Neuroverkkopohjaiset menetelmät unenaikaisten havahtumisten tunnistamisessa polysomnografiarekisteröinneistä

Miika Vainikka

Luonnontieteiden kandidaatin tutkielma

Sovelletun fysiikan koulutusohjelma

Itä-Suomen yliopisto, Teknillisen fysiikan laitos

28. huhtikuuta 2023

ITÄ-SUOMEN YLIOPISTO, Luonnontieteiden ja ympäristötieteiden tiedekunta
Sovelletun fysiikan koulutusohjelma
Miika Vainikka: Neuroverkkopohjaiset menetelmät unenaikaisten havahtumisten tunnistamisessa polysomnografiarekisteröinneistä
Luonnontieteiden kandidaatin tutkielma, 26 sivua
Tutkielman ohjaajat: FT, Sami Nikkonen ja FM, Henna Pitkänen
Huhtikuu 2023

Avainsanat: koneoppiminen, syväoppiminen, keinotekoinen neuroverkko, unitutkimus, PSG, EEG, havahtuminen

Tiivistelmä

Uni on ihmisen terveydelle tärkeä fysiologinen prosessi. Sen laatua voidaan arvioida polysomnografialla, jossa mitataan muun muassa unenaikaista aivosähkökäyrää, elektro-okulografiaa ja lihassähkökäyrää. Lyhytkestoiset unenaikaiset havahtumiset heikentävät unen laatua ja ovat yhteydessä unihäiriöihin, kuten uniapneaan. Koulutetut ammattilaiset tunnistavat unenaikaiset havahtumiset polysomnografiarekisteröinneistä visuaalisesti. Koska tähän kuluu paljon aikaa, viime vuosina on tutkittu erilaisia koneoppimiseen pohjautuvia automaattisia menetelmiä havahtumisten tunnistamisessa. Tässä kirjallisuuskatsauksessa perehdytään ohjattua oppimista hyödyntäviin keinotekoisiin neuroverkkoihin.

Koneoppimisessa keinotekoinen neuroverkko opettelee tunnistamaan sille syötetyn data-alkion piirteitä, joiden perusteella se voi ryhmitellä data-alkiot erilaisiin luokkiin. Ohjatussa oppimisessa nämä luokat ovat ennalta määrättyjä. Neuroverkot koostuvat toisiinsa liitetyistä keinotekoisista neuroneista, joiden välisten yhteyksien voimakkuudet määräytyvät neuroverkon kouluttamisen aikana. Yksinkertaisin neuroverkkotyyppi on täysin kytketty neuroverkko, jossa kaikki neuronit ovat yhteydessä toisiinsa. Muita neuroverkotyyppejä ovat muun muassa konvoluutio-operaatiota hyödyntävä konvoluutioneuroverkko ja pitkän aikavälin riippuvuuksia mallintava LSTM-neuroverkko.

Tässä tutkielmassa esitellään viisi erilaista havahtumisten tunnistamiseen kehitettyä neuroverkkomallia. Malleista jokaisessa hyödynnettiin konvoluutiokerroksia ja yhdessä myös kaksisuuntaista LSTM-kerrosta. Mallien suorituskykyä arvioidaan erilaisilla matemaattisilla mittareilla, jotka kertovat havahtumisten tunnistamisen keskimääräisestä tarkkuudesta.

Tutkielmassa tarkasteltujen tutkimusten perusteella neuroverkoilla voidaan tunnistaa havahtumisia automaattisesti hyödyntämällä polysomnografiarekisteröinnin aivosähkökäyrää, elektro-okulografiaa ja leuan lihassähkökäyrää. Lisäksi muiden signaalien, kuten sydänsähkökäyrän ja hengitysilmavirtauksen käyttäminen voi parantaa havahtumisten tunnistamisen tarkkuutta. Konvoluutiokerrokset olivat olennainen osa tutkielmassa tarkastelluissa havahtumisia tunnistavissa neuroverkoissa, sillä ne keräsivät polysomnografiarekisteröinneistä piirteitä, joiden perusteella havahtumiset voitiin tunnistaa. LSTM-kerrosten lisääminen konvoluutiokerrosten yhteyteen saattoi lisätä havahtumisten tunnistamisen tarkkuutta yhdessä tutkimuksessa. Neuroverkkopohjaisten mallien kehittäminen ja käyttöönotto kliinisessä työssä voisi tulevaisuudessa mahdollistaa nopeamman ja kustannustehokkaamman havahtumisten annotoinnin.

Sisällys

1 Johdanto

Uni on ihmisen terveydelle ja elämänlaadulle tärkeä fysiologinen prosessi, jonka on havaittu olevan yhteydessä esimerkiksi oppimiseen, muistiin, tunteiden säätelyyn ja kuona-aineiden poistamiseen keskushermostosta [1]. Unen laatua voidaan arvioida polysomnografialla (engl. *polysomnography*, PSG), jossa mitataan muun muassa unenaikaista aivosähkökäyrää (engl. *electroencephalogram*, EEG), elektro-okulografiaa (engl. *electrooculogram*, EOG), lihassähkökäyrää (engl. *electromyogram*, EMG), sydänsähkökäyrää (engl. *electrocardiogram*, EKG) sekä hengityksen ilmavirtausta ja veren happisaturaatiota [2]. Polysomnografian lisäksi on olemassa myös muita unen arviointimenetelmiä, kuten liikeaktiviteettia mittaava aktigrafia (engl. *actigraphy*) [3], mutta tässä tutkielmassa keskitytään ainoastaan polysomnografiaan.

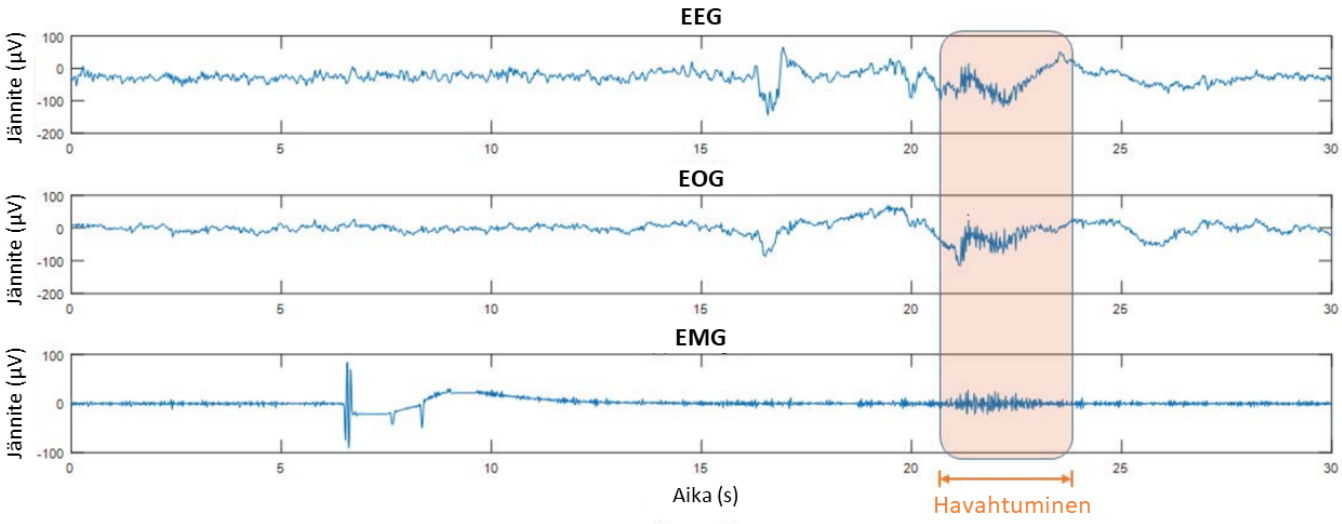
Unen laatua voidaan tarkastella luokittelemalla uni PSG-rekisteröinnissä näkyvien piirteiden perusteella eri univaiheisiin. Näitä vaiheita ovat valve, vilkeuni (engl. *rapid eye movement sleep*, REM-uni) ja perusuni (engl. *non-rapid eye movement sleep*, NREM-uni), joka jaotellaan edelleen N1-, N2- ja N3-vaiheisiin [4]. Luokittelu tehdään 30 sekunnin mittaisissa ajanjaksoissa Amerikan unitutkimusakatemian (engl. *American Academy of Sleep Medicine*, AASM) asettamien ohjesääntöjen mukaisesti [5]. AASM:n ohjeiden mukaan 30 sekunnin unijakso luokitellaan valveeksi, jos ihmisen katsotaan PSG-rekisteröinnin perusteella olleen jakson aikana hereillä yli 15 sekuntia [5]. Ihmisen uni voi kuitenkin keskeytyä myös alle 15 sekunniksi, jolloin se ei näy luokitelluissa univaiheissa.

Perus- ja vilkeunen aikana voi esiintyä myös havahtumisia, joissa ihmisen tajunnan taso lisääntyy johtamatta kuitenkaan valveeseen [5, 6]. AASM määrittelee havahtumiset äkilliksi EEG:n taajuuden muutoksiksi, jotka kestävät vähintään kolme sekuntia ja joita edeltää vähintään kymmenen sekuntia tasaista unta. REM-unen aikaisissa havahtumisissa aktiivisuuden on lisäännyttävä myös leuan EMG-signaalissa vähintään yhden sekunnin ajaksi. Lisäksi AASM:n mukaan havahtumisten määrittämistä voidaan parantaa käyttämällä niiden tunnistamisessa myös muita PSG-rekisteröinnin signaaleja [5]. Kuvassa 1 on esitetty unenaikaisen havahtumisen aiheuttamat muutokset PSG-rekisteröinnin EEG-, EOG- ja leuan EMG-signaaleissa.

Unenaikaisten havahtumisten tunnistaminen on tärkeää, sillä niiden liiallinen esiintyminen aiheuttaa unen pirstaloitumista, joka voi johtaa päiväaikaiseen väsymykseen ja huonompaan elämänlaatuun [2, 7]. Lisäksi tietoa havahtumisista voidaan käyttää hyödyksi unihäiriöiden, kuten uniapnean, diagnosoinnissa [6]. Havahtumisten tunnistamisesta vastaavat unihäiriöiden nykydiagnostiikassa koulutetut ammattilaiset, jotka tarkastelevat PSG-rekisteröintejä visuaalisesti [4]. Havahtumisten annotoinneissa on kuitenkin merkittäviä ammattilaisten välisiä eroja [2, 7]. Lisäksi PSG-dataa kertyy jo yhden yön aikana valtavasti, joten sen analysointiin voi kulua yhdeltä ammattilai-

selta useita tunteja [8]. Koneoppimiseen pohjautuvilla automaattisilla menetelmillä havahtumiset voitaisiin sen sijaan tunnistaa PSG-rekisteröinneistä johdonmukaisesti jopa muutamassa minuutissa [2]. Koneoppimista hyödyntämällä havahtumisten tunnistaminen voitaisiin täten saada huomattavasti manuaalista määrittämistä kustannustehokkaammaksi [9]. Tässä tutkielmassa perehdytään erityisesti syväoppimiseen pohjautuviin eli keinotekoisia neuroverkkoja hyödyntäviin koneoppimismenetelmiin havahtumisten tunnistamisessa. Muut automaattiset menetelmät jätetään tutkielman ulkopuolelle.

Tutkielmassa kerrotaan ensin koneoppimisesta yleisesti ja esitellään kolme eniten käytettyä tyyppiä keinotekoisista neuroverkoista: eteenpäin kytketty neuroverkko, konvoluutioneuroverkko ja takaisinkytkeytyvä neuroverkko. Lisäksi tutkielmassa perehdytään lyhyesti neuroverkkojen kouluttamiseen ja niiden testaamiseen. Lopuksi tutkielmassa esitellään viisi erilaista havahtumisten tunnistamiseen kehitettyä neuroverkkomallia sekä vertaillaan niiden suorituskykyjä.



Kuva 1: Esimerkki unenaikaisesta havahtumisesta, joka voidaan nähdä äkillisenä taajuuden muutoksena aivosähkökäyrässä (EEG), elektro-okulografiassa (EOG) ja leuan lihassähkökäyrässä (EMG). Havahtuminen on korostettu kuvassa oranssilla laatikolla. Kuva on muokattu lähteestä [10].

2 Koneoppiminen

Koneoppimisella tarkoitetaan tietojenkäsittelytieteen osa-aluetta, joka tutkii algoritmeja ja malleja, jotka voivat oppia käsittelemästään datasta. Oppimisella tarkoitetaan, että algoritmi voi tunnistaa datalle ominaisia piirteitä tai suorittaa tiettyjä tehtäviä itsenäisesti ilman suoraa ohjelmointia. Tom Mitchell määritteli tietokonealgoritmin oppimisen vuonna 1997 siten, että algoritmin sanotaan oppivan kokemuksesta E suhteessa johonkin tehtäväluokkaan T ja suorituskykymittariin P, jos algoritmin suorituskyky P paranee tehtävässä T kokemuksen E myötä [11].

Koneoppimismenetelmillä voidaan ratkaista monenlaisia tehtäviä, jotka ovat liian työläitä perinteisellä ohjelmoinnilla suoritettaviksi [12]. Koneoppimismenetelmillä ratkaistavat ongelmat voivat olla kuitenkin ihmiselle helppoja, mutta useissa tapauksissa koneoppimisalgoritmi suorittaa ne ihmistä huomattavasti nopeammin ja jopa paremmin [8, 9, 13].

Tehtäviä, joissa koneoppimista voidaan hyödyntää ovat muun muassa luokittelu, regressio, transkriptio, kääntäminen, poikkeamien havaitseminen ja tiheysfunktion estimointi [12]. Kenties tyypillisimmässä esimerkissä koneoppimista hyödynnetään käsin kirjoitettujen numeroiden tunnistamisessa ja niiden luokittelussa vastaaviin numeroluokkiin [12]. Myös havahtumisten tunnistaminen PSG-rekisteröinnistä voidaan muotoilla luokittelutehtäväksi, missä PSG-rekisteröinti jaotellaan ja luokitellaan joko uneksi tai havahtumiseksi [6].

Koneoppiminen voidaan jakaa karkeasti ohjattuun oppimiseen (engl. *supervised learning*), ohjaamattomaan oppimiseen (engl. *unsupervised learning*) ja vahvistusoppimiseen (engl. *reinforcement learning*) [11]. Useimmat koneoppimisalgoritmit hyödyntävät ohjattua oppimista [9], jossa algoritmille syötettävän koulutusdatan alkioihin liittyy opittavien piirteiden lisäksi ainakin yksi tietty leima (engl. *label*) [12]. Tällöin algoritmin tavoitteena on oppia yhteys data-alkion piirteiden ja leiman välillä, jonka jälkeen algoritmi pystyy liittämään täysin uuteen alkioon jonkin sen jo tuntemista leimoista. Muun muassa kuvien luokittelu ja puheentunnistus ovat ohjattua oppimista [12]. Esimerkiksi edellä mainittu käsin kirjoitettujen numeroiden tunnistaminen on ohjattua oppimista hyödyntävä kuvien luokitteluongelma, jossa data-alkiona toimii kuva käsin kirjoitetusta numerosta ja alkiota vastaavana leimana numeroluokka väliltä 0–9. Havahtumisten tunnistamisessa koulutusdatan alkiona toimii pala PSG-rekisteröintiä, jota vastaava leima on joko havahtuminen tai uni [6].

Ohjaamattomassa oppimisessa koulutusdata ei sisällä leimoja, eli algoritmin tavoitteena on oppia ryhmittelemään data itsenäisesti [11]. Karkeasti sanottuna ohjaamattoman oppimisen tavoitteena on oppia tarkasteltavaa dataa kuvaavan satunnaisvektorin $\mathbf{x}$ todennäköisyysjakauman tiheysfunktio $p(\mathbf{x})$, kun taas ohjatussa oppimisessa algoritmin tavoitteena on ennustaa leimoja vastaava satunnaisvektori $\mathbf{y}$ datasta

$\mathbf{x}$ oppimalla niiden ehdollinen tiheysfunktio $p(\mathbf{y}|\mathbf{x})$ [12]. Ohjatussa oppimisessa on siis tarkoituksena oppia ennustamaan datalle leima tiheysfunktion perusteella, ja ohjaamattomassa oppimisessa tarkoitus on muodostaa datalle tiheysfunktio ilman leiman ennustamista. Ohjaamaton oppiminen on siis selvästi ohjattua oppimista abstraktimpaa, mutta sen etuna on, että sen toteuttaminen ei vaadi juurikaan aikaisempaa tietoa algoritmille syötettävästä datasta [13]. Ohjattu oppiminen sen sijaan vaatii suuren määrän ihmisten ennestään annotoimaa dataa, minkä kerääminen voi viedä jopa vuosia [12]. Vahvistusoppimisessa algoritmia ei puolestaan kouluteta yhden tietyn datajoukon avulla, vaan algoritmi on jatkuvassa vuorovaikutuksessa käyttöympäristönsä kanssa [14]. Tällöin optimaaliset ratkaisut löytää ohjelmistotoimitsija (engl. *software agent*), joka saa jatkuvasti ympäristöltään palautetta toimistaan [13].

2.1 Syväoppiminen ja keinotekoiset neuroverkot

Syväoppiminen on koneoppimisen osa-alue, joka on saanut inspiraationsa aivoissa tapahtuvasta informaation käsittelystä [14]. Syväoppimisalgoritmit mallintavat ilmiöitä hierarkkisina kokonaisuuksina, joiden monimutkaisuus kasvaa edetessä pidemmälle tai syvemmälle algoritmissa, jolloin myös niiden käsitettävyys vaikeutuu [12]. Esimerkiksi kuvantunnistuksessa käytettävä syväoppimisalgoritmi voi ensin etsiä kuvasta pikseleiden välisiä reunoja, seuraavaksi kulmia ja ääriviivoja ja viimeiseksi kokonaisia kappaleita, jonka jälkeen kuvassa oleva ilmiö tai olio pyritään tunnistamaan näiden piirteiden avulla [12]. Syväoppimismallit koostuvat siis useasta kerroksesta, joista jokainen oppii jonkin yksinkertaisen kuvauksen. Syväoppimisalgoritmin voidaan siten ajatella oppivan useasta sisäfunktiosta f_i koostuvan yhdistetyn kuvauksen $f = f_n \circ f_{n-1} \circ ... \circ f_1$, kun taas perinteiset koneoppimisalgoritmit oppivat suoraan funktion f [12].

Syväoppiminen tarjoaa usein tavallista koneoppimista paremman menetelmän monimutkaisiin ongelmiin, sillä useiden kerrosten käyttäminen mahdollistaa monimutkaisten funktioiden oppimisen ja suuridimensioisten data-alkioiden käsittelyn kuormittamatta liikaa laskentatehoa [12]. Kerrosten lukumäärälle ei ole olemassa ylärajaa, joten syväoppimismenetelmät ovat myös hyvin laajennettavissa. Lisäksi samat syväoppimisalgoritmit voivat sopeutua myös muihin tehtäviin kuin mihin ne on alunperin koulutettu [7, 15].

Yleisimpiä syväoppimisalgoritmeja ovat keinotekoiset neuroverkot, jotka ovat saaneet alkunsa tutkimuskäytössä ihmisen keskushermoston malleina [12]. Kuten nisäkkäidenkin keskushermosto, myös keinotekoiset neuroverkot koostuvat neuroneista, jotka viestivät keskenään synapsien välityksellä [16]. Keskushermostossa oppiminen perustuu synapsien fyysisten yhteyksien muuttumiseen, kun taas keinotekoisissa neuroverkoissa se perustuu viestintää säätelevien painokertoimien muuttumiseen [16].

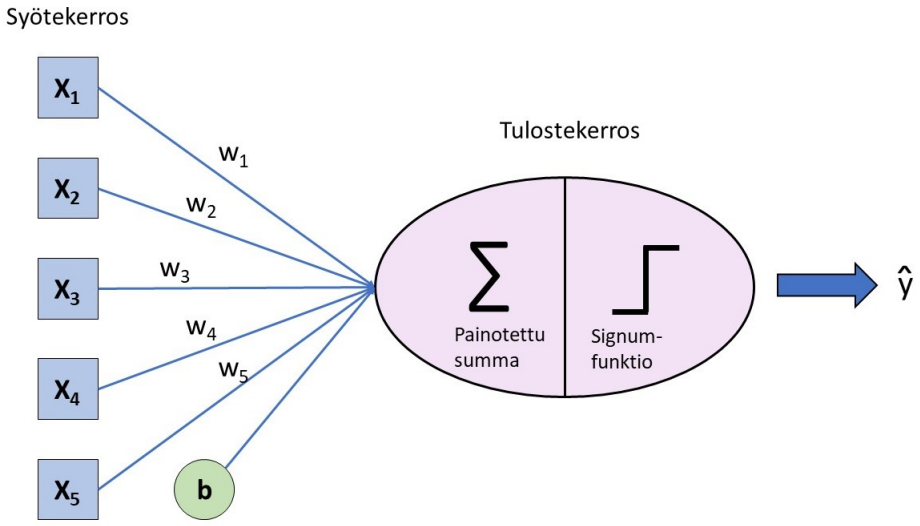
2.1.1 Eteenpäin kytketty neuroverkko

Eteenpäin kytketty neuroverkko (engl. *feedforward neural network*, FNN) on yksinkertaisin syväoppimista hyödyntävä keinotekoinen neuroverkko. Eteenpäin kytketyssä neuroverkossa informaatio kulkee vain yhteen suuntaan muuttuen syötteestä tulosteeksi keinotekoisten neuronien välityksellä [17]. Neuroverkon tavoitteena on oppia funktio $f^*(\mathbf{x}) = y$, joka yhdistää syötteen $\mathbf{x}$ siihen liittyvään leimaan y. Neuroverkko tekee tämän määrittelemällä kuvauksen $f(\mathbf{x}; \boldsymbol{\theta})$, missä vektori $\boldsymbol{\theta}$ sisältää funktion parametrit. Neuroverkko pyrkii optimoimaan nämä parametrit siten, että funktio f arvioi funktiota f^* mahdollisimman hyvin. [12]

Yksinkertaisin eteenpäin kytketty neuroverkko on perseptroni, joka koostuu ainoastaan yhdestä keinotekosesta neuronista [18]. Perseptronin rakenne on esitetty kuvassa 2. Perseptronissa on syötekerros, jonka jokaisella solmulla i on oma painokertoimensa w_i. Perseptroni laskee tulostekerroksessa syötteistä x_i painokertoimilla painotetun summan $\Sigma_{i=1}^{n} w_i x_i$, josta estimaattori $\hat{y}$ lasketaan lopulta signum-funktion avulla binäärisenä tulosteena

$$\hat{y} = \text{sgn}(\Sigma_{i=1}^{n} w_i x_i + b) = \begin{cases} -1, & \text{kun } \Sigma_{i=1}^{n} w_i x_i + b < 0 \\ 1, & \text{kun } \Sigma_{i=1}^{n} w_i x_i + b \geq 0 \end{cases}. \tag{1}$$

Yhtälössä (1) parametri b on estimaattorin $\hat{y}$ harha (engl. *bias*), jolla voidaan parantaa perseptronin suorituskykyä [17].



Kuva 2: Kaavakuva perseptronin rakenteesta. Perseptroni koostuu syötekerroksesta ja tulostekerroksesta, joka laskee harhatermin b ja syötteiden x_i painokertoimilla w_i (i = 1, 2, 3, 4, 5) painotetusta summasta binäärisen tulosteen $\hat{y}$ signum-funktion avulla.

Perseptronin tapauksessa kouluttamisen yhteydessä optimoitavat parametrit ovat syötteiden painokertoimet $\mathbf{w} = (w_1, w_2, ..., w_n)$ ja harha b. Ennen kouluttamista parametrit alustetaan satunnaisesti ja koulutuksen aikana parametreja muutetaan iteratiivisesti siten, että virhe ennustetun luokan $\hat{y}$ ja todellisen luokan välillä minimoituu koulutusdatan joukossa [19]. Parametrien optimointi tehtiin alunperin yrityksen ja erehdyksen kautta [18], mutta ongelma voidaan muotoilla myös modernin koneoppimisen näkökulmasta pienimmän neliösumman sovituksena, jossa parametrien estimaatit $\hat{\boldsymbol{\theta}} = (\hat{\mathbf{w}}, \hat{b})$ lasketaan kaavalla [17]

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{(\mathbf{x},y)\in\mathcal{D}} (y - \hat{y}(\mathbf{x}, \boldsymbol{\theta}))^2, \tag{2}$$

missä $\mathcal{D}$ on koulutusdatan joukko ja $\boldsymbol{\theta} = (\mathbf{w}, b)$ optimoitavat parametrit.

Perseptronin lineaarisesta luonteesta johtuen se sopii ainoastaan yksinkertaisiin luokitteluongelmiin, joissa data on lineaarisesti separoituva [11]. Monet luokitteluongelmat ovat kuitenkin epälineaarisia, joten näiden ratkaiseminen vaatii monimutkaisempia neuroverkkomalleja [17].

Yhdistämällä perseptroneja toisiinsa siten, että edellisten perseptronien tulosteita käytetään seuraavien syötteinä saadaan aikaan verkkomainen malli, jota kutsutaan monikerrokselliseksi perseptroniksi (engl. *multi-layer perceptron*, MLP) [17]. Tällaisia malleja kutsutaan myös täysin kytketyiksi keinotekoisiksi neuroverkoiksi (engl. *fully connected neural network*, FNN), koska niiden jokainen perseptroni on keskushermoston neuronien tavoin yhteydessä toisiinsa [12]. Täysin kytketyn neuroverkon rakenne on esitetty kuvassa 3. Täysin kytketty neuroverkko koostuu syöte- ja tulostekerroksen lisäksi vähintään yhdestä rinnakkaisten perseptronien muodostamasta piilotetusta kerroksesta, jonka tehtävänä on oppia paras tapa arvioida dataa kuvaava funktio f^* [12]. Useasta kerroksesta johtuen täysin kytketty neuroverkko on syväoppimismenetelmä, joka pyrkii esittämään dataa kuvaavan funktion yhdistettynä kuvaksena $f^* = f_n \circ f_{n-1} \circ ... \circ f_1$, missä vektoriarvoiset kuvaukset $f_1, f_2, ..., f_{n-1}$ ovat piilotettujen kerrosten oppimia kuvauksia ja f_n on syötekerroksen oppima kuvaus [12]. Toisaalta piilotetun kerroksen voidaan ajatella koostuvan useasta rinnakkaisesta perseptronista, joista kukin oppii skalaariarvoisen kuvauksen [17]. Tällöin l:nnen piilotetun kerroksen i:nnen neuronin tuloste o_i^l voidaan kirjoittaa muodossa

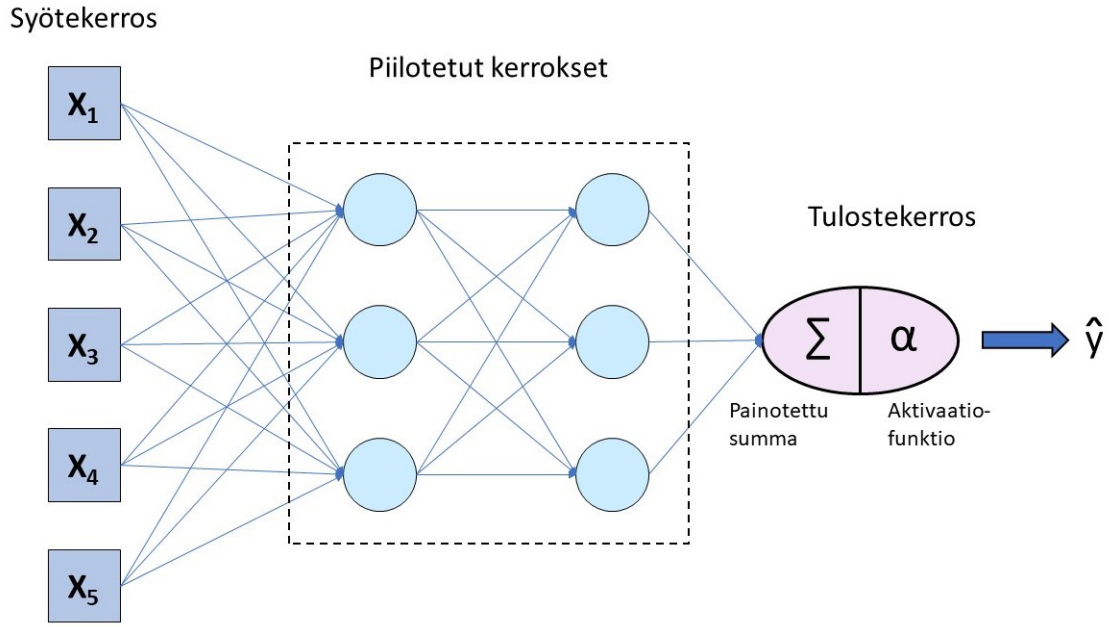
$$o_i^l = \alpha(\sum_j w_{i,j}^l o_j^{l-1} + b_i^l), \tag{3}$$

missä α on aktivaatiofunktio $\mathbb{R} \to \mathbb{R}$, $w_{i,j}^l$ on l:nnen kerroksen i:nnen neuronin ja $(l-1)$:nnen kerroksen j:nnen neuronin välinen painokerroin, b_i^l on l:nnen kerroksen i:nnen neuronin harhatermi ja $o_j^0 = x_j$ on mallille syötettävän datan j:nnes kom-

ponentti [12]. Keräämällä kunkin kerroksen neuronien painokerroinvektorit painokerroinmatriisin W^l riveiksi voidaan yhden kerroksen tuloste kirjoittaa lyhyemmin vektorimuodossa [12]

$$\mathbf{o}^l = \boldsymbol{\alpha}(W^l \mathbf{o}^{l-1} + \mathbf{b}^l), \tag{4}$$

missä vektoriarvoinen aktivaatiofunktio $\boldsymbol{\alpha}$ suorittaa skalaarifunktiota α vastaavan operaation jokaiseen vektorin komponenttiin erikseen.



Kuva 3: Kaavakuva täysin kytketyn neuroverkon rakenteesta. Täysin kytketty neuroverkko koostuu syötekerroksesta, yhdestä tai useammasta rinnakkaisista perseptroneista muodostuvasta piilotetusta kerroksesta ja tulostekerroksesta. Malliin syötetty data $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)$ kulkee piilotettujen kerrosten kautta tulostekerrokselle, joka laskee viimeisen piilotetun kerroksen syötteen painotetusta summasta tulosteen $\hat{y}$ aktivaatiofunktiolla α.

Yhtälössä (1) esitetyn yksittäisen perseptronin tapauksessa aktivaatiofunktiona käytetään signumfunktiota. Monikerroksisessa perseptronissa piilotettujen kerrosten aktivaatiofunktiona käytetään sen sijaan yleensä muita epälineaarisia funktioita, mikä mahdollistaa epälineaaristen funktioiden oppimisen ja siten parantaa neuroverkon suorituskykyä [12]. Yleisimmin piilotettujen kerrosten aktivaatiofunktiona käytetään korjattua lineaarista yksikköä R (engl. *rectified linear unit*) [17], joka määritellään [12]

$$R(x) = max\{0, x\}. \tag{5}$$

Tulostekerroksen aktivaatiofunktion valinta riippuu sen sijaan mallinnettavasta jakaumasta. Binäärisessä luokittelussa käytetään yleensä sigmafunktiota σ [12],

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{6}$$

sillä sen saama arvo on väliltä $(0, 1)$, eli se voidaan siten tulkita toisen luokan todennäköisyydeksi. Useampiluokkaisissa luokitteluongelmissa voidaan käyttää aktivaatiofunktiona vektoriarvoista softmax-funktiota [12],

$$\text{softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\Sigma_j e^{x_j}}. \tag{7}$$

Softmax-yksikön eri komponenttien voidaan tällöin ajatella edustavan eri luokkien todennäköisyyksiä, sillä niiden arvot ovat väliltä $(0, 1)$ ja niiden summa on 1 [12]. Reaaliarvoisen ennusteen tapauksessa, kuten lineaarisessa regressiossa, käytetään tulostekerroksen aktivaatiofunktiona yleensä identtistä kuvausta, jolloin lopullinen tuloste on lineaarinen [12, 17].

Täysin kytketty neuroverkko koulutetaan samalla periaattella kuin yksittäinen perseptroni, eli sen painokertoimet ja harhatermit optimoidaan iteratiivisesti virheen minimoimiseksi koulutusdatan joukossa. Parametrien säätäminen tehdään kerroksittain aloittaen viimeisestä kerroksesta. Täysin kytketyn neuroverkon kouluttamisessa parametreja ei kuitenkaan säädetä satunnaisesti vaan systemaattisesti suuntaan, johon virhettä edustavan funktion arvot pienenevät nopeiten. [12, 17]

Täysin kytketyllä neuroverkolla voidaan teoriassa approksimoida lähes mitä tahansa funktiota mielivaltaisen tarkasti, kunhan neuroneja on verkossa riittävästi [12, 20]. Neuronien määrää voidaan kasvattaa joko lisäämällä niitä olemassaoleviin kerroksiin, jolloin saadaan aikaan matala neuroverkko, tai kasvattamalla piilotettujen kerrosten lukumäärää, jolloin saadaan aikaan syvä neuroverkko. Koska piilotettujen kerrosten lisääminen kasvattaa optimoitavien parametrien lukumäärää vähemmän kuin neuronien lisääminen olemassaoleviin kerroksiin, ovat syvät neuroverkot matalia neuroverkkoja helpommin koulutettavia [12, 17]. Kuitenkin, koska täysin kytketyssä neuroverkossa jokainen neuroni on yhteydessä jokaiseen edellisen ja seuraavan kerroksen neuroniin, lisää niiden lukumäärän kasvattaminen kummalla tahansa tavalla neuronien välisten yhteyksien ja optimoitavien parametrien lukumäärää nopeasti laskentatehoa kuormittavaksi. Tästä syystä täysin kytketty neuroverkko ei sovellu hyvin suurikokoisen datan, kuten kuvien, käsittelyyn [12, 17].

2.1.2 Konvoluutioneuroverkko

Konvoluutioneuroverkot (engl. *convolutional neural network*, CNN) ovat neuroverkkoja, jotka ovat erikoistuneet käsittelemään dataa, jossa on säännöllinen hilamainen rakenne. Tällaista dataa ovat esimerkiksi aikasarjadata, joka koostuu säännöllisin väliajoin tehdyistä mittauksista, sekä kuvat, jotka koostuvat yhtä suurista pikseleistä. Termi konvoluutioneuroverkko juontaa juurensa siitä, että niissä hyödynnetään konvoluutio-operaatiota, joka määritellään matemaattisesti kahden jatkuvan reaalifunktion x ja k välille kaavalla [12]

$$s(t) = (x * k)(t) = \int_{-\infty}^{\infty} x(a)k(t-a)da. \tag{8}$$

Konvoluutioneuroverkot koostuvat tyypillisesti neljästä eri kerrostyypistä: konvoluutiokerroksista, aktivaatiokerroksista, yhdistämiskerroksista (engl. *pooling layer*) ja täysin kytketyistä kerroksista [14]. Konvoluutiokerroksessa datasta muodostetaan konvoluutio-operaatiota hyödyntämällä piirrekarttoja (engl. *feature map*), joiden perusteella data-alkiot voidaan luokitella [12]. Koska koneoppimisessa käsitellään diskreettiä dataa, hyödynnetään konvoluutiokerroksessa diskreettiä konvoluutiota

$$s(i) = (x * k)(i) = \sum_{a=-\infty}^{\infty} x(a)k(i-a), \tag{9}$$

missä funktio x on konvoluutiokerroksen syöte ja funktiota k sanotaan ytimeksi (engl. *kernel*). Yhtälössä (9) nämä ovat määritelty vain diskreeteissä pisteissä i, $i \in \mathbb{N}$. Ydin sisältää konvoluutioneuroverkon optimoitavat parametrit. Konvoluution tuloksena saatava vektori s on ydintä k vastaava piirrekartta. [12]

Äärellisen yksiulotteisen datan $x \in \mathbb{R}^l$ kuten aikasarjojen tapauksessa konvoluutio voidaan kirjoittaa sen vaihdannaisuuden nojalla myös muodossa

$$s(i) = (k * x)(i) = \sum_{m=1}^{N} x(i-m)k(m), \tag{10}$$

missä $i = 1, 2, 3, ..., l$ ja N on ytimen optimoitavien parametrien lukumäärä. Yhtälön (10) perusteella konvoluutio voidaan tulkita operaatioksi, joka ikään kuin skannaa syötteen ytimellä laskien syötteen ja ytimen välisen tulon jokaisessa datapisteessä. Konvoluutio-operaation suorittavaa ydintä siirretään datassa eteenpäin askeltamalla. Konvoluutioneuroverkon kuormitusta voidaan vähentää säätämällä askelluksen pituutta isommaksi, jolloin ydin jättää osan datapisteistä välistä paikkaa vaihtaessaan. Yhtälöstä (10) havaitaan myös, että ydin aloittaa skannauksen osittain alkuperäisen datan ulkopuolelta. Tämä on mahdollista, jos syötettävän datan reunoille lisätään nollia täytteeksi (engl. *padding*). Konvoluutio voidaan laskea myös ilman täytettä

datassa, mutta tällöin laskettu piirrekartta on alkuperäistä dataa pienempi. [12]

Konvoluutiokerroksessa muodostetut piirrekartat siirretään lopulta aktivaatiokerrokseen, jossa lasketaan epälineaarisen aktivaatiofunktion $\boldsymbol{\alpha}$ avulla tuloste

$$\mathbf{o}^l = \boldsymbol{\alpha}(w^l * \mathbf{o}^{l-1} + \mathbf{b}^l), \tag{11}$$

missä yläindeksi l viittaa neuroverkon kerrokseen [14]. Vertaamalla yhtälöä (11) yhtälöön (4) havaitaan, että ainoa ero täysin kytketyn neuroverkon piilotetun kerroksen ja konvoluutioneuroverkon konvoluutiokerroksen aktivaatiossa on konvoluutio-operaation käyttö matriisitulon sijasta.

Yhdistämiskerroksen tehtävänä on vähentää datapisteiden lukumäärää tiivistämällä konvoluutio- ja aktivaatio-operaatioilla laskettuja piirrekarttoja pienemmiksi säilyttäen piirrekarttojen sisältämä informaatio datasta mahdollisimman hyvin. Tavallisia yhdistämisoperaatioita ovat muun muassa jonkin ympäristön suurimman arvon valitseminen (engl. *max pooling*) ja jonkin ympäristön keskiarvon laskeminen (engl. *average pooling*) [12]. Kummankin operaation seurauksena datapisteiden lukumäärä lasketuissa piirrekartoissa pienenee, mikä vähentää neuroverkon kuormitusta [14]. Yhdistämisoperaatioiden jälkeen tiivistetyt piirrekartat syötetään täysin kytkettyyn kerrokseen, jossa data-alkiolle määrätään piirrekartan piirteiden perusteella luokka [14].

2.1.3 Takaisinkytkeytyvä neuroverkko

Takaisinkytkeytyvät neuroverkot (engl. *recurrent neural network*, RNN) ovat neuroverkkoja, jotka ovat erikoistuneet käsittelemään peräkkäistä dataa, kuten aikasarjoja ja puhuttuja tai kirjoitettuja lauseita [12]. Esimerkiksi aikasarjadatassa peräkkäiset datapisteet ovat mitattu ajallisesti lähellä toisiaan, jolloin myös niiden arvot tyypillisesti riippuvat toisistaan [17]. Täysin kytketyillä neuroverkoilla tämä aikariippuvuus jää huomioimatta, sillä ne käsittelevät jokaista niihin syötettyä datapistettä erillisenä, jolloin menetetään datan järjestyksen sisältämä informaatio [17]. Konvoluutioneuroverkot sen sijaan pystyvät mallintamaan lokaaleja ajallisia riippuvuuksia hyödyntämällä pienikokoisia ytimiä ja yhdistämiskerroksia, mutta takaisinkytkeytyvät neuroverkot soveltuvat niitä paremmin pitkän aikävälin riippuvuuksien mallintamiseen [12].

Tässä kappaleessa käsiteltävää aikariippuvaista dataa kutsutaan datatensoriksi, joka koostuu ajallisesti peräkkäisistä data-alkioista, jotka ovat yleisesti vektoreja tai matriiseja. Data-alkion järjestyslukua datatensorissa kutsutaan puolestaan aikapisteeksi. Takaisinkytkeytyvät neuroverkot ottavat datatensorin aikariippuvuuden huomioon käsittelemällä sen yksi alkio kerrallaan tallentaen samalla aiemmista data-

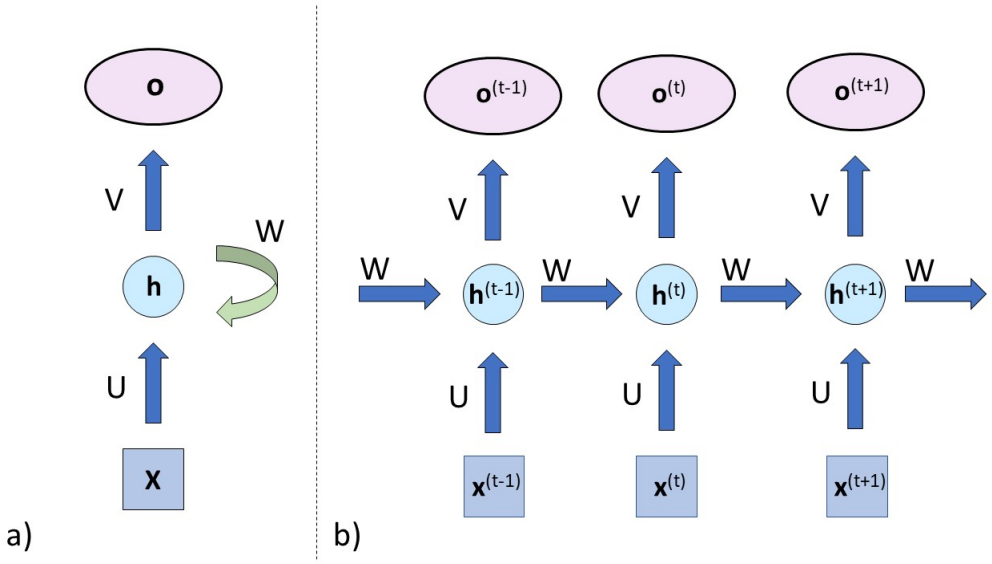
alkioista kerättyä informaatiota piilotetun neuronin tilavektoriin, jota kutsutaan piilotetuksi tilaksi [9]. Aikapistettä t vastaava piilotettu tila $\mathbf{h}^{(t)}$ saadaan kaavalla

$$\mathbf{h}^{(t)} = \boldsymbol{\alpha}(\mathbf{b} + U\mathbf{x}^{(t)} + W\mathbf{h}^{(t-1)}), \tag{12}$$

missä $\boldsymbol{\alpha}$ on vektoriarvoinen aktivaatiofunktio, $\mathbf{x}^{(t)}$ on aikapistettä t vastaava data-alkio, $\mathbf{b}$ on harhatermi ja U ja W ovat painokerroinmatriiseja [12]. Matriisi W välittää piilotettujen tilojen välisen vuorovaikutuksen ja matriisi U välittää syötteen ja piilotetun tilan välisen vuorovaikutuksen. Yhtälöstä (12) nähdään, että kutakin aikapistettä vastaava piilotettu tila riippuu rekursiivisesti kaikista aiemmista piilotetuista tiloista. Aikapistettä t vastaava tuloste $\mathbf{o}^{(t)}$ lasketaan piilotetusta tilasta kaavalla

$$\mathbf{o}^{(t)} = \boldsymbol{\alpha}(\mathbf{c} + V\mathbf{h}^{(t)}), \tag{13}$$

missä $\mathbf{c}$ on tulosteen harha ja V painokerroinmatriisi [12]. Takaisinkytkeytyvän neuroverkon toimintaa on havainnollistettu kuvassa 4. Takaisinkytkeytyvä neuroverkko voidaan esittää myös eteenpäin kytkettynä neuroverkkona, jossa jokaista aikapistettä vastaa oma kerroksensa, joista jokaisessa käytetään samoja painokerroinmatriiseja U, W ja V [17].



Kuva 4: Kaavakuva takaisinkytkeytyvän neuroverkon toiminnasta. Kaaviossa a) on havainnollistettu takaisinkytkeytyvä neuroverkko, joka laskee datatensorista $\mathbf{X}$ tulosteen $\mathbf{o}$ mallintaen aikariippuvuutta piilotetun tilan $\mathbf{h}$ avulla. Vihreä nuoli kuvaa piilotetun tilan iteratiivista laskemista jokaiselle datatensorin data-alkiolle. U, W ja V ovat laskennan aikana optimoitavat painokerroinmatriisit. Kaaviossa b) takaisinkytkeytyvä neuroverkko on kuvattu täysin kytkettynä neuroverkkona, jossa jokainen kerros vastaa yhtä aikapistettä t ja jokaisen kerroksen syötteenä on kyseistä aikapistettä vastaava datatensorin data-alkio ja edellisen kerroksen piilotettu tila.

Jos käsiteltävät datatensorit ovat suuria, käy tavallisen takaisinkytkeytyvän neuroverkon kouluttaminen haastavaksi [17]. Lisäksi tavallisella takaisinkytkeytyvällä neuroverkolla on hankala mallintaa pitkien aikavälien riippuvuuksia [9]. Näiden ongelmien ratkaisemiseksi takaisinkytkeytyvistä neuroverkoista on kehitetty paranneltuja versioita, joista yleisin on LSTM-neuroverkko (engl. *long short-term memory*). [12]

2.1.4 LSTM-neuroverkko

LSTM-neuroverkoissa hyödynnetään LSTM-soluja, joilla on piilotetun tilan $\mathbf{h}$ lisäksi solutila $\mathbf{s}$ (engl. *cell state*), joka mahdollistaa informaation keräämisen pitkällä aikavälillä [17]. LSTM-solun toimintaa on havainnollistettu kuvassa 5. Solutilan ja piilotetun tilan aktivaatiota säädellään kolmen funktion avulla, joita kutsutaan syöteportiksi (engl. *input gate*), unohdusportiksi (engl. *forget gate*) ja tulosteportiksi (engl. *output gate*) [12]. Kaikkien porttien aikapistettä t vastaavat arvot $\mathbf{g}^{(t)}$ lasketaan kaavalla [12]

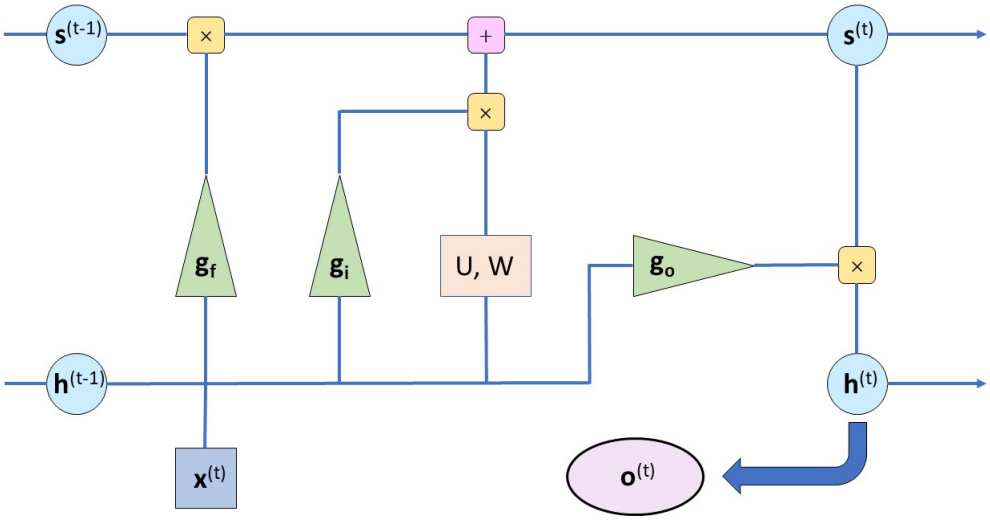
$$\mathbf{g}^{(t)} = \boldsymbol{\sigma}(\mathbf{b}^{\mathrm{g}} + U^{\mathrm{g}}\mathbf{x}^{(t)} + W^{\mathrm{g}}\mathbf{h}^{(t-1)}), \tag{14}$$

missä $\boldsymbol{\sigma}$ on vektoriarvoinen sigmafunktio, jonka komponenttifunktiot ovat yhtälön (6) mukaisia. Edelleen $\mathbf{b}^{\mathrm{g}}$ on portin harhatermi, U^{g} syötteen $\mathbf{x}^{(t)}$ vaikutuksen välittävä painokerroinmatriisi ja W^{g} piilotetun tilan $\mathbf{h}^{(t-1)}$ vaikutuksen välittävä painokerroinmatriisi. Syöteportille, unohdusportille ja tulosteportille on kullekin omat harhatermisä ja painokerroinmatriisinsa, joiden alkiot ovat mallin optimoitavia parametreja [12]. Aikapistettä t vastaavan solutilan arvo lasketaan syöteportin $\mathbf{g}_{\mathrm{i}}$ ja unohdusportin $\mathbf{g}_{\mathrm{f}}$ avulla kaavalla [12]

$$\mathbf{s}^{(t)} = \mathbf{g}_{\mathrm{f}}^{(t)} \odot \mathbf{s}^{(t-1)} + \mathbf{g}_{\mathrm{i}}^{(t)} \odot \boldsymbol{\sigma}(\mathbf{b} + U\mathbf{x}^{(t)} + W\mathbf{h}^{(t-1)}), \tag{15}$$

missä $\mathbf{b}$ on LSTM-solun harha sekä U ja W ovat syötteen ja piilotetun tilan painokerroinmatriisit. Vektorien välinen operaatio $\odot$ tarkoittaa pisteittäistä kertolaskua.

Kuten yhtälöstä (15) havaitaan, syöteportin tehtävänä on säädellä LSTM-solun solutilan muistiin lisättävää informaatiota syötteestä ja piilotetusta tilasta. Signumfunktion arvojoukosta johtuen syöteportin komponentit saavat arvoja väliltä $(0, 1)$. Kun syöteportin arvo on lähellä arvoa 1, syötteen ja piilotetun tilan informaatio lisätään solutilan muistiin ja kun arvo on lähellä nollaa, informaatio jätetään huomiotta. Unohdusportin tehtävänä puolestaan on säädellä sitä, kuinka pitkään informaatiota säilytetään LSTM-solun pitkäkestoisessa muistissa. Kun unohdusportin arvo on lähellää nollaa, solutilaan kertynyt informaatio unohdetaan, jonka jälkeen sinne voidaan kerätä uutta informaatiota. [12]



Kuva 5: Kaavakuva LSTM-solun toiminnasta. LSTM-solulla on kaksi erillistä tilaa: piilotettu tila $\mathbf{h}$ ja solutila $\mathbf{s}$, joiden arvoja päivitetään kussakin aikapisteessä t syöteportin $\mathbf{g}_\mathrm{i}$, unohdusportin $\mathbf{g}_\mathrm{f}$ ja tulosteportin $\mathbf{g}_\mathrm{o}$ välityksellä. Kutakin aikapistettä vastaava data-alkio $\mathbf{x}^{(t)}$ sekä edellistä aikapistettä vastaava piilotettu tila $\mathbf{h}^{(t-1)}$ vaikuttavat sekä suoraan porttien arvoihin että solutilan arvoon painokerroinmatriisien U ja W välityksellä. Päivitetystä piilotetusta tilasta $\mathbf{h}^{(t)}$ lasketaan tuloste $\mathbf{o}^{(t)}$. Keltaiset laatikot kuvaavat pisteittäistä kertolaskuoperaatiota ja violetti laatikko yhteenlaskua.

LSTM-solun piilotetun tilan arvo lasketaan kussakin aikapisteessä t solutilasta $\mathbf{s}^{(t)}$ tulosteportin $\mathbf{g}_\mathrm{o}$ avulla kaavalla

$$\mathbf{h}^{(t)} = \boldsymbol{\alpha}(\mathbf{s}^{(t)}) \odot \mathbf{g}_\mathrm{o}^{(t)}, \tag{16}$$

missä aktivaatiofunktiona $\boldsymbol{\alpha}$ käytetään yleensä hyperbolista tangenttia [12]. Piilotetusta tilasta lasketaan edelleen tuloste yhtälön (13) mukaisesti. Tulosteportin tehtävänä on siis säädellä sitä, kuinka paljon mitäkin solutilan komponentteihin säilöttyä informaatiota hyödynnetään piilotetun tilan ja siten tulosteen laskemisessa kussakin aikapisteessä. [17]

2.1.5 Keinotekoisten neuroverkkojen kouluttaminen

Keinotekoisten neuroverkkojen kouluttamisessa käytettävä data jaetaan kolmeen erilliseen osajoukkoon: koulutusjoukkoon, validointijoukkoon ja testijoukkoon. Neuroverkon parametrit säädetään koulutus- ja validointijoukkojen avulla. Testijoukkoa puolestaan ei hyödynnetä ollenkaan parametrien optimoinnissa, vaan sen avulla testataan valmiin mallin yleistyvyyttä eli sitä, kuinka hyvin malli kykenee luokittelemaan sille tuntemattomat data-alkiot. [12]

Neuroverkon kouluttamisen tavoitteena on, että virhe data-alkioille ennustettujen leimojen ja niiden todellisten leimojen välillä olisi mahdollisimman pieni. Kouluttamisen aikana tämä virhe minimoidaan koulutujoukossa säätämällä neuroverkon parametreja eli painokerroinmatriisien ja harhatermien alkioita. [12] Virheelle on olemassa useita erilaisia matemaattisia mittareita ja niitä kutsutaan yleisesti häviöfunktioiksi (engl. *loss function*). Neuroverkon kouluttaminen voidaan siten nähdä optimointiongelmana, jossa tavoitteena on minimoida häviöfunktion L arvo koulutusjoukossa $\mathcal{D}$. [12] Eräitä yksinkertaisia häviöfunktioita regressio-ongelmissa ovat keskineliövirhe

$$L = \sum_{(\mathbf{x},y)\in\mathcal{D}} (y - \hat{y}(\mathbf{x}, \boldsymbol{\theta}))^2 \tag{17}$$

ja absoluuttinen keskivirhe

$$L = \sum_{(\mathbf{x},y)\in\mathcal{D}} |y - \hat{y}(\mathbf{x}, \boldsymbol{\theta})|. \tag{18}$$

Yhtälöissä (17) ja (18) $\mathbf{x}$ on koulutusjoukon data-alkio, y siihen liittyvä leima, $\hat{y}$ neuroverkon data-alkiolle ennustama leima ja $\boldsymbol{\theta}$ optimoitavat parametrit [12]. Yhtälön (7) mukaista softmax-tulostetta hyödyntävien luokitteluongelmien yhteydessä neuroverkkojen kouluttamisessa käytetään kuitenkin yleensä häviöfunktiona ristientropiaa (engl. *cross-entropy*), joka on muotoa

$$L = -\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} \mathbf{y} \cdot \log(\hat{\mathbf{y}}(x, \boldsymbol{\theta})), \tag{19}$$

missä $\mathbf{y}$ on eri luokkien todelliset todennäköisyydet ja $\hat{\mathbf{y}}$ neuroverkon ennustamat luokkien todennäköisyydet sisältävä vektori [12, 17]. Operaatio $\cdot$ tarkoittaa vektorien pistetuloa. Binäärisessä luokitteluongelmassa, kuten havahtumisten tunnistamisessa ristientropia saa tällöin muodon

$$L = -\sum_{i=1}^{N} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i), \tag{20}$$

missä N on data-alkioiden lukumäärä, y_i havahtumisen todennäköisyys, joka on joko 0 tai 1, ja $\hat{y}_i$ havahtumiselle ennustettu todennäköisyys [2, 7].

Toisin kuin perseptronien kouluttamisessa [18], neuroverkkojen parametreja ei säädetä satunnaisesti, vaan gradienttilaskumenetelmällä (engl. *gradient descent*) iteratiivisesti aina häviöfunktion negatiivisen gradientin $-\nabla_{\boldsymbol{\theta}} L$ suuntaan, jossa häviöfunktion arvo pienenee nopeiten. Neuroverkon oppimisnopeudeksi kutsuttava parametri määrää, kuinka paljon neuroverkon parametreja säädetään kerrallaan. [12]

Häviöfunktion gradientti lasketaan optimoitavien parametrien suhteen vastavirta-

algoritmilla (engl. *backpropagation*). Vastavirta-algoritmi määrittää gradientin ensin neuroverkon viimeisessä kerroksessa, jonka jälkeen lasketaan sen riippuvuus neuroverkon aiempien kerrosten parametreista derivoinnin ketjusääntöä hyödyntämällä edeten viimeisestä kerroksesta ensimmäiseen. [17]

Kun koulutusdatan alkioiden lukumäärä on suuri, käy häviöfunktion gradientin määrittäminen vastavirta-algoritmilla kuitenkin kuormittavaksi, sillä se on määritettävä erikseen jokaiselle datapisteelle. Tästä syystä neuroverkkojen optimoinnissa hyödynnetään yleensä stokastista gradienttilaskua, jossa koulutusdatasta valitaan satunnaisesti pienempi osajoukko gradientin määrittämistä varten. [12]

Koulutusdata jaetaan usein muistin säästämiseksi pienempiin osajoukkoihin eli eriin (engl. *batch*) [21]. Kun kaikki erät eli koko koulutusdata on syötetty neuroverkkoon, sanotaan että neuroverkko on käynyt läpi yhden koulutusjakson (engl. *epoch*) [22]. Kouluttamisen aikana neuroverkko käy läpi useita koulutusjaksoja virheen minimoimiseksi koulutusdatan joukossa [12].

Neuroverkon kouluttamisessa tulee kuitenkin välttää ylisovittamista, jossa neuroverkko oppii ulkoa koulutusjoukon alkioiden leimat, mutta ei opi alkioiden ja leimojen välisiä riippuvuuksia [11, 21]. Ylisovittamisen seurauksena häviöfunktion arvo minimoituu koulutusjoukossa, mutta testijoukossa se on kuitenkin suuri. Ylisovittamisen välttämiseksi neuroverkon virhettä voidaan testata koulutusjaksojen välissä koulutusjoukosta erillisellä validointijoukolla, jonka osuus on yleensä 20 % neuroverkon kouluttamiseen käytettävästä datasta. Kouluttaminen voidaan esimerkiksi keskeyttää, jos virhe validointijoukossa lähtee kasvamaan, vaikka koulutusjoukossa se vielä pienenisi. Ylisovittamista voidaan vähentää myös säätämällä neuroverkon hyperparametreja, kuten sen oppimisnopeutta ja kerrosten lukumäärää, sillä nämä vaikuttavat neuroverkon kykyyn mallintaa erilaisia funktioita. Lisäksi eräs laskennallisesti edullinen ja tehokas keino ylisovittamisen välttämiseen on lisätä neuroverkon kerrosten väliin häviämiskerroksia (engl. *dropout layer*), joissa satunnainen osa kerroksen neuroneista poistetaan asettamalla niitä vastaavat parametrit nollaksi. Tämä ehkäisee ylisovittamista, koska se estää neuroverkkoa oppimasta liiallisia riippuvuuksia neuronien välillä. [12]

Jos neuroverkon kouluttamiseen käytettävissä olevan kokonaisdatan määrä on pieni, voi testijoukko jäädä liian pieneksi, jotta neuroverkon suorituskykyä voitaisiin arvioida luotettavasti. Neuroverkon testaamiseen käytettävissä olevaa dataa voidaan tällöin lisätä ristivalidoinnilla (engl. *cross-validation*). K-kertainen ristivalidointi tarkoittaa, että kokonaisdata jaetaan k:hon erilliseen osajoukkoon, joista kutakin käytetään vuorollaan testijoukkona muiden osajoukkojen muodostaessa koulutus- ja validointijoukot. Neuroverkon suorituskyvylle saadaan tällöin arvio keskiarvona k:n eri koulutuskerran testijoukoissa saavutetuista suorituskyvyistä. [12]

2.1.6 Keinotekoisten neuroverkkojen suorituskyvyn mittaaminen

Neuroverkkojen suorituskyvylle on olemassa erilaisia mittareita, joiden avulla eri neuroverkkojen suorituskykyjä voidaan verrata toisiinsa [12]. Binäärisissä eli kaksiluokkaisissa luokitteluongelmissa käytettäviä suorituskykymittareita ovat muun muassa tarkkuus p (engl. *precision*) ja herkkyys r (engl. *recall*), jotka määritellään

$$p = \frac{TP}{TP + FP} \quad \text{ja} \quad r = \frac{TP}{TP + FN}, \tag{21}$$

missä TP on todelliset positiiviset, FP väärät positiiviset, ja FN väärät negatiiviset havainnot [14]. Havahtumisten tunnistamisessa TP vastaa oikein tunnistettuja havahtumisia, FP väärin tunnistettuja havahtumisia ja FN havahtumisia, jotka neuroverkolta on jäänyt tunnistamatta. Koska neuroverkon tulosteena saadaan luokkien todennäköisyydet, tarvitaan neuroverkon ennustaman leiman määräämiseksi todennäköisyydelle kynnysarvo [2, 7]. Kynnysarvoa muuttamalla saadaan neuroverkolle useita (p, r)-pareja, jotka kuvataan usein PR-käyrän avulla, jossa esitetään mallin tarkkus y-akselilla ja herkkyys x-akselilla [12]. Neuroverkon suorituskyvyn mittarina voidaan tällöin käyttää myös PR-käyrän alle jäävää pinta-alaa, jota merkitään AUPRC [14]. Usein neuroverkon suorituskykyä luokitteluongelmissa kuvataan myös tarkkuuden ja herkkyyden yhdistävällä F1-pisteytyksellä, joka määritellään [14]

$$\mathrm{F1} = \frac{2pr}{p + r}. \tag{22}$$

Yhtälöistä (21) ja (22) havaitaan, että kaikki edellä esitetyt suorituskyvyn mittarit saavat arvoja väliltä $[0, 1]$ ja suuremmat arvot vastaavat parempaa suorituskykyä. Suorituskyvyn mittarina voidaan käyttää myös vääriä positiivisia korostavaa FPR:ää (engl. *false positive rate*), joka määritellään [14]

$$\mathrm{FPR} = \frac{FP}{FP + TN}. \tag{23}$$

Esittämällä eri kynnysarvoilla saavutetut mallin herkkyydet y-akselilla ja FPR:t x-akselilla saadaan ROC-käyrä (engl. *receiver operating characteristic curve*). Tällöin neuroverkon suorituskyvyn mittarina voidaan käyttää myös ROC-käyrän alle jäävää pinta-alaa, jota merkitään AUROC. [6]

3 Havahtumisten tunnistaminen keinotekoisilla neuroverkoilla

Unenaikaisia havahtumisia on onnistuttu tunnistamaan tehokkaasti erilaisilla neuroverkkomalleilla [2, 7, 10, 15, 23]. Yhteenveto tässä kirjallisuuskatsauksessa käsiteltävistä malleista on esitetty Taulukossa 1.

Taulukko 1: Yhteenveto tutkielmassa käsiteltävistä havahtumisten tunnistamiseen kehitetyistä neuroverkkomalleista sekä niillä saavutetut suorituskyvyt testijoukoissa.

Viite	**Datajoukko**	**PSG-signaalit**	**Neuroverkon rakenne**	**Suorituskyky testijoukossa**
Miller ym. (2018) [23]	PhysioNet 2018 (n = 1983)	EEG, EOG, EMG, EKG, hengitys$^{(*)}$ & SaO2	16 CL + 1 FCL	AUROC = 0,855 AUPRC = 0,369
Zabihi ym. (2019) [15]	PhysioNet 2018 (n = 1983)	EEG, EOG, EMG, hengitys$^{(*)}$ & SaO2	19 CL + 2 FCL	AUROC = 0,84 AUPRC = 0,31
Li & Guan (2021) [7]	PhysioNet 2018 (n = 1983)	EEG, EOG, EMG, EKG, hengitys$^{(*)}$ & SaO2	35 CL	AUROC = 0,927 AUPRC = 0,55
Jia ym. (2020) [10]	Beijing Tongren Hospital (n = 323)	EEG, EOG & EMG	36 CCL + 1 FCL	F1 = 0,863
Brink-Kjaer ym. (2020) [2]	MrOS (n = 2888), CFS (n = 726), WSC$^{(**)}$ (n = 269) & SSC$^{(**)}$ (n = 30)	EEG, EOG & EMG	13 CL + 1 biLSTML + 3 FCL	F1 = 0,76 AUPRC = 0,82

PSG = polysomnografia, n = koulutuksessa ja testauksessa käytettyjen PSG-rekisteröintien lukumäärä, EEG = aivosähkökäyrä, EOG = elektro-okulografia, EMG = elektromyografia, EKG = elektrokardiografia, SaO2 = valtimoveren happisaturaatio, MrOS = MrOS sleep study, CFS = Cleveland Family Study, WSC = Wisconsin Sleep Cohort, SSC = Stanford Sleep Cohort, CL = konvoluutiokerros, FCL = täysin kytketty kerros, biLSTML = kaksisuuntainen LSTM-kerros, AUROC = ROC-käyrän alle jäävä pinta-ala, AUPRC = PR-käyrän alle jäävä pinta-ala.

$^{(*)}$ Koostuu hengityssignaalista ja hengitysilmavirtauksesta.

$^{(**)}$ Dataa käytettiin vain neuroverkon testaamiseen.

Taulukon 1 malleista kolmen [7, 15, 23] kouluttamiseen ja testamiseen käytettiin PhysioNet 2018 -datajoukkoa, joka kerättiin vuonna 2018 järjestettyyn PhysioNet/Computing in Cardiology -kilpailuun, jossa kilpailijoiden tehtävänä oli kehittää automaattisia menetelmiä unenaikaisten havahtumisten tunnistamiseen [24]. Datajoukko koostui Massachusettsin yleissairaalan unilaboratoriossa mitatusta 1983 PSG-rekisteröinnistä, joka jaettiin 994 rekisteröinnin koulutusjoukkoon ja 989 rekisteröinnin testijoukkoon. Rekisteröinnit sisälsivät dataa kuudesta EEG-kanavasta, yhdestä EOG-kanavasta, leuan EMG-kanavasta, hegitysliikkeestä, hengityksen ilmavirtauksesta, yhdestä EKG-kanavasta sekä pulssioksimetrilla mitatusta valtimoveren happisaturaatiosta. PSG-rekisteröinneissä esiintyvien havahtumisten annotointiin osallistui yhteensä seitsemän ammattilaista niin, että jokaisen PSG-rekisteröinnin annotoi yksi ammattilainen. [24]

Miller ym. [23] kehittivät havahtumisten tunnistamiseen konvoluutioneuroverkkomallin, jonka kouluttamisessa ja testaamisessa hyödynnettiin PhysioNet 2018 -datajoukkoa, jonka koulutusjoukosta erotettiin 199 PSG-rekisteröintiä mallin vali-

dointiin. Malli koostui kahdeksasta konvoluutiokerroksesta ja kahdeksasta niitä vastaavasta piirrekarttoja laajentavasta dekonvoluutiokerroksesta. Toisiaan vastaavien konvoluutio- ja dekonvoluutiokerrosten välille oli myös muodostettu oikoteitä (engl. *skip connection*), joita pitkin datan oli mahdollista ohittaa syvemmällä olevat kerrokset. Jokaista konvoluutiokerrosta seurasi häviämiskerros ja eränormalisointi (engl. *batch normalization*), jossa kerroksen tulosteet normitetaan samalle jakaumalle [12]. Malli saavutti testijoukossa AUROC-arvon 0,855 ja AUPRC-arvon 0,369. [23]

Myös Zabihi ym. [15] hyödynsivät PhysioNet 2018 -datajoukkoa kehittäessään viisi rakenteeltaan erilaista konvoluutioneuroverkkomallia havahtumisten tunnistamiseen. Parhaan suorituskyvyn saavuttanut malli koostui tavallisten konvoluutiokerrosten lisäksi kausaalisesta konvoluutikerroksesta, joka pyrkii huomioimaan datan sisältämiä syy-seuraussuhteita, ja yhdeksästä laajennetusta konvoluutiokerroksesta, joiden ytimet muodostavat piirrekarttansa resoluution kustannuksella laajemmalta alueelta ja harvemmalla näytteistystaajuudella tavalliseen konvoluutioytimeen verrattuna [15, 25]. Mallissa hyödynnettin myös Millerin ym. [23] mallin tapaan häviämiskerroksia, kerrosten välisiä oikoteitä sekä eränormalisointia. Malli koulutettiin 3-kertaisella ristivalidoinnilla 20 koulutusjakson aikana. Zabihin ym. malli saavutti PhysioNet 2018 -testijoukossa AUROC-arvon 0,84 ja AUPRC-arvon 0,31. [15]

Viimeinen PhysioNet 2018 -datajoukkoa hyödyntänyt havahtumisia tunnistava neuroverkkomalli on Lin ja Guanin kehittämä DeepSleep-konvoluutioneuroverkko [7]. Tämä malli pohjautui U-Net-konvoluutioneuroverkkoon [26], joka koostui Millerin ym. [23] mallin tapaan piirrekarttoja tiivistävästä maksimiyhdistämistä hyödyntävästä enkooderista ja niitä purkavasta dekooderista. Malli sisälsi yhteensä 35 konvoluutiokerrosta, joista jokaisen jälkeen suoritettiin eränormalisaatio. Mallissa hyödynnettiin myös enkooderin ja dekooderin välisiä oikoteitä. Li ja Guan tutkivat myös datan resoluution vaikutusta neuroverkon suorituskykyyn kehittämällä useamman eri neuroverkon, joista yksi käsitteli täysiresoluutioista, yksi puoliresoluutioista ja yksi kahdeksasosaresoluutioista dataa. Parhaan suorituskyvyn saavuttanut malli sai syötteenään kaikki eri resoluutioiset datat. Li ja Guan lisäsivät myös malliinsa algoritmin, joka vaihtoi satunnaisesti leuan EMG- ja EEG-kanvavien signaaleja kouluttamisen aikana. DeepSleep-neuroverkkomalli saavutti PhysioNet 2018 -testidatassa AUROC-arvon 0,927 ja AUPRC-arvon 0,550. [7]

Jia ym. [10] kehittivät havahtumisten tunnistamiseen EEG-, EOG-, ja leuan EMG-signaaleista konvoluutioneuroverkkomallin, joka koostui yhteensä 36 konvoluutiokerroksesta ja yhdestä täysin kytketystä kerroksesta. Konvoluutiokerroksissa hyödynnettiin myös eränormalisointia ja yhdistämisoperaatioita. Heidän mallinsa sai syötteenä viisi peräkkäistä kuuden sekunnin mittaista PSG-rekisteröinnin pätkää, jotka luokiteltiin samaan luokkaan. He kouluttivat ja testasivat mallinsa Pekingin Tongrenin

sairaalassa mitattujen 323 PSG-rekisteröinnin avulla 3-kertaisella ristivalidoinnilla. Malli saavutti testijoukossa F1-arvon 0,863. [10]

Brink-Kjaer ym. [2] kehittivät havahtumisen tunnistamiseen neuroverkkomallin, joka hyödynsi muista taulukon 1 malleista poiketen myös kaksisuuntaista LSTM-neuroverkkoa, jossa luvussa 2.1.4 kuvattu LSTM-verkko on ajettu kahteen kertaan: alusta loppuun ja lopusta alkuun, jolloin saadaan hyödynnettyä sekä mennyttä että tulevaa informaatiota PSG-signaaleista. Heidän mallinsa koostui kokonaisuudessaan 13 konvoluutiokerroksesta, joita seurasi keskiarvoyhdistämiskerros, sekä yhdestä kaksisuuntaisesta LSTM-kerroksesta ja kolmesta täysin kytketystä kerroksesta. Jokaista konvoluutiokerrosta ja täysin kytkettyä kerrosta seurasi myös eränormalisointi. Mallin kouluttamisessa käytettiin 2308 annotoitua PSG-rekisteröintiä iäkkäiden miesten MrOS unitutkimuskohortista (engl. *MrOS sleep study*) ja 581 PSG-rekisteröintiä Clevelandin perhetutkimuskohortista (engl. *Cleveland family study*, CFS). Mallin testaamisessa käytettiin 580 PSG-rekisteröintiä MrOS-kohortista, 145 rekisteröintiä CFS-kohortista, 269 rekisteröintiä Wisconsinin unikohortista (engl. *Wisconsin Sleep Cohort*, WSC) ja 30 rekisteröintiä Stanfordin unikohortista (engl. *Stanford Sleep Cohort*, SSC). Malli saavutti testijoukossa havahtumisten tunnistamisessa AUPRC-arvon 0,82 ja F1-arvon 0,68, kun havahtumisen todennäköisyyden kynnysarvo oli 0,225. [2]

Jian ym. [10] tutkimusta lukuunottamatta kaikki edellä esitellyt mallit koulutettiin Adam-optimointialgoritmia käyttämällä [27], joka on oppimisnopeutta koulutuksen aikana säätelevä stokastisen gradienttilaskun laajennus [2, 7, 15, 23]. Jia ym. [10] eivät maininneet mallin kouluttamiseen käytettyä optimointialgoritmia eivätkä häviöfunktiota. Muiden mallien häviöfunktiona käytettiin ristientropiaa tai sen johdannaista [2, 7, 15, 23]. Miller ym. [23], Zabihi ym. [15] sekä Brink-Kjaer ym. [2] käyttivät tulostekerroksessa softmax-aktivaatiota. Li ja Guan [7] puolestaan käyttivät tulostekerroksen aktivaatiofunktiona sigmafunktiota. Jia ym. [10] eivät raportoineet tulostekerroksessa käyttämäänsä aktivaatiofunktiota.

4 Pohdinta

Kaikki taulukossa 1 esitetyt mallit onnistuivat tunnistamaan havahtumisia testijoukoistaan luotettavasti. Mallien välisissä suorituskyvyissä on kuitenkin myös eroavaisuuksia, ja suorituskykyihin vaikuttavia tekijöitä tulee myös pohtia kriittisesti. Millerin ym. [23], Zabihin ym. [15] sekä Lin ja Guanin [7] malleja voidaan suoraan verrata toisiinsa, sillä niiden kouluttamiseen ja testaamiseen on käytetty samoja PhysioNet 2018 -datajoukon osajoukkoja.

Millerin [23] ja Zabihin [15] tutkimusryhmien mallien suorituskyvyt olivat hyvin lähellä toisiaan. Millerin ja kollegoiden malli [23] saavutti testijoukossa hieman suuremmat AUROC- ja AUPRC-arvot, mikä kertoo heidän mallinsa tunnistaneen havahtumiset keskimäärin paremmin kuin Zabihin ym. [15] malli. Osaltaan tätä eroa voi selittää se, että Miller ym. [23] käyttivät neuroverkon syötteenä kaikkia 13 PhysioNet 2018 -datajoukon PSG-kanavaa, kun taas Zabihi ym. [15] hyödynsivät mallissaan vain 11 PSG-kanavan informaatiota. Millerin ym. [23] mallilla oli täten käytettävissään enemmän informaatiota havahtumisten tunnistamisessa. Toisaalta suorituskykyjen välistä eroa voi selittää myös hyvin erilaiset neuroverkkojen rakenteet. Vaikka sekä Miller ym. [23] että Zabihi ym. [15] hyödynsivät malleissaan sekä konvoluutiokerroksia että täysin kytkettyjä kerroksia, oli niiden toimintaperiaate melko erilainen: Millerin ym. [23] malli perustui alunperin kuvansegmentoitiin kehitettyyn enkooderista ja dekooderista koostuvaan U-Net-neuroverkkoon [26], kun taas Zabihin ym. [15] malli perustui alunperin puheen tuottamiseen kehitettyyn kausaalista ja laajentavaa konvoluutiota hyödyntävään WaveNet-neuroverkkoon [25]. Erilaisista konvoluutiokerroksista johtuen heidän mallinsa saattavat löytää keskenään erilaisia piirteitä unenaikaisesta PSG-datasta havahtumisten luokittelemiseksi.

Lin ja Guanin [7] malli sen sijaan saavutti Millerin [23] ja Zabihin [15] tutkimusryhmien malleja huomattavasti paremman suorituskyvyn AUROC- ja AUPRC-arvoilla mitattuna. Heidän mallinsa oli rakenteeltaan ja toimintaperiaatteeltaan hyvin lähellä Millerin ym. [23] mallia: molemmat mallit saivat syötteenään kaikki 13 PSG-signaalia ja molempien rakenne muistutti U-Net-neuroverkkoa. Li ja Guan [7] eivät mainitse artikkelissaan täysin kytkettyjä kerroksia, mutta alkuperäinen U-Net sisältää myös kaksi täysin kytkettyä kerrosta konvoluutiokerrosten jälkeen [26], jotka oletettavasti suorittavat havahtumisten luokittelun myös Lin ja Guanin mallissa. Eräs syy Lin ja Guanin [7] mallin paremmalle suorituskyvylle voi olla huomattavasti suurempi konvoluutiokerrosten lukumäärä, sillä Lin ja Guanin mallissa konvoluutiokerroksia on yli kaksinkertainen määrä Millerin ym. [23] malliin verrattuna. Lin ja Guanin [7] mallin suurempi konvoluutiokerrosten lukumäärä saattaa mahdollistaa useampien ja monimutkaisempien piirteiden löytämisen PSG-datasta, mikä voi helpottaa havahtumisten tunnistamista täysin kytketyissä kerroksissa. Lisäksi Lin ja Guanin [7] mallin parem-

paa suorituskykyä voi selittää heidän hyödyntämänsä koulutusdatan vahvistusstrategia, jossa he vaihtoivat satunnaisesti keskenään sekä eri EEG-kanavien signaaleja että eri leuan EMG-kanavien signaaleja mallin kouluttamisen aikana. Tämän tekniikan ansiosta Lin ja Guanin [7] mallilla oli mahdollisesti monipuolisempi koulutusaineisto käytettävissään, mikä on saattanut tehostaa mallin havahtumisten tunnistamista sille tuntemattomassa testidatassa.

Jian ym. [10] sekä Brink-Kjaerin ym. [2] mallien suorituskykyä on haastavaa verrata suoraan muihin taulukossa 1 esitettyihin malleihin, sillä heidän mallinsa hyödynsivät sekä kouluttamisessa että testauksessa eri datajoukkoja. Tämä vaikeuttaa mallien vertailua, sillä neuroverkon rakenteen lisäksi myös käytetyllä koulutusjoukolla on suuri vaikutus neuroverkon suorituskykyyn [12]. Lisäksi raportoidut suorituskyvyn mittarit kertovat mallin suorituskyvystä ainoastaan testijoukossa, eli jos testijoukko on pieni tai huonosti globaalia populaatiota edustava, suorituskyvyn arviot voivat olla epäluotettavia.

Erityisesti Jian ym. [10] mallin suorituskykyä on hankala arvioida luotettavasti. He käyttivät neuroverkon kouluttamiseen ja testaamiseen datajoukkoa, joka ei ole julkisesti saatavilla, joten heidän tutkimuksensa ei ole toistettavissa. He eivät myöskään kertoneet tarkemmin datajoukon jaottelusta koulutus-, validointi- ja testijoukkoihin. Mallin saavuttama suuri F1-arvo voi siten olla seurausta esimerkiksi ylisovittamisesta, eikä se tällöin edusta mallin suorituskykyä globaalilla tasolla.

Brink-Kjaer ym. [2] hyödynsivät mallinsa kouluttamisessa ja testaamisessa sekä PSG-rekisteröintien että tutkimuskohorttien lukumääriltään suurinta datajoukkoa. Tämä lisää heidän mallinsa suorituskyvyn arvioinnin luotettavuutta, sillä PSG-rekisteröinnit ovat peräisin useammalta eri potilaalta ja havahtumisten annotointiin on osallistunut useampi eri asiantuntija. AUPRC-arvolla mitattuna Brink-Kjaerin ym. [2] malli saavutti havahtumisten tunnistamisessa jopa Lin ja Guanin [7] mallia huomattavasti paremman suorituskyvyn, vaikka se hyödynsi siinä vain PSG-rekisteröinnin EEG-, EOG- ja leuan EMG-signaaleja. On kuitenkin huomioitava, että näiden kahden mallin suorituskykyä ei voida verrata toisiinsa globaalisti, sillä mallien testaamisessa on käytetty eri datajoukkoja. Brink-Kjaerin ja kollegoiden mallin mahdollista parempaa suorituskykyä voisi kuitenkin selittää siinä hyödynnetty kaksisuuntainen LSTM-kerros, joka voi auttaa havahtumisten tunnistamisessa mallintamalla pitkän aikavälin riippuvuuksia. Toisaalta Li ja Guan [7] raportoivat, että LSTM-kerrosten lisääminen heidän malliinsa ei tehostanut sen suorituskykyä.

Kaikissa taulukossa 1 esitetyissä neuroverkkomalleissa hyödynnettiin konvoluutiokerroksia. Näillä on tärkeä rooli havahtumisten luokittelussa, sillä ne vastaavat niiden PSG-datan piirteiden keräämisestä, joiden perusteella data-alkiot voidaan luokitella havahtumiseksi tai valveeksi [2, 7]. Havahtumisten tunnistamiseen on kehitetty myös

neuroverkkomalleja, joissa ei ole konvoluutiokerroksia, mutta näihin malleihin on sisällytetty PSG-datan piirteiden keräämistä varten muita käsin kirjoitettuja algoritmeja, jotka voivat hyödyntää esimerkiksi nopeaa Fourier-muunnosta (engl. *fast Fourier transform*, FFT) tai jatkuvaa aaltomuunnosta (engl. *continuous wavelet transform*, CWT) [6]. Näiden piirteitä keräävien algoritmien ohjelmointi on kuitenkin työlästä [6], joten piirteitä itsenäisesti keräävien konvoluutioneuroverkkojen käyttö voi säästää huomattavasti aikaa ja resursseja.

Tutkielmassa käsitelty havahtumisten tunnistaminen on luokitteluongelma, joka edustaa ohjattua oppimista. Neuroverkkojen kouluttaminen tähän tehtävään vaatii kattavia ja tarkasti annotoituja datajoukkoja PSG-rekisteröinneistä. Näiden datajoukkojen kerääminen vaatii paljon työtä sekä PSG-rekisteröintien mittaamisen että havahtumisten annotointien vuoksi. Lisäksi datajoukon laatu asettaa rajat sen avulla koulutetun neuroverkon suorituskyvylle, sillä ohjatussa oppimisessa neuroverkko voi oppia datasta ainoastaan sen annotointien perusteella [12], eivätkä ne ihmisten tekemänä ole täysin luotettavia [2]. Näistä syistä tulevaisuudessa voisi olla kannattavaa tutkia ohjaamattoman oppimisen hyödyntämistä havahtumisten tunnistamisessa, sillä se ei edellytä PSG-datan annotointeja. Ohjaamaton oppiminen ei suoraan sovellu luokitteluongelmiin, koska siinä käsiteltävässä datassa ei ole ennalta määriteltyjä luokkia, vaan neuroverkko ryhmittelee saamansa datan itsenäisesti parhaalla löytämällään tavalla. Ohjaamatonta oppimista voitaisiin siis hyödyntää muun muassa datan esikäsittelyssä esimerkiksi ryhmittelemällä samankaltaisia data-alkioita eri joukkoihin, joita voitaisiin käyttää erillisen luokittelualgoritmin piirteinä parantaen luokittelun tarkkuutta.

Viime vuosina kerätyt laajat annotoidut datajoukot ovat kuitenkin jo nyt mahdollistaneet neuroverkkopohjaisten luokittelualgoritmien kehittämisen havahtumisten tunnistamiseksi ja näiden mallien suorituskyky tulee varmasti tulevaisuudessa paranemaan entisestään algoritmien kehittymisen ja uusien datajoukkojen myötä. Vaikka ihmisten annotoimat datajoukot eivät olekaan täysin luotettavia, se ei ole esteenä niillä koulutettujen neuroverkkojen käytölle. Mallien tavoitteena on nimittäin korvata ihminen annotoijana, jolloin niiden riittää saavuttaa ihmisen suorituskyky. Lisäksi neuroverkkojen hyödyntämistä havahtumisen tunnistamisessa puoltaa se, että niiden käyttöönotto havahtumisten tunnistamiseen on suhteellisen suoraviivaista niiden monimutkaisesta rakenteesta huolimatta. Esimerkiksi Li ja Guan [7] sekä Zabihi ym. [15] osoittivat, että alunperin muihin tarkoituksiin kehitetyt neuroverkkoarkkitehtuurit voidaan kouluttaa myös havahtumisten tunnistamiseen.

5 Johtopäätökset

Tässä tutkielmassa vertailtujen tutkimusten perusteella havahtumisia on mahdollista tunnistaa automaattisesti polysomnografiarekisteröinneistä neuroverkkopohjaisia menetelmiä hyödyntämällä. Osa tutkielmassa esitellyistä neuroverkkomalleista kykeni tunnistamaan havahtumiset pelkästään PSG-rekisteröinnin EEG-, EOG- ja leuan EMG-signaalien perusteella, mutta lisäsignaalien, kuten EKG:n ja hengityksen ilmavirtauksen käyttäminen saattoi parantaa havahtumisten tunnistamisen tarkkuutta joissakin tutkimuksissa.

Konvoluutiokerrokset ovat olennainen osa havahtumisia tunnistavissa neuroverkkomalleissa, sillä ne vastaavat niiden PSG-rekisteröintien piirteiden keräämisestä, joiden perusteella havahtumiset voidaan tunnistaa. LSTM-kerrosten käyttäminen mallissa saattaa neuroverkon muusta arkkitehtuurista riippuen parantaa sen suorituskykyä, mutta havahtumisia voidaan tunnistaa jo pelkkiä konvoluutiokerroksia ja täysin kytkettyjä kerroksia käyttämällä.

Tällä hetkellä neuroverkkopohjaiset menetelmät ovat olleet havahtumisten tunnistamisessa vain tutkimuskäytössä, mutta niiden kehittäminen ja käyttöönotto kliinisessä työssä voi tulevaisuudessa mahdollistaa nopeamman ja kustannustehokkaamman tavan annotoida havahtumisia. Tähän pisteeseen pääseminen vaatii kuitenkin vielä paljon työtä mallien kehittämisessä ja niiden saavutettavuuden helpottamisessa.

Viitteet

[1] S. Mukherjee, S. R. Patel, S. N. Kales, N. T. Ayas, K. P. Strohl, D. Gozal, A. Malhotra ja American Thoracic Society ad hoc Committee on Healthy Sleep, "An Official American Thoracic Society Statement: The Importance of Healthy Sleep. Recommendations and Future Priorities," *Am J Respir Crit Care Med*, vol. 191, nro 12, s. 1450–1458, 2015.

[2] A. Brink-Kjaer, A. Olesen, P. Peppard, K. Stone, P. Jennum, E. Mignot ja H. Sorensen, "Automatic detection of cortical arousals in sleep and their contribution to daytime sleepiness," *Clinical Neurophysiology*, vol. 131, nro 6, s. 1187–1203, 2020.

[3] M. Blood, R. Sack, D. Percy ja J. Pen, "A comparison of sleep detection by wrist actigraphy, behavioral response, and polysomnography," *Sleep*, vol. 20, s. 388–95, 1997.

[4] R. K. Malhotra ja A. Y. Avidan, "Chapter 3 - Sleep Stages and Scoring Technique," teoksessa *Atlas of Sleep Medicine (Second Edition)*, St. Louis: W.B. Saunders, 2014, s. 77–99.

[5] R. Berry, R. Brooks, C. Gamaldo, S. Harding, R. Lloyd, C. Marcus ja B. Vaughn, "The AASM Manual for the Scoring of Sleep and Associated Events: Rules, Terminology and Technical Specifications, Version 2.2.," *Darien, Illinois: American Academy of Sleep Medicine*, 2015.

[6] X. Qian, Y. Qiu, Q. He, Y. Lu, H. Lin, F. Xu, F. Zhu, Z. Liu, X. Li, Y. Cao ym. "A Review of Methods for Sleep Arousal Detection Using Polysomnographic Signals," *Brain Sci*, vol. 1274, nro 11, 2021.

[7] H. Li ja Y. Guan, "DeepSleep convolutional neural network allows accurate and fast detection of sleep arousal," *Commun Biol*, vol. 4, nro 1, s. 18, 2021.

[8] H. Korkalainen, *Deep Learning for Next-Generation Sleep Diagnostics: sophisticated computational methods for more efficient and accurate assessment of sleep and obstructive sleep apneas*. Itä-Suomen yliopisto, Sovelletun fysiikan laitos: Väitöskirja, 2020.

[9] Y. LeCun, Y. Bengio ja G. Hinton, "Deep learning," *Nature*, vol. 521, nro 7553, s. 436–444, 2015.

[10] Z. Jia, X. Wang, X. Zhang ja M. Xu, "Automatic Arousal Detection Using Multi-model Deep Neural Network," *2020 5th International Conference on Computer and Communication Systems (ICCCS)*, s. 130–133, 2020.

[11] T. Mitchell, *Machine Learning*, 17. McGraw-Hill, 1997.

[12] I. Goodfellow, Y. Bengio ja A. Courville, *Deep Learning*, 2. Cambridge: MIT Press, 2017.

[13] S. Nikkonen, *Novel computational tools for enhanced diagnostics of obstructive sleep apnea*. Itä-Suomen yliopisto, Sovelletun fysiikan laitos: Väitöskirja, 2020.

[14] L. Alzubaid, J. Zhang, A. Humaldi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamarfa, M. Fadhel, M. Al-Amidie ja L. Farhan, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J Big Data*, vol. 8, nro 1, s. 53–127, 2021.

[15] M. Zabihi, A. B. Rad, S. Kiranyaz, S. Särkkä ja M. Gabbouj, "1D Convolutional Neural Network Models for Sleep Arousal Detection," *arXiv*, 2019.

[16] R. Rojas, *Neural Networks: A Systematic Introduction*. Springer, 1996.

[17] C. Aggarwal, *Neural Networks and Deep Learning*. Springer, 2018.

[18] F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Cornell Aeronautical Laboratory Inc, 1961.

[19] C. Bishop, *Neural Networks for Pattern Recognitions*. Oxford University Press, 1995.

[20] K. Hornik, M. Stinchcombe ja H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, nro 5, s. 359–366, 1989.

[21] A. Gulli, *Deep Learning With Keras*. Packt, 2017.

[22] E. Alpaydin, *Introduction to Machine Learning*, 2. The MIT Press, 2010.

[23] D. Miller, A. Ward ja N. Bambos, "Automatic Sleep Arousal Identification From Physiological Waveforms Using Deep Learning," 2018.

[24] M. Ghassemi, B. Moody, L.-w. Lehman, C. Song, Q. Li, H. Sun, B. Westover ja G. Clifford, "You Snooze, You Win: The PhysioNet/Computing in Cardiology Challenge 2018," 2018.

[25] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior ja K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," *arXiv*, 2016.

[26] O. Ronneberger, P. Fischer ja T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, s. 234–241, 2015.

[27] P. Diederik, Kingma ja J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv*, 2017.