyn lauseen kielioppimallia (engl. definite clause grammar, DCG). DCG:ssä toimitaan kaksivaiheisen rakenteen mukaisesti: ensin käsitellään ja leimataan graafinen sisältö, ja seuraavaksi käsitellään ensimmäisen vaiheen sisällön syntaksi, eli käytännössä tarkastetaan ensimmäisen vaiheen antamat tulokset virheiden varalta. Yleisesti korkeamman tason kielioppimalleissa vaihejaossa alempi taso tunnistaa nuotin



Kuva 3.5: Yksittäisen nuotin tarkastelua. Ylempi taso käsittelee sinisellä rajatut merkinnät, kun taas alempi taso vihreällä ja punaisella rajatut.

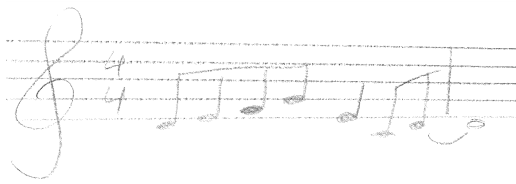
ja sen välittömän läheisyyden, kuten staccato-merkinnät, kun taas ylempi taso hahmottaa fraaseja ja laajempia kokonaisuuksia, eli esimerkiksi sävellajin määrittelemät etumerkit tai dynamiikkamerkinnot. Korkeampi taso myös varmistaa alemman tason merkintöjen olevan loogisia suhteuttamalla niiden kestoja tahtilajiin ja -viivoihin, tarkistaen niiden kestoja olevan annettujen raamien mukaisia. [8] [13]

3.4 Tiedon tallennus

Tiedon tallennuksella tarkoitetaan tässä tiedon rekonstruktion jälkeen tai yhdessä sen kanssa tapahtuvaa varsinaisen tiedon tallentamista koneelle ymmärrettävään muotoon, esimerkiksi MIDI- tai MusicXML-tiedostoksi. Usein tämä vaihe tehdään yhdessä tiedon rekonstruktion kanssa, mutta selkeyden vuoksi on se eritelty kuitenkin erilliseen vaiheeseen yleisessä OMR:n tutkimuksen rakennejaossa. [8]

Tallennettaessa saatua syötettä tiedostoksi on oleellista valita tarkoitukselle sopiva tallennusmuoto. Monet tallennusmuodot eivät sovellu nuottien välisten suhteiden tallentamiseen, ja jotkin jättävät myös yksittäisten nuottien semanttiset piirteet huomioimatta. MusicXML-tallennusmuoto kykenee huomioimaan nuottikirjoituksen tekniset piirteet, muttei kykene tallentamaan yleisnotaation ulkopuolisia soveltavia merkintöjä, kuten vapaamuotoisia nuolia, viivoja tai adverbejä, joita esimerkiksi

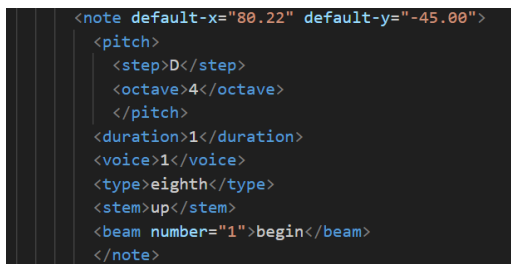
säveltäjä on saattanut nuottiin kirjoittaa soittajaa varten tulkittavaksi omalla tavallaan. Kehittyneempiä tallennusmuotoja ovat esimerkiksi MEI (Music Encoding Initiative) ja RDF (Resource Description Framework), jotka kykenevät MusicXML:n ominaisuuksien lisäksi muun muassa tulkitsemaan eri äänien välisiä suhteita paremmin, mutta lopulliselle tallennusmuodolle ei vielä ole yhtä yleistä standardia monipuolisempien tallennusmuotojen ollessa vasta kehitysvaiheessa. [8]



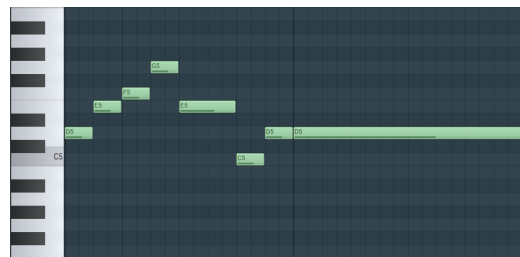
(a) Skannattu katkelma kappaleesta



(b) Digitaalinen kuva katkelmasta



(c) Nuotti MusicXML-tiedostossa.



(d) MIDI-tiedosto sovelluksessa.

Kuva 3.6: Esimerkkejä nuottikirjoituksen tallennusmuodoista.

4 Yhteenveto

Nuottikirjoituksen tunnistus on monivaiheinen, monelta eri näkökulmalta lähestyttävissä oleva prosessi. Kuvasyötettä voidaan esikäsitellä tarpeen mukaan vinouman korjauksen, kohinanvaimennuksen, tarkennuksen ja binärisaation avulla, sekä harvemmin myös morfologisten operaatioiden keinoin. Esikäsitelty syöte helpottaa etenkin kuvattujen tai skannattujen käsikirjoitettujen syötteiden käsittelyä tunnistusvaiheessa. Tunnistusvaihe voidaan jakaa kahteen osioon: nuottiviivastojen erottamiseen ja yksittäisten merkintöjen tunnistamiseen. Tämä vaihejako helpottaa yksittäisten merkintöjen tunnistamista, sillä monilla merkinnöillä on vakiintunut sijainti nuottiviivastolla, ja muilla merkinnöillä sijainnilla on merkitystä sen piirteiden, esimerkiksi sävelkorkeuden, kuvaamisessa. Symbolien tunnistamisen jälkeen tulee vielä tunnistaa niiden semanttinen tieto, joka onnistuu hyödyntämällä kielioppimalleja. Kielioppimallit kykenevät yhdistämään laajempia fraaseja käsitteleviä merkintöjä, joten niiden soveltaminen tiedon tallentamisessa on oleellista laadukkaan lopputuloksen saavuttamiseksi. Lopullisella tallennusmuodollakin on merkitystä, sillä kaikki tallennusmuodot eivät tue korkeamman abstraktiotason tiedon, kuten äänenvoimakkuutta kuvaavien merkintöjen, tallentamista.

Tutkielman alussa esitettiin tutkimuskysymykset “miten nuottikirjoitusta voi muuntaa koneelle luettavaan muotoon” ja “miten kuvantunnistusta voi käyttää nuottikirjoituksen tunnistuksessa”. Jotta nuottikirjoitusta voi muuntaa koneelle ymmärrettävään muotoon tulee kuvasyöte esikäsitellä, jotta kuvantunnistusalgoritmin toi-

minta olisi tehokkaampaa. Nuottiviivastojen tunnistaminen onnistuu graafien avulla ja muiden merkintöjen tunnistuksessa voidaan hyödyntää kuvantunnistusta konvoluutioverkkojen avulla. Semanttisen sisällön nuottikirjoituksesta saa tallennettua hyödyntämällä kielioppimalleja ja jäsentelijöitä, ja sopiva lopullinen tallennusmuoto tukee kielioppimallien ja jäsentelijöiden poimimien piirteiden tallentamista.

Nuottikirjoitusta voidaan siis tunnistaa suhteellisen tehokkaasti konvoluutioverkkojen ja graafien avulla, olettaen että kuvasyöte on tarkoituksen mukaisesti esikäsitelty, ja että algoritmi pystyy poimimaan nuottien lisäksi niiden semanttiset piirteet. Jääkin vielä nähtäväksi, milloin nuotinlukualgoritmit kehittyvät tehtävässä ihmistä tehokkaammaksi.

Lähdeluettelo

- [1] A. Baró, P. Riba, J. Calvo-Zaragoza ja A. Fornés, ”From optical music recognition to handwritten music recognition: a baseline”, *Pattern Recognition Letters*, vol. 123, s. 1–8, 2019.
- [2] J. Calvo-Zaragoza, J. H. Jr ja A. Pacha, ”Understanding optical music recognition”, *ACM Computing Surveys (CSUR)*, vol. 53, s. 1–35, 2020.
- [3] J. Sloboda, ”The uses of space in music notation”, *Visible Language*, vol. 15, s. 86–110, 1980.
- [4] C. Gallo, ”Artificial neural networks tutorial”, teoksessa *Encyclopedia of Information Science and Technology, Third Edition*, IGI Global, 2015, s. 6369–6378.
- [5] V. Sze, Y.-H. Chen, T.-J. Yang ja J. S. Emer, ”Efficient processing of deep neural networks: A tutorial and survey”, *Proceedings of the IEEE*, vol. 105, s. 2295–2329, 2017.
- [6] J. Singh ja R. Banerjee, ”A study on single and multi-layer perceptron neural network”, teoksessa *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, IEEE, 2019, s. 35–40.
- [7] J. E. Hopcroft, J. D. Ullman ja A. V. Aho, *Data structures and algorithms*. Addison-wesley Boston, MA, USA: 1983, vol. 175.

-
- [8] E. Shatri ja G. Fazekas, "Optical music recognition: State of the art and major challenges", *arXiv preprint arXiv:2006.07885*, 2020.
- [9] A. Pacha, "Incremental supervised staff detection", teoksessa *Proceedings of the 2nd international workshop on reading music systems*, 2019, s. 16–20.
- [10] J. dos Santos Cardoso, A. Capela, A. Rebelo, C. Guedes ja J. P. da Costa, "Staff detection with stable paths", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, s. 1134–1139, 2009.
- [11] S. Lee, S. J. Son, J. Oh ja N. Kwak, "Handwritten music symbol classification using deep convolutional neural networks", teoksessa *2016 International Conference on Information Science and Security (ICISS)*, IEEE, 2016, s. 1–5.
- [12] Andrea, Paoline ja A. Zahra, "Music note position recognition in optical music recognition using convolutional neural network", *International Journal of Arts and Technology*, vol. 13, s. 45–60, 2021.
- [13] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. Marcal, C. Guedes ja J. S. Cardoso, "Optical music recognition: state-of-the-art and open issues", *International Journal of Multimedia Information Retrieval*, vol. 1, s. 173–190, 2012.