

Maksakuvien semanttinen segmentointi

(Liver Medical Image Semantic Segmentation)

Itä-Suomen yliopisto Luonnontieteiden ja metsätieteiden tiedekunta Tietojenkäsittelytieteen laitos

Denis Eskelinen ja Lauri Komulainen

Sisällys

[1 Johdanto 2](#_Toc181446161)

[2 Projektin kuvaus ja ratkaisumenetelmät 3](#_Toc181446162)

[3 Toteutus 4](#_Toc181446163)

[3.1 Mallin koulutus strategia ja tulokset 5](#_Toc181446164)

[3.2 Mallin suorituskyky 6](#_Toc181446165)

[4 Lainaukset ja viittaukset 6](#_Toc181446166)

[5 Ryhmätyön jako 6](#_Toc181446167)

[6 Johtopäätökset 6](#_Toc181446168)

[Liitteet 6](#_Toc181446169)

[Lähdeluettelo 7](#_Toc181446170)

# **1 Johdanto**

Lääketieteellisten kuvien analysointi on keskeinen osa nykyaikaista diagnostiikkaa ja potilashoitoa. Kuvantamistekniikoiden, kuten tietokonetomografian (TT) ja magneettikuvauksen (MRI), kehittyessä terveydenhuollon ammattilaiset voivat tarkastella ihmiskehon sisäisiä rakenteita yksityiskohtaisesti. Erityisesti maksasairauksien, kuten maksasyövän tai maksakirroosin, diagnosointi ja seuranta edellyttävät tarkkaa ja luotettavaa kuvantamista.

Semanttinen segmentointi on kuvankäsittelytekniikka, jossa kuva jaetaan semanttisesti merkityksellisiin osiin siten, että jokaiselle pikselille annetaan luokka. Tämä mahdollistaa esimerkiksi maksan erottamisen muista elimistä ja kudoksista lääketieteellisissä kuvissa. Manuaalinen segmentointi on aikaa vievää ja altis inhimillisille virheille, joten automaattiset segmentointimenetelmät voivat vähentää ihmisen tekemiä inhimillisiä virheitä.

Syväoppiminen ja erityisesti U-Net-arkkitehtuuri ovat osoittautuneet tehokkaiksi lääketieteellisten kuvien segmentoinnissa. U-Net on konvoluutio-neuroverkko, joka on suunniteltu erityisesti biolääketieteellisten kuvien analysointiin. Se yhdistää alas- ja ylöspäin suuntautuvat verkko-osat, mikä mahdollistaa sekä globaalien että paikallisten piirteiden hyödyntämisen tarkassa segmentoinnissa.

Tämän projektin päätavoitteena on kehittää syväoppimiseen perustuva järjestelmä, joka suorittaa semanttisen segmentoinnin maksakuville käyttäen U-Net-mallia, ja siten syventää osaamista konvoluutioneuroverkoista ja niiden soveltamisesta käytännön lääketieteellisiin ongelmiin.

# **2 Projektin kuvaus ja ratkaisumenetelmät**

Tähän vois kertoo projektista ja U-Net neuroverkosta. Varmaan myös mitä muuta voisi käyttää tähän ongelmaan – lähteistettyä tekstiä. Mitä mahdollisia haasteita ja rajotteita on? U-net arkkitehtuurin hyödyt ja haitat, miksi paras tähän?

# **3 Toteutus**

Mallimme on toteutettu Python-ohjelmointikielellä ja PyTorch-koneoppimiskirjastoa käyttäen. Malli käyttää U-Net arkkitehtuuria, joka koostuu kymmenestä konvoluutiokerroksesta. Toteutus on jaettu kolmeen eri tiedostoon:

* **Main.py**, jossa ajetaan jokaisella kierroksella mallin koulutus ja siitä luodaan raportti
* **Model.py**, jossa luodaan mallin arkkitehtuuri ja suoritetaan mallin koulutus erilaisilla hyperparametri yhdistelmillä.
* **data\_handler.py**, jossa esikäsitellään ja jaetaan kuvat labeleihin, testaukseen, mallin harjoitukseen sekä validointiin.

## **3.1 Mallin koulutus strategia ja tulokset**

Mallin koulutuksessa pyrimme löytämään optimaalisen hyperparametriyhdistelmän, joka maksimoisi mallin suorituskyvyn ja parantaisi sen yleistymiskykyä. Koska yksittäisen koulutuskierroksen kesto vaihteli asetuksista riippuen noin 1–4 tuntiin, laajan grid search -haun käyttö ei ollut aikataulullisesti mahdollista. Tämän vuoksi hyödynsimme ***model.py***-tiedostossa toteutettua ’*random\_search’*-funktiota, jonka avulla mallia koulutettiin satunnaisesti valituilla hyperparametriyhdistelmillä. Lisäsimme myös ***model.py***-tiedostoon *’early stopping*’ rakenteen, jossa käytimme viiden kierroksen raja-arvoa. Jos validaatiohäviö ei parantunut viiden kierroksen aikana, koulutus keskeytyi automaattisesti, ja tallensimme siihen mennessä parhaan mallin analysointia varten. Tämä mahdollisti mallin ylikoulutuksen havaitsemisen ja sen automaattisen lopettamisen, ennen kuin suorituskyky heikkeni.

Koulutus suoritettiin kahdella tehokkaalla tietokoneella, joissa molemmissa oli suorituskykyinen CPU ja GPU. Satunnaisen haun avulla kävimme läpi yhteensä **XX** erilaista parametriyhdistelmää seuraavasta hyperparametrijoukosta:

* Eräkoko (batch size): [4, 8, 16]
* Oppimisnopeus (learning rate): [1e-3, 1e-4, 1e-5]
* Dropout-määrä: [0.1, 0.2, 0.3]
* Optimoijatyyppi: [Adam, SGD]
* Epoch-määrä: [1, 10, 20, 30]

Jokaisen koulutuskierroksen jälkeen ***Main.py***-tiedosto loi yksityiskohtaisen raportin, joka sisälsi:

* Koulutuksessa käytetyt hyperparametrit
* Harjoitus- ja validointitappiot
* Mallin keskimääräinen Intersection over Union (mIoU) ja keskimääräinen Pixel Accuracy (mPA)
* Segmentointikuvat

Nämä raportit on koottu liitteisiin, ja niiden analyysin perusteella valitsimme parhaat hyperparametrit. Havaitsimme, että seuraavalla yhdistelmällä saavutettiin paras suorituskyky:

* **Eräkoko**: x
* **Oppimisnopeus**: y
* **Dropout-arvo**: t
* **Optimoija**: z
* **Kierrosten määrä (Epochs)**: xx

Tällä parametriyhdistelmällä mIoU oli 0.xxx ja mPA-arvo 0.xxx, mikä osoittaa mallin saavuttaneen erittäin hyvän segmentointitarkkuuden sekä yleistyskyvyn validaatiodatan kanssa. Lopullinen malli tallennettiin tiedostoon x\_y\_model.pth, joten sitä voidaan hyödyntää jatkossa ilman uudelleenkoulutusta. Tämä mahdollistaa mallin suoran lataamisen ja käyttöönoton tuotantoympäristöissä.

## **3.2 Mallin suorituskyky**

Tähän voisi sitten laittaa parhaasta mallista hieman lisää ja erilaisia kuvia!

# **4 Lainaukset ja viittaukset**

Mallin rakentamisen apuna on käytetty ChatGPT 4o mallia, mutta malli on optimoitu meidän toimesta.

Ensimmäisessä vaiheessa

Toisessa vaiheessa

Kolmannessa vaiheessa

# **5 Ryhmätyön jako**

Ryhmätyö jaettiin tasasti molemmille. Kumpikin pääsi tekemään tekoälymallia sekä raporttia. Molempien osaamista hyödynnettiin aiemman kokemuksen pohjalta.

# **6 Johtopäätökset**

Tiivistä työn keskeiset löydökset.

Pohdi mahdollisia parannuksia ja jatkotutkimusaiheita.

# **Liitteet**

# **Lähdeluettelo**