Norges miljø- og biovitenskapelige universitet

Masteroppgave 2021 30 stp
Fakultet for realfag og teknologi

Automatisk segmentering av hode- og halskreft i PET/CT-bilder ved bruk av konvolusjonsnettverk

Automatic segmententation of head and neck cancer in PET/CT images using convolutional neural networks

Sofie Roko Krogstie
Miljøfysikk og fornybar energi

Forord

Denne masteroppgaven er skrevet ved Fakultet for realfag og teknologi ved Norges miljø- og biovitenskapelige universitet våren 2021. Oppgaven utgjør 30 studiepoeng og markerer avslutningen på en femårig mastergrad i Miljøfysikk og fornybar energi.

Jeg vil gjerne rette en stor takk til Professor Cecilia Marie Futsæther. Cecilia har vært min hovedveileder og har bidratt med inspirerende og grundige tilbakemeldinger samt fantastisk oppfølging. Jeg vil også rette en stor takk til Ph.d.-stipendiatene Aurora Rosvoll Grøndahl og Bao Ngoc Huynh for gode svar på spørsmål og for å alltid ha vært behjelpelige. Jeg føler meg heldig som har fått muligheten til å samarbeide med dere.

Takk til Førsteamanuensis Oliver Tomic og gruppen til Professor Kathrine Røe Redalen fra Trondheim for interessante og spennende diskusjoner på våre møter. Takk til Professor Eirik Malinen fra Universitet i Oslo for å ha bidratt med datasettet brukt i denne oppgaven.

Videre vil jeg takke venner og familie for støtte og oppmuntring under arbeidet med denne oppgaven. En spesielt stor takk rettes til Malene Gjengedal og Maria Ødegaard for godt samarbeid, støtte og motivasjon gjennom hele studietiden og denne masteroppgaven. Tusen takk til alle mine medstudenter, og spesielt mitt kjære kollektiv, for fem fantastiske år på Ås, og uforglemmelige minner.

Ås, 01.06.2021

_____

Sofie Roko Krogstie

Sammendrag

Formål

Nøyaktig inntegning av kreftsvulster blir ansett som det svakeste leddet i planleggingen av strålebehandling, både fordi det er en tidskrevende oppgave, men også grunnet inter- og intravariabilitet. En nøyaktig inntegning er viktig for å sikre høy nok stråledose til kreftsvulsten og affiserte lymfeknuter samt å forhindre skade på omkringliggende vev og risikoorganer. Dette er spesielt viktig hos pasienter med hode- og halskreft, da dette er områder med kompleks anatomi. Formålet med denne masteroppgaven er å undersøke et konvolusjonsnettverk for bruk til automatisk inntegning av kreftsvulster og affiserte lymfeknuter i PET/CT-bilder av pasienter med hode- og halskreft. Bruken av en automatisk inntegningsmodell vil potensielt gjøre inntegningene mindre tidkrevende samt gi mer konsekvente og nøyaktige inntegninger.

Metode

Datasettet som ble brukt bestod av PET- og CT-bilder fra 197 pasienter med hode- og halskreft som mottok behandling ved Oslo universitetssykehus med behandlingsstart mellom 2007 og 2013. Unionen av manuelle inntegninger fra tre erfarne radiologer, ble brukt som sanne inntegninger til trening og evaluering av konvolusjonsnettverket. Datasettet ble delt i et trenings-, validerings- og testsett, stratifisert etter tumorstadium.

Den automatiske inntegningen av kreftsvulster og affiserte lymfeknuter i PET/CT-bildene ble gjort ved å bruke et konvolusjonsnettverk med en 2D VoxResNet-arkitektur. Rammeverket *deoxys* ble brukt for å kjøre 36 eksperimenter med forskjellige parameternivå til parameterne læringsrate, antall filtre, dropoutrate og batchnormalisering. Ytelsen ble vurdert etter grad av overlapp mellom sann inntegning og predikert inntegning, med ytelsesmålet Dice-score. Resultatene fra eksperimentene ble brukt i statistiske tester for å vurdere om valget av parameternivå ga en signifikant forskjell i modellytelsen og for å bestemme hvilken kombinasjon av parameternivå som ga den mest nøyaktige inntegningen og dermed den beste modellen. Deretter ble datasettet utvidet med bildeaugmentering for å undersøke hvordan dette påvirket ytelsen til denne beste modellen. Både modellen uten og med bildeaugmentering ble vurdert med testsettet. For å til slutt undersøke hvordan modellen presterte på helt usett og ukjent data fra et annet sykehus, ble også modellene vurdert med et eksternt testsett fra Maastro Clinic, Nederland.

Resultat

De statistiske testene viste at det var en signifikant forskjell mellom valg av parameternivå for de ulike parameterne. Nivåene av parametere som ga høyest gjennomsnittlig Dice-score var 48 filtre, 10^{-4} i læringsrate, null i dropoutrate og å inkludere batchnormalisering i modellen. Denne beste modellen fikk en gjennomsnittlig Dice-score per pasient på 0,714 på testsettet. Etter bruken av bildeaugmentering presterte modellen bedre, med en gjennomsnittlig Dice-score per pasient på 0,731 vurdert på testsettet. For Maastro-testsettet oppnådde modellen uten bildeaugmentering en gjennomsnittlig Dice-score per pasient på 0,629 og modellen med bildeaugmentering fikk en gjennomsnittlig Dice-score per pasient på 0,635.

For å undersøke hvor modellen feilet, ble bilder med sanne og predikerte inntegninger sammenliknet. Modellen ser ut til å ha vanskeligheter med å tegne inn små tumorstørrelser og å foreta inntegninger på bilder som er atypiske fra resten av bildene.

Konklusjon

Bruken av konvolusjonsnettverket 2D VoxResNet til automatisk inntegning av kreftsvulster og affiserte lymfeknuter i PET/CT-bilder hos pasienter med hode- og halskreft har vist lovende resultater og har et stort potensial. Til tross for dette, presterer modellen dårlig i enkelte tilfeller, noe som indikerer at modellen bør forbedres før den kan brukes klinisk. En videre utvikling av modellen innebærer videre undersøkelser av parameternivå, undersøkelse av interaksjon mellom parameternivåene og utprøving av flere bildeaugmenteringsteknikker.

Abstract

Purpose

Precise delineation of tumors is considered the weakest link in radiotherapy treatment planning because it is a time-consuming task, that can be affected by inter- and intraobserver variability. Precise delineation is crucial to prevent irradiation and damage to surrounding tissue and organs at risk. This is especially important in the head and neck region, as these are areas with complex anatomy. The purpose of this thesis was to explore a convolutional neural network (CNN) for automatic delineation of tumors and affected lymph nodes in PET/CT-images of patients with head and neck cancer. The use of an automatic delineation model may potentially save time, providing more consistent and accurate delineations.

Method

The dataset used in this thesis consisted of PET and CT images from 197 patients with head and neck cancer who received treatment at Oslo University Hospital between 2007 and 2013. The union of manual delineations from three experienced radiologists was used as ground truth for training and evaluation of the CNN. The dataset was split into a training, validation and test set, stratified by the tumor stage.

The CNN 2D VoxResNet was used for automatic segmentation of cancerous tumors and affected lymph nodes in PET/CT images. The *deoxys* framework was used to run 36 experiments with different levels for the parameters learning rate, number of filters, dropoutrate and batchnormalization. The performance was assessed according to the degree of overlap between the ground truth and the predicted segmentation, using the performance metric Dice-score. The results from the experiments were used in statistical tests to assess whether there were significant differences between the model performances obtained for the different parameter levels and to determine which combination of parameter levels that gave the most accurate segmentation and thus the best model. The dataset was then expanded with image augmentation to examine how this affected the performance of the best model. Both the model without and with image augmentation were evaluated with the test set. Finally, to examine how the model performed on completely unseen and unknown data, the models were evaluated with the external test set from the Maastro Clinic, The Netherlands.

Results

The statistical tests showed that there was a significant difference between the choice of parameter levels for the different parameters. The parameter levels which gave the highest average Dice score were 48 filters, 10^{-4} in the learning rate, zero in dropoutrate and inclusion of batch normalization in the model. The best model achieved an average Dice-score per patient of 0.714 on the test set. Image augmentation improved the model performance, with an average Dice-score per patient of 0.731 on the test set. For the Maastro test set, the model without augmentation achieved an average Dice-score per patient of 0.629 and the model with augmentation achieved an average Dice-score per patient of 0.635.

To examine where the model failed, images showing the ground truth and predicted segmentations were examined. The model seems to have difficulties segmenting small tumorvolumes and making delineations on images that are atypical from the rest of the images.

Conclusions

The convolutional neural network 2D VoxResNet has shown promising results and has potential for automatic delineation of head and neck cancerous tumors and affected lymph nodes in PET/CT images. However, the best model performs poorly in some cases, indicating that the model should be improved before it can be used clinically. Further development of the model involves further investigations of the parameter levels, investigation of interactions between the parameter levels and testing of several other image augmentation techniques.

Innholdsfortegnelse

Liste over forkortelser

Adam	Adaptive momentum estimation
ANOVA	Analysis of variance / Variansanalyse
API	Application Programming Interface
CIGENE	Centre for Interactive Genetics
CNN	Convolutional neural network / Konvolusjonsnettverk
CPU	Central Processing Unit
CT	Computertomografi
CTV	Clinical target volume
DBMS	Database Management System
EU	Den europeiske union
FCN	Fully convolutional neural network / Fullt konvolusjonsnettverk
FDG eller 18F-FDG	Fluorodeoksyglukose
FN	Falsk negativ
FP	Falsk positiv
GPU	Graphics Processing Unit
GTV	Gross tumor volume
HD	Hausdorff-distanse
HD_{95}	95. persentil Hausdorff-distanse
HDF5	Hierarchical Data Format version 5
HECKTOR	HEad and neCK TumOR
HNC	Head and neck cancer
HPV	Humant papillomavirus
HU	Hounsfield Unit
JSON	JavaScript Objection Notation
LOR	Line of Response
MICCAI	Medical Image Computing and Computer Assisted Intervention
MR	Magnetisk resonans
MSD	Median surface distance / Median overflatedistanse
NMBU	Norges miljø- og biovitenskapelige universitet

OUS	Oslo universitetssykehus
PET	Positronemisjonstomografi
PPV	Positiv prediktiv verdi, Presisjon
QQ-plot	Quantile-Quantile plot
RAM	Random Access Memory
ReLU	Rectified Linear Unit
RGB	Rød, Grønn, Blå
SN	Sann negativ
SNR	Sann positiv rate, Spesifisitet
SP	Sann positiv
SPR	Sann positiv rate, Sensitivitet
SSE	Sum of squared error
SSH	Secure Shell Client
SUV	Standardized Uptake Value
TNM	Tumor, nodes og metastaser

Kapittel 1: Innledning

1.2 Motivasjon

1.2.1 Hode og halskreft

Kreft er en samlebetegnelse for sykdom som skyldes ukontrollert celledeling. Kreftceller har evnen til å spre seg til omkringliggende vev samt andre organer i kroppen gjennom for eksempel blodsystemet eller lymfesystemet [1]. Ifølge Verdens helseorganisasjon var kreft dødsårsaken til om lag 10 millioner mennesker i verden i 2020 [2]. Hode- og halskreft («Head and neck cancer», HNC) omfatter kreft som rammer strupehodet, lepper, munnhule, svelg, nese, bihuler og spyttkjertler [3]. Dersom denne typen kreft sprer seg skjer det vanligvis til lymfeknuter i halsen, og det er sjeldent at hode- og halskreft sprer seg til andre steder i kroppen. Antall nye tilfeller per år av kreft i hode- og halsregionen har de siste årene vært svakt stigende. Forskning viser at økningen kan i noen grad skyldes kreft forårsaket av infeksjon av HPV [3]. I 2019 ble 648 mennesker diagnostisert med kreft i hode- og halsregionen i Norge, 423 menn og 255 kvinner [3]. Dersom kreften blir oppdaget tidlig er overlevelsesraten 84,8 prosent for menn og 90,2 prosent for kvinner fem år etter diagnosen. Dersom kreften har spredd seg før den oppdages er det 68,6 prosent av mennene og 75,7 prosent av kvinnene som lever etter fem år [3].

Helsedirektoratet har utviklet retningslinjer for diagnostikk, behandling og oppfølging av pasienter med hode- og halskreft. Ifølge disse retningslinjene skal pasienter som er under utredning for hode- og halskreft gjennomgå en CT-undersøkelse med kontrastmiddel for å kartlegge tumorutbredelse og lymfeknutemetastaser [4]. En PET/CT-undersøkelse brukes for å skille godartede forandringer fra kreftsvulster, vurdere utbredelse av kreftsykdom samt vurdere effekt av behandling. PET-bildene viser områder hvor det er en opphopning av celler med høy metabolsk aktivitet. CT-bildene viser absorpsjonen av røntgenstråler i vevet, som gir et detaljert anatomisk bilde av området [5].

De vanligste behandlingsformene for hode- og halskreft er strålebehandling, kirurgi og cellegift. Ved noen tilfeller av spredning av kreftsykdommen eller tilbakefall kan immunterapi brukes [3]. Kirurgisk behandling vil bestå av å fjerne deler av eller hele svulster og eventuelle lymfeknuter dersom det er spredning til disse. Cellegift brukes noen ganger før strålebehandling for å minske svulstens størrelse eller ved tilbakefall hvor svulsten ikke kan fjernes kirurgisk eller behandles med stråling. Strålebehandling kan brukes i tilfeller hvor

kirurgi ikke er mulig grunnet plassering og omfang av svulsten. Strålebehandling brukes også som en tilleggsbehandling etter kirurgisk behandling, da det kan sitte igjen rester av kreftceller i vevet rundt inngrepet [3]. For å sikre en at svulsten får høy nok stråledose, men også at det omkringliggende vevet til svulsten ikke blir skadet av strålebehandlingen, er det avgjørende med en nøyaktig inntegning av svulsten.

1.2.2 Utfordringer innen kreftbehandling

En av utfordringene når det kommer til kreftbehandling er ventetiden for pasienten fra diagnose til behandlingsstart. Inntegning av kreftsvulstene gjøres i dag manuelt av radiologer eller onkologer som et ledd i planleggingen av strålebehandling, noe som er en tidkrevende oppgave. Selv om Louireno et at. [6] har vist at ventetiden mellom diagnose og behandlingsstart ikke har noen signifikant betydning for behandlingsutfallet, kan tiden være kostbar for sykehusene og krevende for pasientene.

En annen utfordring er nøyaktigheten til inntegningene av kreftsvulstene gjort av radiologene. På grunn av intervariabilitet vil inntegningene kunne variere ut ifra hvilken radiolog som har gjort inntegningen. Den samme radiologen kan også ha en variasjon i inntegningen av samme kreftsvulst tegnet inn på ulike dager, dette kalles intravariabilitet [7]. Nøyaktighet i inntegningene av kreftsvulstene er en avgjørende del av planleggingen av kreftbehandlingen, og ifølge Njeh [8] blir det regnet som det svakeste leddet og den største bidragsyteren til usikkerhet i planleggingsprosessen. En feil i inntegningen vil generere en systematisk feil i den endelige behandlingsplanen [8]. Dette vil avhenge av området i kroppen hvor inntegningen blir gjort. Weiss og Hess [7] viste i sin studie at områdene som ga størst variabilitet i inntegningene var hode- og halskreft samt spiserør- og lungekarsinomer. En nøyaktig inntegning av kreftsvulstene vil være avgjørende for behandlingsutfallet og livskvaliteten til pasienten etter kreftbehandlingen. Målet er å gi en høy dose til kreftsvulsten og begrense dosen til risikoorganer og friskt vev [3].

1.2.3 Kunstig intelligens i helsevesenet

Interessen og utviklingen innenfor kunstig intelligens og dyp læring har vært økende de siste årene [9]. Den økende interessen kan delvis skyldes utviklingen som har skjedd innenfor beregningskraft og datalagring [10]. Dyplæringsmetoder som baseres på konvolusjonsnettverk har vist lovende resultater for segmentering i medisinske avbildninger [11, 12]. Ved bruk av disse metodene kan det være mulig å lage en modell som automatisk segmenterer kreftsvulsten, som gir et standard resultat for inntegningen. Dersom den automatiske

inntegningen er tilstrekkelig nøyaktig i forhold til den manuelle inntegningen, vil dette eliminere problemet med inter- og intravariabilitet og være tidsbesparende for radiologene.

Det er naturligvis en viss grad av skepsis knyttet til bruken av kunstig intelligens i helsevesenet. En av disse bekymringene kan være at radiologer kan være bekymret for at de ikke lengre vil være nødvendige å ha i arbeid, for algoritmer basert på kunstig intelligens vil gjøre jobben deres mer nøyaktig og raskere [13]. Andre bekymringer kan være tilknyttet troverdigheten og nøyaktigheten, samt usikkerhet tilknyttet personvern og sikkerhet for medisinsk data, som kunstig intelligens kan gi [13].

1.3 Automatisk svulstinntegning i hode- og halskreft

Flere nylige studier som tar i bruk konvolusjonsnettverk for automatisk svulstinntegning av hode- og halskreft har vist lovende resultater. Modellene foretar inntegninger av *gross tumor volume* (GTV) eller *clinical target volume* (CTV). GTV er definert som omfanget av tumoren som kan påvises i blant annet medisinske bilder. CTV inkluderer både tumoren og annet vev som kreften har spredning til, slik som f.eks. lymfeknuter [7]. *International Conference on Medical Image Computing and Computer Assisted Intervention* (MICCAI) organiserte i 2020 utfordringen HECKTOR (*HEad and NeCK tumOR*). Oppgaven i utfordringen var å lage en modell for automatisk segmentering av GTV i FDG-PET/CT-bilder hos pasienter med hode- og halskreft, med fokus på kreft i svelget [14]. Bidragene til konkurransen brukte ulike nettverksarkitekturer og flere oppnådde gode resultater. Lin et al. [12] har også undersøkt automatisk inntegning av kreftsvulster i MR-bilder ved bruken av nettverksarkitekturen VoxResNet, noe som ga lovende resultater. De viser også i sitt arbeid at bruken av automatisk segmentering av kreftsvulster som et hjelpemiddel for radiologer, reduserer inntegningstiden samt inter- og intravariabiliteten. Selv med gode resultater, tar ikke disse studiene hensyn til affiserte lymfeknuter. Dette er noe som bør inkluderes for å oppnå adekvat behandling av pasienten, siden en typisk pasient med hode- og halskreft har en eller flere affiserte lymfeknuter [15]. Denne problemstillingen ble det derfor tatt hensyn til og undersøkt av Guo et al. [16] med DenseNet samt Moe et al. [17] og Groendahl et al [18], begge med U-Net-arkitektur.

1.4 Mål for masteroppgaven

Formålet med denne masteroppgaven var å undersøke konvolusjonsnettverket 2D VoxResNet basert på Lin et al. [12] sin arkitektur, for automatisk segmentering av kreftsvulster og affiserte lymfeknuter i PET/CT-bilder for pasienter med hode- og halskreft. Datasettet består

av pasienter med hode- og halskreft behandlet ved Oslo universitetssykehus (OUS) med behandlingsstart i perioden 2007 til 2013 [19]. Det ble undersøkt hvilke verdier av fire ulike parametere i VoxResNet-modellen som ga de mest nøyaktige inntegningene. Det ble også undersøkt effekten av bildeaugmentering på VoxResNet-modellen. Nøyaktigheten til modellene ble evaluert og sammenlignet med konvolusjonsnettverket U-Net fra Moe et al. [17], som også er trent og evaluert på datasettet med pasienter fra OUS. Den endelige modellen ble også testet på et eksternt testsett, besående av pasienter med hode- og halskreft fra Maastro Clinic, Nederland.

1.5 Organisering av oppgaven

Oppgaven begynner med å forklare teorien bak metodene brukt. I Kapittel 2 blir CT, PET og PET/CT beskrevet. Dette innebærer de fysiske prinsippene brukt i avbildningsmetodene samt rekonstrueringen av bildene. Videre blir prinsipper innen kunstig intelligens beskrevet i Kapittel 3. Metoder for klassifisering av bilder og mål av modellytelse blir også beskrevet. Kapittel 4 tar for seg en gjennomgang av datagrunnlaget, rammeverket, optimaliseringsprosessen og vurderingen av ytelsen. Resultatene fra eksperimentene blir fremstilt i Kapittel 5 og diskutert i Kapittel 6. Til slutt blir det gitt en konklusjon av resultatene og eksperimentene i Kapittel 7.

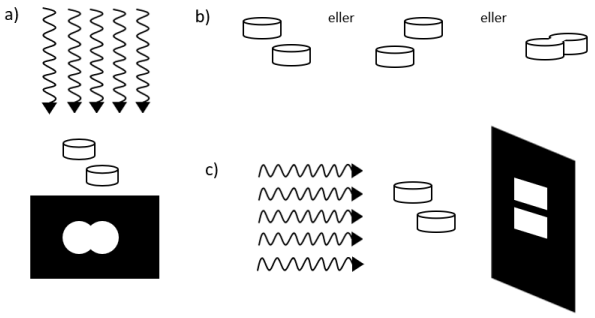
Kapittel 2: Medisinsk avbildning

2.1 Prinsipper innen medisinsk avbildning

Innenfor medisinsk diagnostikk er bruken av avbildning en nyttig og mye brukt metode for å kartlegge ulike organer og vev i menneskekroppen uten et invasivt inngrep på pasienten [20]. Denne kartleggingen er en viktig del av prosessen med å stille en diagnose, men brukes også til planlegging av behandling og oppfølging av en rekke sykdommer. Det finnes flere ulike metoder for avbildning som kan brukes avhengig av hvilken vevstype eller organ som ønskes å kartlegges. Computertomografi (CT), positronemisjonstomografi (PET) og magnetisk resonans (MR) er eksempler på avbildningsmetoder som utnytter ulike fysiske prinsipper [20].

2.2 Computertomografi

Røntgenstråler er fotoner med bølgelengde mellom 0,01 og 10 nanometer [21]. I et konvensjonelt røntgenbilde blir røntgenstråler sendt gjennom en del av kroppen, med en detektor på baksiden av kroppen som detekterer røntgenstrålene som har blitt transmittert [20]. De forskjellige vevstypene i kroppen vil absorbere røntgenstrålene i ulik grad. Vev som absorberer mye røntgenstråler, slik som bein, vil fremstå som lyse skygger på røntgenbildet. Mykt vev (muskler, organer) har en mindre absorberingsevne og vil derfor fremstå mørkere. Siden røntgenbildet er en projeksjon av regionens evne til å absorbere røntgenstråler, vil det være vanskelig å se hvordan objektene er i forhold til hverandre. Det er ikke mulig å se om et objekt er foran eller bak et annet, på bildet vil de overlappe, slik som vises i Figur 2.1a). Dette gjør at røntgenbildet mister dybdeinformasjon og gir et flatt planbilde av kroppen [22]. En metode for å skape bedre dybdeinformasjon er å ta flere røntgenbilder av samme region i flere ulike vinkler, slik som i CT [21]. Figur 2.1 viser hvordan dette prinsippet gir en bedre dybdeinformasjon.

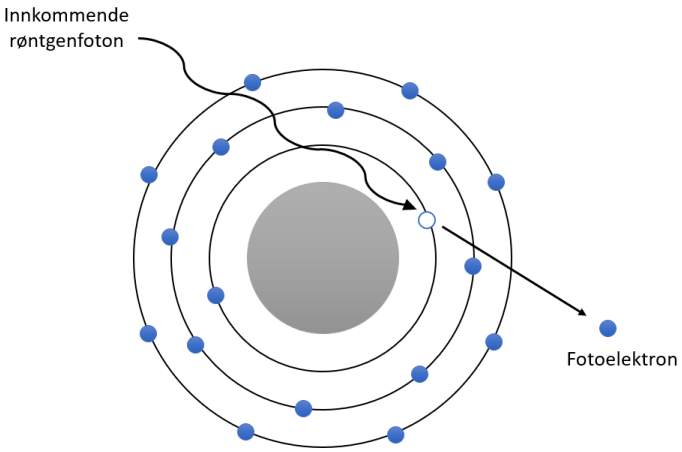


***Figur 2.1**: a) Illustrasjon som viser hvordan projeksjonen i et konvensjonelt røntgenbilde ikke klarer å skille om objektene overlapper. Bildet klarer ikke å vise hvilket av de tre tilfellene i b) som er den virkelige plasseringen. c) viser hvordan flere projeksjoner kan bidra til å avgjøre orienteringer til objekter. Illustrasjon laget med inspirasjon fra figur i Introduction to Physics in Modern Medicine av S. Kane og B. Gelman (2020) [22].*

Røntgenstråler har to ulike egenskaper som er viktig å ta hensyn til når det kommer til avbildning av kroppens anatomi. Den ene er røntgenstrålens evne til å trenge seg gjennom kroppen og nå detektoren på baksiden. Den andre egenskapen er at røntgenstråler blir absorbert i ulik grad i forskjellig vev i kroppen, avhengig av tettheten og den kjemiske sammensetningen av vevet [22]. Når en røntgenstråle sendes gjennom kroppen kan den vekselvirke på tre ulike måter: fotoelektrisk effekt, Comptonspredning eller pardannelse [22]. Dersom ingen av disse hendelsene skjer, vil røntgenstrålen gå gjennom kroppen uten å bli absorbert. Detektoren vil observere både de røntgenstrålene som ikke absorberes og de som blir spredt i kroppen [22].

2.2.1 Fotoelektrisk effekt

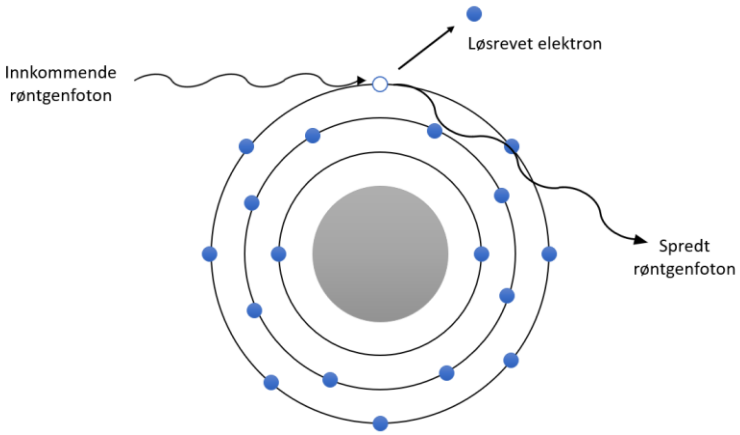
Når en røntgenstråle sendes gjennom kroppen, kan stålen absorberes. Dersom energien til den innkommende røntgenstrålen er under 25 keV, vil absorbsjonen skje ved fotoelektrisk effekt [23]. Når dette skjer, vil et foton vekselvirke med et elektron i et av de innerste elektronskallene til et atom i mediet, vist i Figur 2.2. Elektronene i de innerste skallene er tett bundet til atomet, og krever mer energi for å løsrives sammenlignet med elektroner i de ytre elektronskallene. All energien fra det innkommende fotonet vil bli overført til elektronet, og det opprinnelige fotonet vil ikke lengre eksistere. Dersom denne overførte energien er lik eller større enn bindingsenergien til elektronskallet vil elektronet få nok energi til å løsrives fra atomet. Et slikt fritt elektron kalles et fotoelektron. Den ledige plassen i elektronskallet vil raskt fylles opp igjen av et annet elektron, og fotoelektronet vil kunne bevege seg en liten distanse i kroppen før det blir fanget opp av et annet atom [22].



***Figur 2.2**: Fotoelektrisk effekt. Et foton løsriver et elektron fra det innerste elektronskallet til et atom.*

2.2.2 Comptonspredning

Sannsynligheten for fotoelektrisk effekt synker når energien til røntgenstrålene øker [22]. Ved høyere energier er det en annen prosess som blir mer betydelig, og denne kalles Comptonspredning. Et røntgenfoton vil vekselvirke og løsrive et elektron i et av de ytre elektronskallene til atomet, illustrert i Figur 2.3. Denne interaksjonen er mye mindre energikrevende enn ved fotoelektrisk effekt. Fotonet fortsetter etter interaksjonen, men med en endret vinkel. Bevaringen av energi i sammenstøtet gjør at energien til det opprinnelige fotonet, nå er delt mellom fotonet og elektronet. Siden det utgående fotonet har mindre energi enn det innkommende, vil bølgelengden være større for det utgående [24]. Fotonet vil kunne fortsette med Compton-interaksjoner og gradvis miste energien. Vi sier at dette fotonet har blitt Compton-spredt, og dette kan oppfattes som støy på røntgenbilder [22].



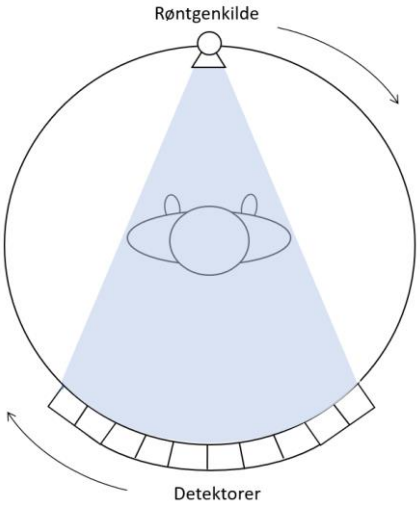
***Figur 2.3:** Comptonspredning. Et innkommende røntgenfoton vekselvirker med et elektron i det ytterste elektronskallet til et atom. Dette fører til et løsrevet elektron og et spredt foton med lavere energi enn det innkommende fotonet.*

2.2.3 Pardannelse

Når et foton med tilstrekkelig energi beveger seg inn i det elektriske feltet til et atom, dempes fotonet fullstendig og det dannes et elektron-positronpar. Pardannelse kan kun oppstå dersom energien til det innkommende fotonet er over 1022 keV [25]. Årsaken til at det kreves denne energien til det innkommende fotonet, er at den må minst tilsvare hvilemassen til de to partiklene som oppstår. For et elektron-positronpar er denne 511 keV × 2 = 1022 keV. Dersom det innkommende fotonet har høyere energi enn det som minimum kreves, vil energien fordele seg tilfeldig mellom elektronet og positronet [25].

2.2.4 Prinsipper innen CT

Computertomografi (CT) bruker røntgenbilder tatt fra ulike vinkler i samme plan for å skape todimensjonale snittbilder av en del av kroppen [20]. En CT-skanner består av en ring med hundrevis av detektorer og en røntgenkilde som roterer rundt pasienten, slik som vises i Figur 2.4 [26]. Pasienten ligger på en seng som beveges inn og ut av ringen, og på denne måten dannes det sekvenser med snittbilder av ønskede regioner i kroppen [27].



***Figur 2.4**: Oppsettet av en CT-skanner. Både rekken med detektorer og røntgenkilden roterer rundt pasienten og lager projeksjoner i mange vinkler innenfor et plan. Illustrasjon laget med inspirasjon fra figur i Introduction to Physics in Modern Medicine av S. Kane og B. Gelman (2020) [22].*

Som nevnt tidligere er et røntgenbilde en projeksjon av røntgenstrålene som har blitt transmittert gjennom kroppen. Hvor mye av røntgenstrålene som transmitteres avhenger av attenuasjonsevnen til vevet [22]. Attenuasjonsevnen til et stoff er avhengig av attenuasjonskoeffisienten μ og massetettheten til vevet. Desto høyere en attenueringskoeffisient er, jo lettere demper dette materiale røntgenstrålen. Enhver kjemisk

forbindelse har en spesifikk attenueringskoeffisient. Materiale som består av flere ulike kjemiske forbindelser, slik som kroppen, vil ha en attenueringskoeffisient som er avhengig av gjennomsnittet til de kjemiske forbindelsene [22]. Den totale attenuasjonskoeffisienten er gitt som en sum av bidragene fra de ulike vekselvirkningene:

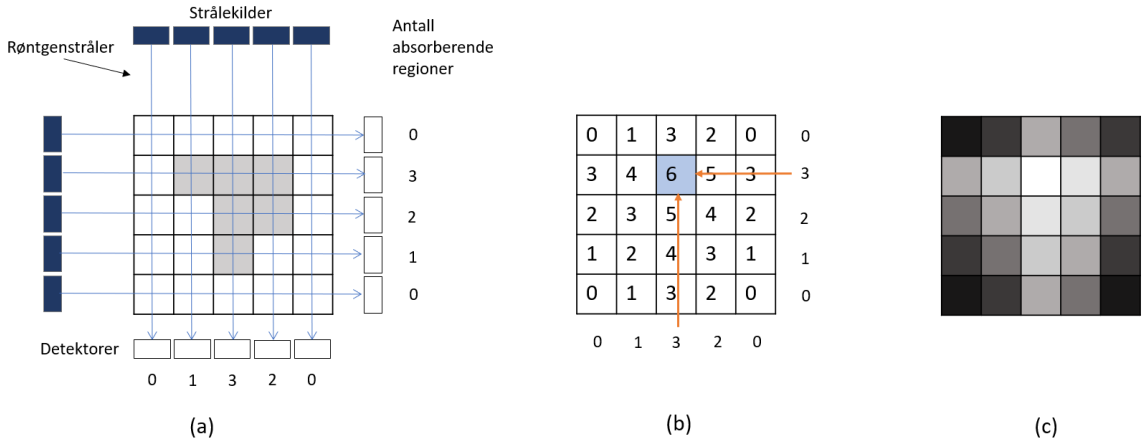
$$\mu = \mu_{fotoelektrisk} + \mu_{compton} + \mu_{pardannelse} \tag{2.1}$$

der μ er attenueringskoeffisienten til henholdsvis fotoelektrisk effekt, Comptonspredning og pardannelse [28]. Intensiteten til en transmittert røntgenstråle gjennom et materiale er gitt ved likningen

$$I_{trans} = I_0 e^{-\mu x} \tag{2.2}$$

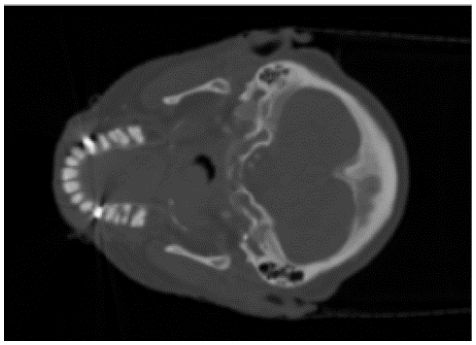
der I_0 er intensiteten til den innkomne røntgenstrålen, μ er den lineære attenuasjonskoeffisienten og x er tykkelsen på materiale [26]. I_{trans} vil da være de røntgenstrålene som har blitt transmittert gjennom kroppen og som detektoren registrerer.

CT-scanneren vil rekonstruere projeksjonene til et bilde. Metoden som brukes til denne rekonstruksjonen er komplisert, men ideen baserer seg på hvordan øynene våre oppfatter dybde. Ved å se ett objekt fra to retninger, kan man danne et bilde av hvordan objektet ser ut i tre dimensjoner. Det finnes flere ulike matematiske metoder for å gjøre dette, og en mye brukt metode er *filtered back projection.* Prinsippet bak *filtered back projection* er vist i Figur 2.5. Under en avbildning registrerer detektoren det som kalles *forward projection.* Her registrerer detektorene intensiteten, slik som forklart tidligere. Intensiteten til røntgenstrålen uten en pasient til stede er også kjent, og ut ifra dette kan summen av alle absorbsjonene langs en linje av røntgenstrålen beregnes. For å rekonstruere vevets røntgenabsorbsjon, brukes en prosess som kjent som *back projection.* I denne metoden blir hver overlappende region tildelt en absorbsjonsverdi basert på summen til hver av detektorenes registrerte absorberende regioner. Summeringen gjort i *back projection* er illustrert i Figur 2.5 b). Projeksjonsmatrisen kan vises som et bilde med gråskalaverdier [22]. *Back projection* gir alene ikke en nøyaktig gjengivelse av de originale absorbsjonsevnene, det kan for eksempel være regioner i bildet som ikke har riktig absorbsjon eller glatte strukturer kan være hakkete. En matematisk metode kalt filtrering kan brukes på projeksjonen etter *forward projection* for å fjerne artefakter i bildet rekonstruert fra *back projection* [22].



Figur 2.5: *Fremgangsmåten for filtered back projection med a) forward projection, her er de hvite feltene ikke-absorberende regioner og de grå feltene er absorberende regioner. b) back projection og c) det endelige bildet. Illustrasjon laget med inspirasjon fra figur i Introduction to Physics in Modern Medicine av S. Kane og B. Gelman (2020) [22].*

CT-bilder er rekonstruerte bilder, og er derfor bare digitale. I Figur 2.6 vises et CT-bilde fra hoderegionen. Pikslene («pixel elements») som utgjør snittbildet i kroppen, representerer et lite volum av vev kalt en voksel («voxel element»). Den minste oppløsningen som er praktisk mulig å oppnå i et CT-bilde er 1 mm. Det vil i teorien være mulig med en enda mindre oppløsning, men dette krever at sterkere røntgenstråler brukes over lengre tid [22].



Figur 2.6*: CT-bilde fra hoderegionen med CT-intensitetsvindu med W = 200 og L =70.*

CT-scannere representerer evnen til å absorbere røntgenstråler med enheten CT-nummer. Verdiene måles i Hounsfield units (HU). CT-nummeret kan beregnes fra likningen:

$$CTnummer = \frac{\mu_{vev} - \mu_{vann}}{\mu_{vann}} \times 100 \qquad (2.3)$$

hvor μ_{vev} er attenuasjonskoeffisienten til vevet og μ_{vann} er attenuasjonskoeffisienten til vann. Vann har et CT-nummer på null, luft har en verdi på -1000 HU, fettrikt vev vil ha en verdi på mindre enn null, mens de fleste andre vev vil ha en positiv verdi [22, 29]. De høyeste CT-

numrene kommer fra bein med høy tetthet, mens de laveste CT-numrene kommer fra områder i kropper med luft, som for eksempel lunger eller tarm [22].

CT-intensitetsvindu

CT-intensitetsvindu («CT-*windowing*») brukes for å justere intervallet til gråskalaen gjennom å manipulere CT-numrene [30]. Årsaken til at man ønsker å gjøre dette er for å kunne endre fremtredelsen til ønskede organer, eller for å skille ut strukturer. Kontrasten i bildet styres av vindu-bredden (W) og lysstyrken i bildet bestemmes fra vindu-senteret (L). Mykt vev i hode- og nakkeregionen har vanligvis vindu-verdier på for eksempel W = 300-400 HU og L = 20-60 HU, men dette kan variere mellom ulike institusjoner og leverandører [30].

Kontrast

Noen områder i kroppen kan være vanskelig å skille fra hverandre på både røntgen og CT, fordi de kan ha ganske like verdier av atomnummeret Z (antall protoner i atomkjernen) [22]. Dette kan spesielt være et problem i områder med mye mykt vev. For å forsterke synligheten til det ønskede vevet på røntgen og CT-bilder, kan man ta i bruk kontrastmidler. Et kontrastmiddel er en flytende forbindelse som gis intravenøst i en blodåre eller oralt. Kontrastvæsken blir værende i kroppen lenge nok til bildene er tatt, og skilles deretter naturlig ut av kroppen [22]. Barium (Z = 56) og jod (Z = 53) er stoffer som ofte brukes i kontrastmidler fordi de har høye atomnumre. Sannsynligheten for attenuering ved fotoelektrisk effekt øker med atomnummeret opphøyet i tredje, altså Z^3 [22]. Mykt vev består i hovedsak av atomer med lave Z-verdier som karbon (Z=6), nitrogen (Z=7), oksygen (Z=8) og hydrogen (Z=1). Kontraststoffene barium og jod vil derfor absorberer mer av røntgenstrålene med fotoelektrisk effekt enn hva annet mykt vev vanligvis ville gjort, og vil gjøre at vevet som har blitt injisert med kontrastvæske vil lyse opp på bildene [22].

2.3 Positronemisjonstomografi

Positronemisjonstomografi (PET) er en bildeteknikk som kan visualisere organenes funksjon ved hjelp av positronemisjon fra radioaktive nuklider [21]. Et slikt bilde er vist i Figur 2.11. Radioaktive nuklider er atomer med ustabile kjerner, og de nuklidene som brukes i PET emitterer betastråling i form av positroner. En atomkjerne består av to typer nukleoner; protoner og nøytroner. Atomnummeret til et atom er bestemt av antallet protoner i atomkjernen [31]. Protoner og nøytroner har omtrent samme masse, men de har ulike ladninger. Protoner har en positiv elektrisk ladning mens nøytronene er elektrisk nøytrale.

Atomer som har ulikt antall protoner og nøytroner i atomkjernen, kalles nuklider. Flere atomer kan ha likt antall protoner, men forskjellig antall nøytroner, disse kalles isotoper [22]. Bare spesifikke kombinasjoner av nøytroner og protoner gir stabile nuklider. De nuklidene som ikke er stabile vil desintegrere og frigjøre energi i form av radioaktiv stråling, og kalles radionuklider. Eksempler på radionuklider som brukes i PET er ^{11}C, ^{13}N, ^{15}O, ^{18}F. Disse har halveringstider på henholdsvis 20,5 min, 9,97 min, 122 sek og 110 min [20]. Halveringstiden til en radionuklide er tiden det tar til mengden nuklider i et radioaktivt stoff er halvert [22]. En fordel med en kort halveringstid er at det radioaktive stoffet ikke blir i kroppen over lang tid. Ulempen med en kort halveringstid er at de må produsert der de skal brukes [20].

2.3.1 Positronemisjon

Radionuklidene bindes til et molekyl gjennom en prosess kalt radiomerking, og injiseres i kroppen [22]. Molekylene som radiomerkes er ofte grunnstoffer det allerede finnes mye av i kroppens organiske forbindelser. Kreftceller har en høy metabolsk aktivitet og ved å radiomerke glukose med f.eks. fluor-18, så kan den metabolske aktiviteten kartlegges ut ifra fordelingen av den radioaktive glukosen [22].

Radionuklider som ikke er stabile fordi de har et overskudd av protoner kan desintegrere via emisjon av et positron:

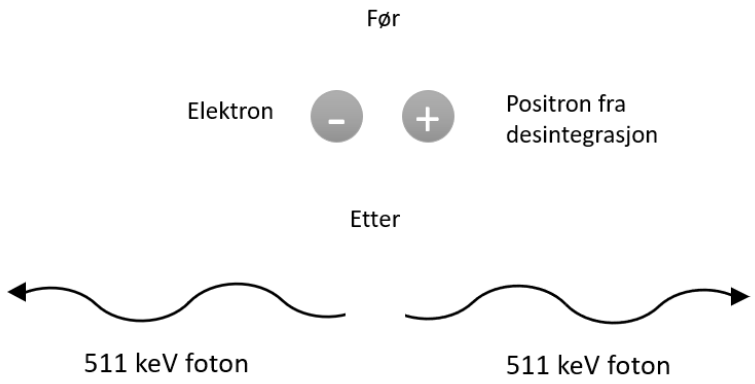
$$ {}^{A}_{Z}X_N \rightarrow {}_{Z-1}{}^{A}Y_{N+1} + e^+ + \nu + energi \tag{2.4} $$

der X er den opprinnelige nukliden med Z protoner, N nøytroner og A er massetallet. Y er datternukliden med ett mer nøytron, ett mindre proton og samme massetall. e^+ er positronet og ν er nøytrinoet [31]. Et positron er antipartikkelen til et elektron. Det innebærer at positronet har samme egenskaper om elektronet, bare motsatt elektrisk ladning [22]. Energi som frigjøres i desintegrasjonene er kinetisk energi som deles mellom datternukliden, positronet og nøytrinoet [32]. Positronet kan nå bevege seg en liten distanse før det interagerer med et elektron som er bundet til et atom i kroppen [20].

2.3.2 Annihilering

Elektroner og positroner vil ha en sterk tiltrekning mot hverandre grunnet deres motsatte elektriske ladninger, og vil danne et positronium. Dette er en ustabil tilstand, og elektron-positronparet vil etter kort tid gjennomgå en annihilering [22]. Annihileringen produserer to fotoner, og for at bevaringsloven for energi og bevegelsesmengde skal gjelde, må fotonene ha en energi på nøyaktig 511 keV hver og bevege seg i motsatt retning av hverandre, illustrert i

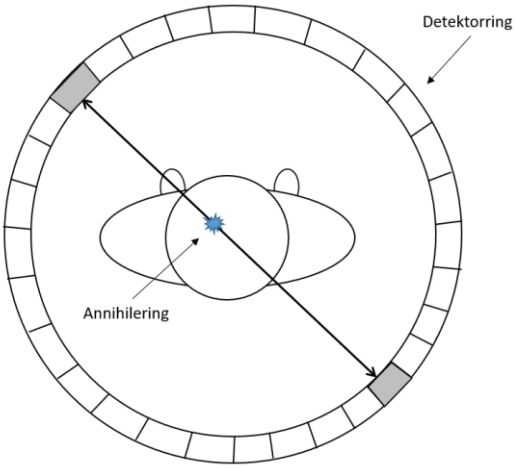
Figur 2.7. Ingen partikler med masse forblir etter dette sammenstøtet [31]. Fotonene som dannes i annihileringen har høy energi og kan dermed bevege seg ut av kroppen. Det er disse fotonene PET-skanneren registrerer [22].



***Figur 2.7**: Annihilering. Elektroner og positroner tiltrekker hverandre pga. deres motsatte ladning. Sammenstøtet resulterer i to fotoner som beveger seg i motsatt retning av hverandre. Illustrasjon laget med inspirasjon fra figur i Introduction to Physics in Modern Medicine av S. Kane og B. Gelman (2020) [22].*

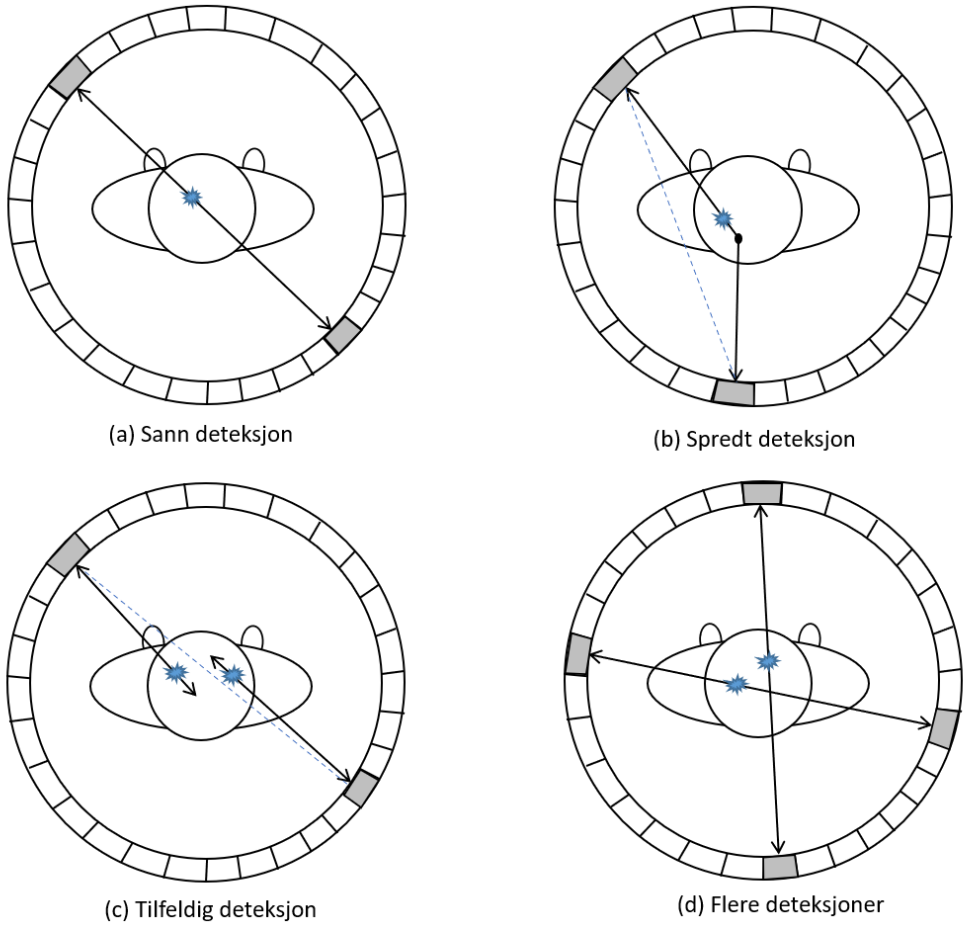
2.3.3 PET-skanneren

Etter at pasienten har fått den radioaktive markøren injisert, må det ventes rundt en time slik at stoffet får fordelt seg i kroppen [5]. I en PET-skanner er pasienten omringet av an ring med detektorer. Fotoner fra annihileringsprosessen som forlater kroppen blir registrert av detektorene, som vist i Figur 2.8**.** Fotonene som sendes ut i motsatt retting av hverandre, vil bli registrert av detektorer på hver sin side av ringen. Denne hendelsen må skje samtidig eller innen en tid på 10-25 ns fra hverandre for at den skal bli registrert [20]. Det kan trekkes en linje mellom disse to detektorene, som vil gå gjennom punktet der annihileringen skjedde. Linjen kalles *Line of Response* (LOR). Antall hendelser innenfor hver LOR telles og brukes som data til rekonstruering av PET-bildet [22]. Siden positronet som er med i annihileringsprosessen har en kort rekkevidde i kroppen (mindre enn 1 mm), vil dette punktet ikke være langt unna punktet der positronemisjonen fant sted. Det er da mulig å finne ut av hvor radionuklidet befant seg i kroppen på tidspunktet for positronemisjonen [20, 31]



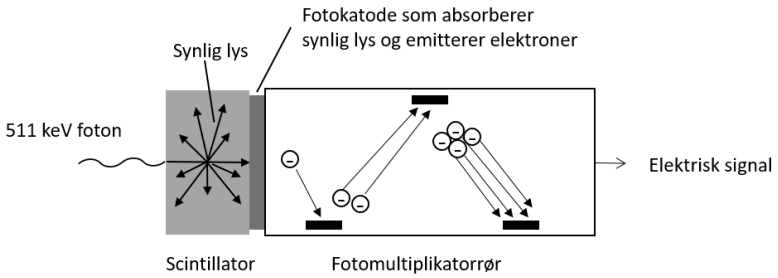
***Figur 2.8**: Deteksjon i en PET-skanner. To detektorer plassert 180 grader fra hverandre, observerer et foton tilnærmet samtidig. Illustrasjon laget med inspirasjon fra figur i Introduction to Physics in Modern Medicine av S. Kane og B. Gelman (2020) [22].*

Det finnes fire forskjellig type deteksjoner i en PET-scanner, illustrert i Figur 2.9 [31]. En sann deteksjon innebærer at begge fotonene når detektorene og registreres på riktig plass (Figur 2.9 a). En spredt deteksjon (Figur 2.9 b) vil si at et eller begge fotonene endrer retning, etter å ha vekselvirket med en partikkel i kroppen. Dette vil skape en falsk LOR, som ikke angir rett punkt for annihileringen. Av de fotonene som blir spredt, er det bare en liten andel som blir detektert. Tapet av sanne deteksjoner grunnet attenuering har flere negative sider. Det kan føre til økt støy i bildet og målinger av radioaktiviteten vil heller ikke være nøyaktige. En korreksjon av attenueringen vil kunne ta høyde for disse artefaktene. En attenueringskorreksjon i PET er en måling eller en gjetning av sannsynligheten for attenuering langs en linje. Når PET og CT kombineres brukes den målte attenueringen fra CT-skanningen som attenueringskoeffisient [31]. Under en tilfeldig deteksjon (Figur 2.9 c) vil to fotoner fra ulike annihileringer treffe detektorene samtidig, og skape en falsk LOR. Det kan også bli registrert flere deteksjoner samtidig (Figur 2.9 d), noe som gjør det vanskelig å bestemme hvilke deteksjoner som kommer fra hvilke annihileringer [31].



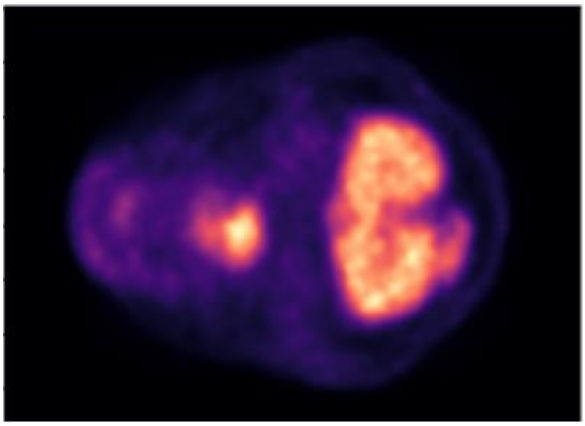
Figur 2.9: *Fire hovedtyper av deteksjoner som kan oppstå i en PET-skanner. Illustrasjon laget etter inspirasjon fra figur i PET – Physics, instrumentation and scanners av M. E. Phelps (2006) [32].*

Detektorene som brukes i PET-skannere består av krystalliserte scintillatorer og fotomultiplikatorrør. Fotonene interagerer med krystallene og produserer synlig lys. Noe av lyset blir fanget opp i fotomultiplikatorrørene og konvertert til et elektrisk signal, slik som vises i Figur 2.10 [31].



Figur 2.10: *Illustrasjon av hvordan et foton gjøres om til et elektrisk signal i en PET-detektor. Illustrasjon laget med inspirasjon fra figur i Introduction to Physics in Modern Medicine av S. Kane og B. Gelman (2020) [22].*

Oppløsningen til PET-bildene ligger på rundt 3-5 mm [32]. Oppløsningen til PET-bildene påvirkes av de få millimeterne som positronene beveger seg før de annihilerer. Det vil være en uskarphet i bildene som skyldes positronets gjennomsnittlige rekkevidde [31]. Figur 2.11 vises et eksempel på et PET-bilde fra hoderegionen. En annen begrensning for oppløsningen til PET-bilder er at fotonene ikke alltid har nøyaktig 180 grader mellom seg når de emitteres. Dette er et problem som har en større effekt for detektorringer som har en stor diameter. En tredje faktor som kan påvirke oppløsningen er størrelsen på detektorene. En mindre størrelse på detektorene vil gjøre at tykkelsen på LOR vil bli smalere, slik at området hvor radionukliden befant seg på ved emisjonstidspunktet begrenses [31]. PET-bilder vil derfor ha dårligere oppløsning enn CT- og MR-bilder.



Figur 2.11: *PET-bilde fra hoderegionen. Områder med høy metabolsk rate lyser opp.*

2.3.4 Fluorodeoksyglukose

Fluorodeoksyglukose ([^{18}F] 2-fluoro-2-deoxy-d-glucose, ^{18}F-FDG eller FDG) er den forbindelsen som er mest brukt som radioaktiv markør innen PET-avbildning. Denne forbindelsen består av glukose hvor en hydroksylgruppe er erstattet med fluor-18 [33]. Fluor-18 har en halveringstid på nesten to timer (110min) og 97 % av den radioaktive strålingen kommer fra positronemisjon [31]. I likhet med mange av de andre ionene som blir brukt i radioaktive markører, blir fluor-18 produsert i en partikkelakselerator kalt en syklotron. Her blir oksygen-18-beriket vann bombardert med protoner. Den relativt lange halveringstiden til fluor-18 gjør at den ikke trengs å produseres der PET-avbildningen skjer [22, 33].

Kreftceller har en høy metabolsk rate sammenlignet med vanlige celler, og vil av den grunn ta opp mer av den glukoseliknende radioaktive markøren [34]. På denne måten kan områder med høy metabolsk rate i kroppen kartlegges, slik som vist i Figur 2.11.

2.3.5 SUV

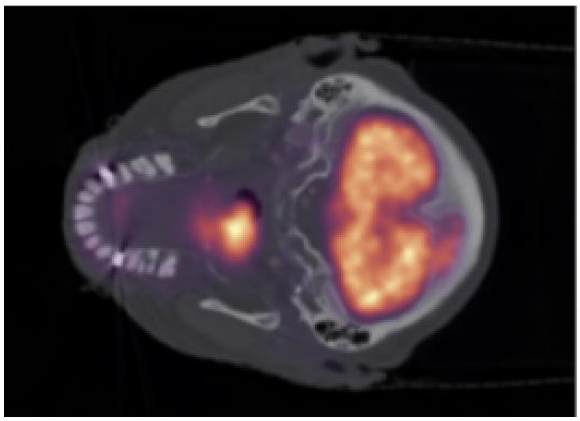
Opptaket av den radioaktive markøren kan variere mellom hver PET-avbildning. Mengden injisert FDG og vekten til pasienten er de to faktorene som i størst grad påvirker denne variasjonen [31, 35]. *Standardized Uptake Value* (SUV) brukes derfor ofte som et relativt mål på FDG opptak:

$$SUV = \frac{r}{(a/w)} \tag{2.5}$$

hvor *r* er konsentrasjonen av radioaktiv aktivitet [kBq/mL] i et område målt av PET-skanneren, *a* er aktiviteten til injisert FDG [kBq] og *w* er vekten til pasienten målt i gram [35].

2.4 PET/CT

Å lokalisere hvor i kroppen den radioaktive aktiviteten er i et PET-bilde kan være vanskelig, mye fordi de genererte bildene gir lite anatomisk informasjon. Et PET-bilde har en oppløsning på mellom 3-5 mm [32] som er dårlig sammenlignet med et CT-bilde, hvor oppløsningen kan være helt ned til 1 mm [22]. En kombinasjon av PET og CT har flere fordeler. CT-bildene gir et mer detaljert bilde av kroppens anatomi, og kan brukes for å tolke PET-bildene. Her brukes PET-bildene til å finne de områdene i kroppen som har en forhøyet metabolsk rate, og CT-bildene viser nøyaktig hvor i kroppen. I Figur 2.12 vises et eksempel på et PET/CT-bilde fra hoderegionen. Kombinasjonen kan også gjøre diagnoser mer nøyaktige. PET kan skille mellom godartede og ondartede tumorer som ser like ut på CT-bildene, noe som reduserer antallet falske positive. CT kan finne områder med kreftceller som ikke aktivt metaboliserer glukose, noe som kan bidra til å redusere antall falske negative [22]. I en PET/CT-skann blir først en CT-skann utført etterfulgt av en PET-skann. Fra CT-skannen brukes informasjon om vevstettheten for å beregne attenueringskorreksjonen for PET-bildene [32]. Dette gjør at en PET/CT-skann er tidsbesparende samtidig som den genererer et bilde med mer anatomisk informasjon.

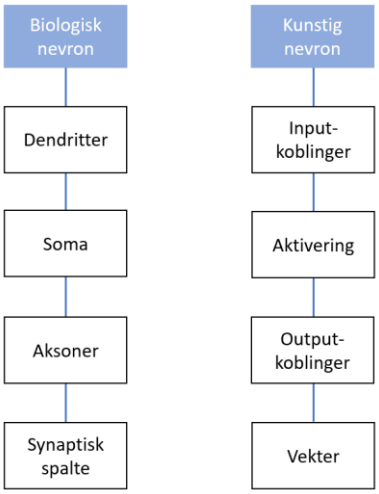


***Figur 2.12**: PET/CT-bilde fra hoderegionen. Her vises både områdene med høy metabolsk rate og et godt bilde av kroppens anatomi.*

Kapittel 3: Maskinlæring

3.1 Prinsipper innen maskinlæring

Maskinlæring er et begrep under kunstig intelligens som omhandler selvlærende algoritmer som kan trekke ut informasjon og mønstre fra data, og gjøre beslutninger basert på dette. Metoden algoritmene bruker for å lære er inspirert av hvordan et biologisk nevron sender nervesignaler i hjernen [36]. En skjematisk sammenlikning av et biologisk nevron og et kunstig nevron er vist i Figur 3.1.



Figur 3.1: *Skjematisk sammenlikning mellom et biologisk nevron og et kunstig nevron. Illustrasjon laget etter inspirasjon fra figur i Python Deep Learning av V. Zocca et al. (2017) [37].*

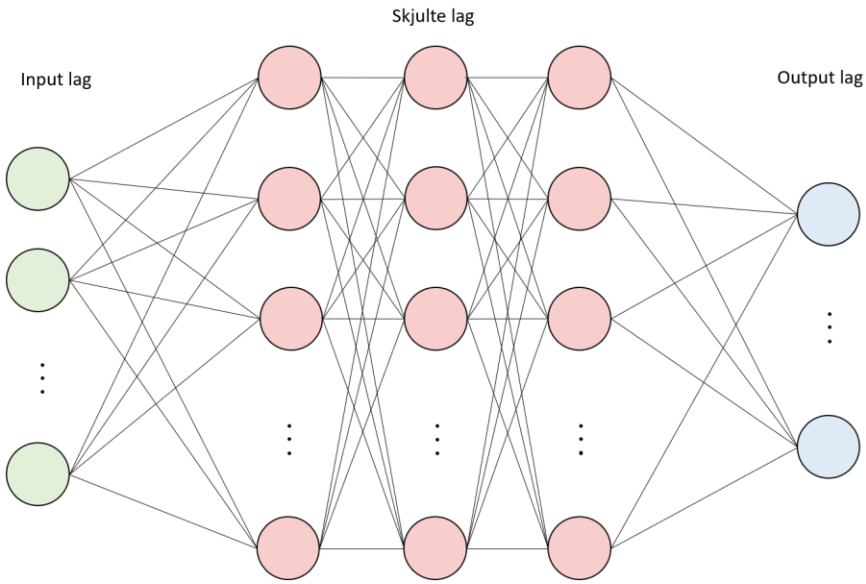
Maskinlæring kan deles opp i tre hovedtyper: veiledet læring, ikke-veiledet læring og forsterket læring [37]. I veiledet læring brukes data som har kjent klassifisering til å trene maskinlæringsmodellen. Ytelsen til modellen kan beregnes ut ifra hvor stor andel av de predikerte klassifiseringene som er riktig. Ikke-veiledet læring har data som ikke har en kjent klassifisering og det er opp til maskinlæringsmodellen å finne mønstre i dataen. Klyngeanalyse («Clustering») er et eksempel på ikke-veiledet trening som vil prøve å dele dataen opp i undergrupper [37]. I forsterket læring er målet å utvikle et system som forbedrer ytelsen basert på tilbakemeldinger. Forsterket læring er ofte brukt for å lære datamaskiner ulike spill. Her vil modellen få tilbakemelding basert på utfallet av spillet [37].

Prediksjonene gjort av en maskinlæringsmodell kan være enten kvalitative eller kvantitative. En kvalitativ verdi vil bli satt til en av *N* ulike kategorier. En klassifiseringsmodell vil ha kvalitative verdier som output. Dette kan være f.eks. kjønn (mann eller kvinne) eller resultatet av en prøve (karakterskala). En kvantitativ verdi er en numerisk verdi. Dette kan for eksempel være forventet levealder. Et regresjonsproblem vil ha kvalitative verdier som output [38].

Dyp læring er en underkategori av maskinlæring hvor modellene består av nettverk med flere hierarkiske lag [9]. I motsetning til enklere maskinlæringsmetoder som lærer seg å kjenne igjen et objekt, vil dype nevrale nettverk lære seg viktige egenskaper som er unike for de ulike objektene. Dype nevrale nettverk kan finne komplekse egenskaper og representasjoner av dataen. Enklere maskinlæringsmodeller med kun ett skjult lag ser kun på en representasjon av dataen. Disse modellene er derfor avhengig av at denne representasjonen inneholder egenskaper som er tydelig relatert til det forventede output. De enklere maskinlæringsmodellene vil av denne grunn også ikke være like gode på å lære seg komplekse mønstre i dataen [37]. I enklere maskinlæringsalgoritmer må brukeren selv bestemme hvilken eller hvilke egenskaper som representerer dataen best, noe som er et tidkrevende arbeid. Dyplæringsalgoritmer har derimot fordelen med at de automatisk kan gjenkjenne egenskaper som er relevante for prediksjonene [37]. De neste sidene vil ta for seg en introduksjon av nevrale nettverk og hvordan disse brukes i bildeprosessering.

3.1.1 Nevrale nettverk

Et nevralt nettverk består av et inputlag, et eller flere skjulte lag og et outputlag, slik som vist i Figur 3.2. Hvert lag består av prosesseringsenheter kalt noder. Nodene tar inn verdier og prosesserer denne dataen før den sendes videre til nodene i neste lag. Koblingen mellom nodene kalles vekter, og det er disse som kobler lagene sammen. Vektene kan variere i styrke, og det er dette som avgjør hvordan dataen blir prosessert [37]. Før treningsprosessen blir vektene satt til tilfeldige små tall. Dersom alle nodene er koblet sammen med alle nodene i neste lag, er laget et fullt koblet lag [36], slik som lagene i nettverket i Figur 3.2.



Figur 3.2: *Illustrasjon av et nevralt nettverk med tre skjulte lag. Sirklene representerer nodene og linjene mellom nodene representerer koblingene/vektene mellom nodene. Illustrasjon laget etter inspirasjon fra figur i Python Deep Learning av V. Zocca et al. (2017) [37].*

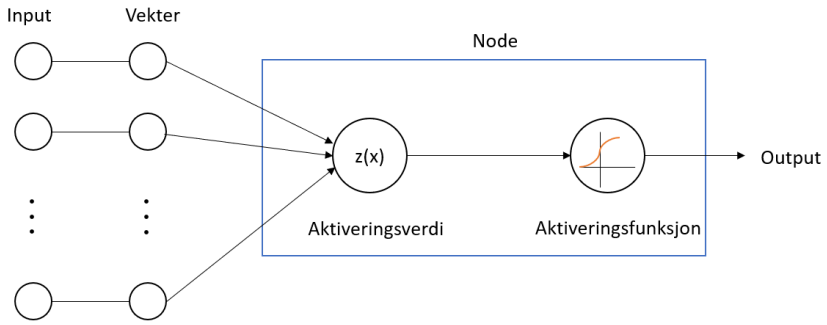
Hver node vil ta inn verdier kalt aktiveringsverdier, som er en summasjon av outputet fra nodene i det forrige laget. Dersom aktiveringsverdien er over en gitt grense, vil noden aktiveres. Aktiveringsverdien $z(\boldsymbol{x})$ er gitt ved

$$z(\boldsymbol{x}) = \sum_i w_i \, x_i \tag{3.1}$$

hvor x_i er verdien hver node i tar inn og w_i er styrken til koblingene mellom lagene [37]. Dersom $\mathbf{w}$ og $\mathbf{x}$ er vektorer er aktiveringsverdien skalarproduktet mellom disse to vektorene. $\mathbf{x} \cdot \mathbf{w} = 0$ definerer et hyperplan i $\mathbf{R}^d$, hvor d er dimensjonen til $\mathbf{x}$. Alle vektorer som oppfyller kravet $\mathbf{x} \cdot \mathbf{w} > 0$ eller $\mathbf{x} \cdot \mathbf{w} < 0$ vil ligge på hver sin side av hyperplanet definert av $\mathbf{w}$. Dette viser hvordan hver enkelt node kan virke som en lineær klassifiserer som aktiveres når inputet er over en gitt grense [37]. Ved å legge til en bias b til aktiveringsverdien vil flytte hyperplanet vekk fra origo. Aktiveringsverdien får da følgende likning:

$$z(\boldsymbol{x}) = \sum_i w_i x_i + \, b \tag{3.2}$$

Dataprosesseringen som skjer i hver node er illustrert i Figur 3.3. Her blir outputet fra en node bestemt av en aktiveringsfunksjon som tar aktiveringsverdiene som input.



Figur 3.3: *Illustrasjon av prosessen som skjer i en node. Inputverdiene blir multiplisert med tilhørende vekter før de blir summert opp til aktiveringsverdien. Videre blir aktiveringsverdien sendt til aktiveringsfunksjonen som bestemmer outputet fra noden. Illustrasjon laget etter inspirasjon fra figur i Python Deep Learning av V. Zocca et al. (2017) [37].*

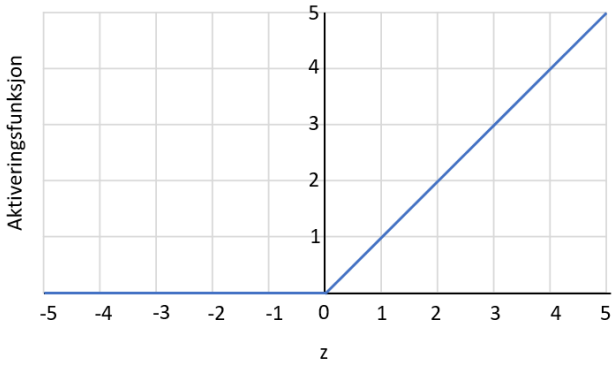
Den enkleste aktiveringsfunksjonen som brukes er identitetsfunksjonen, $\varphi(z) = z$. Dette er en lineær funksjon hvor outputet til noden er det samme som aktiveringsverdien [37]. En ikke-lineær aktiveringsfunksjon er terskelfunksjonen i likning 3.3. Denne funksjonen aktiverer en node dersom aktiveringsverdien er over en gitt terskel, og er gitt ved:

$$\phi(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases} \tag{3.3}$$

En kombinasjon av identitetsfunksjonen og terskelfunksjonen er *Rectified Linear Unit* eller ReLU:

$$\phi(z) = \begin{cases} z, & z \geq 0 \\ 0, & z < 0 \end{cases} \tag{3.4}$$

ReLU vil kun sende en aktiveringsverdi videre til neste lag dersom aktiveringsverdien er positiv, slik som vises i Figur 3.4. En fordel med dette er at funksjonene introduserer ikke-linearitet for aktiveringen [36].



Figur 3.4: *ReLU aktiveringsfunksjon hvor x-aksen representerer aktiveringsverdiene og y-aksen representerer aktiveringsfunksjonen.*

En aktiveringsfunksjon som brukes mye er sigmoidfunksjonen, hvor outputet er en verdi mellom 0 og 1, som kan tolkes som sannsynligheten for at noden aktiveres [37]. Sigmoidfunksjonen er definert som:

$$\phi(z) = \frac{1}{1 + e^{-z}} \tag{3.5}$$

Hvilken aktiveringsfunksjon som velges, avhenger av problemet som skal løses. Identitetsfunksjonen og terskelfunksjonen brukes blant annet til binære klassifiseringsproblemer. Sigmoidfunksjonen brukes til klassifiseringsproblemer der det er mer enn to klasser. I et nevralt nettverk har som regel hver node innad i ett lag den samme aktiveringsfunksjonen, mens de ulike lagene kan ha forskjellige aktiveringsfunksjoner [37].

3.1.2 Optimalisering av modellen

Koblingene mellom nodene og vektene i et nettverk kan sammenliknes med koblingene mellom nevroner i hjernen: de koblingene som blir ansett som viktige og brukes ofte, vil bli styrket. Mens koblingene som er mindre viktig og brukes mindre, vil bli svekket [37]. Vektene i et nettverk settes først til små tilfeldige tall. Signalene itereres gjennom nettverket for å finne de best tilpassede vektene, og på denne måten øke ytelsen til modellen [36].

Målet med å trene et nevralt nettverk er å oppdatere vektene slik at feilen i prediksjoner gjort av nettverket blir minimalisert. For å oppnå dette må oppdateringen av vektene gjøres med et mål om å minimalisere tapsfunksjonen [37]. Denne kalles også kostfunksjonen eller errorfunksjonen [36]. Tapsfunksjonen *J* kan være en hvilken som helst funksjon definert som en funksjon av vektene, og angir hvor godt nettverket predikerer ved å beregne forskjellen mellom predikert og sann verdi [9]. En mye brukt tapsfunksjon er *sum of squared error* (SSE):

$$J(\boldsymbol{w}) = \frac{1}{2}\sum_{i}\left(y^{(i)} - \phi(z^{(i)})\right)^2 \tag{3.6}$$

hvor $y^{(i)}$ er den virkelige verdien og $\phi(z^{(i)})$ er den predikerte verdien for eksemplar *i* [36]. Kryssentropi-tapsfunksjonen brukes ofte til klassifiseringsproblemer, og for et problem med *n* klasser er den definert som:

$$J(\boldsymbol{w}) = -\sum_{i=1}^{n} y^{(i)} ln\left(p^{i}\right) \tag{3.7}$$

hvor $y^{(i)}$ er den virkelige verdien og p^i er sannsynligheten for at eksemplar *i* tilhører klasse 1 med de gitte vektene. Dersom kryssentropi-tapsfunksjonen skal brukes er det viktig å bruke en aktiveringsfunksjon som gir ut verdier mellom 0 og 1, og som kan tolkes som en sannsynlighetsfordeling, slik som sigmoid funksjonen [37]. Hvilken tapsfunksjon som brukes er avhengig av problemet og antall klasser.

For datasett med ubalanserte klasser, hvor man har mange flere observasjoner av en klasse enn andre, vil ikke alle tapsfunksjoner være gunstige å bruke fordi de kan være sensitive mot ubalanserte klasser. Milletari et al. [39] har definert en tapsfunksjon som er basert på Dice-score og tar hensyn til ubalanse mellom klassene i datasettet. Dice Loss er et spesialtilfelle av FBetaLoss hvor beta lik 1. For et binært klassifiseringsproblem er Dice Loss definert som:

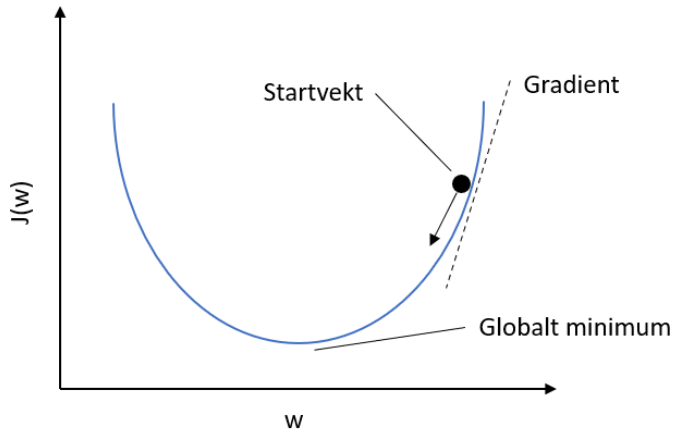
$$J_{DiceLoss} = 1 - \frac{2SP}{\sum_i p_i^2 + \sum_i g_i^2} \tag{3.8}$$

hvor *SP* er antallet sanne positive, p_i er den predikerte sannsynligheten for at piksel *i* tilhører positiv klasse og g_i er den sanne verdien til piksel *i* [40]. I *deoxys* er Dice loss implementert ved å bruke binaryFBetaLoss med beta lik 1.

Vektene i nettverket blir oppdatert iterativt, og målet er å finne vekter som minimaliserer tapsfunksjonen. Gradient-nedstigning («Gradient descent») er en mye brukt metode for oppdatering av vektene, illustrert i Figur 3.5. Som forklart tidligere er tapsfunksjonen en funksjon av vektene i nettverket. Ved å beregne gradienten til tapsfunksjonen, kan man finne retningen som har brattest stigning [36]. Dersom vektene oppdateres i motsatt retning av gradienten, vil neste iterasjon ha et lavere tap. Vektene oppdateres etter følgende likning:

$$\boldsymbol{w} = \boldsymbol{w} - \eta \nabla J(\boldsymbol{w}) \tag{3.9}$$

hvor η er læringsraten, og $\nabla J(\boldsymbol{w})$ er gradienten til tapsfunksjonen [36]. Ideelt sett oppdateres vektene helt til man når et globalt minimumstap. Læringsraten angir steglengden som man beveger seg i motsatt retning av gradienten med, og det er derfor viktig å velge en rimelig verdi. En for liten læringsrate kan føre til unødvendig mange iterasjoner og minimumet som nås kan være et lokalt minimum. På den andre siden kan en for stor læringsrate føre til at man ikke finner noe minimum, og at punktet bare er et tilfeldig punkt langs kurven [36].



***Figur 3.5**: Illustrasjon av gradient-nedstigning. Illustrasjon laget etter inspirasjon fra figur i Python Machine Learning av S. Raschka og V. Mirjalili (2019) [36].*

Momentum er en metode som kan hjelpe å «dytte» gradient-nedstigning i riktig retning og dermed føre til konvergens raskere [37]. Algoritmen øker hastigheten til optimaliseringen i områdene langs kurven hvor gradienten ikke endrer retning, og senker hastigheten til optimaliseringen i områdene hvor gradienten er alternerende. Områdene med alternerende gradient korresponder til områder med minimum [37].

En adaptiv optimaliseringsalgoritme kalt *Adaptive moment estimation* eller Adam, ble presentert av Kingma og Ba [41]. Adam er en allsidig optimaliseringsalgoritme som kan brukes på store maskinlæringsproblemer med mange dimensjoner, og har dermed blitt en populær algoritme for nevrale nettverk [41]. Adam inkluderer en modifisering av læringsraten for hver vekt, i tillegg til momentum. Optimaliseringsmetoden bruker færre iterasjoner gjennom nettverket for å få tapsverdien til å konvergere, og er dermed beregningsmessig mindre kostbar sammenliknet med gradient-nedstigning. Algoritmen bak Adam som oppdaterer vektene bruker det estimerte gjennomsnittet av tidligere gradienter, den usentrerte variansen av gradienten, en støyparameter samt læringsraten [41, 42].

I et nevralt nettverk med kun et lag, kan metoden som brukes for optimaliseringen av vektene enkelt forståes. Her kan vektene oppdateres samtidig for å minimere tapsfunksjonen. For nevrale nettverk med flere skjulte lag, kan denne metoden kun brukes på vektene som kobler det siste skjulte laget sammen med outputlaget. Dette kan gjøres fordi vi vet hva vi vil at outputlaget skal være. Metoden kan derimot ikke brukes på de resterende skjulte lagene, siden vi ikke vet hva verdiene til nodene burde være [37]. Løsningen på dette problemet vil være å bruke en metode som kalles tilbakepropagering («Backpropagation»). Tilbakepropagering begynner med den siste tapsverdien og jobber seg bakover i nettverket og bruker

kjerneregelen for å beregne bidraget fra hver parameter til tapsverdien [9]. Den deriverte av aktiveringsfunksjonen brukes i tilbakepropagering. For enkelte aktiveringsfunksjoner vil den deriverte av funksjonen avta når aktiveringsverdien blir stor. Dette fører til at oppdateringen av vektene vil skje sakte, ettersom gradienten kan være tilnærmet lik null. Dette er problemet som kalles *vanishing gradient problem*. I dype nevrale nettverk vil det derfor være en fordel å bruke ReLU som aktiveringsfunksjon i de skjulte lagene, ettersom den deriverte til ReLU alltid er 1 for positive inputverdier [36].

3.1.3 Splitting av datasettet: Trening, validering og testsett

Treningen av et nevralt nettverk er prosessen hvor nettverket optimaliseres, og det er under denne prosessen modellen lærer relevante mønstre basert på inputdataen [36]. Nettverkets kapasitet angir hvor mye et nettverk kan lære, og dette er avhengig av antall noder og antall lag i modellen [36]. Datasettet deles vanligvis i tre deler. Under treningsprosessen brukes treningssettet, og det er i tillegg nødvendig med et valideringssett og et testsett.

Valideringssettet brukes til å evaluere modellen etter treningsprosessen. Her valideres verdiene til vektene ved å observere hvordan modellen yter på ny og usett data. Dersom ytelsen til modellen ikke oppnår ønsket resultat på valideringssettet, kan man gå tilbake til treningen for å endre vektene. Når modellen oppnår ønsket resultat på valideringssettet, brukes testsettet som en siste evaluering på modellen, og inneholder kun usette prøver [36].

Når datasettet deles i trening, validering og treningssett bør de deles slik at de ulike settene representerer mangfoldet i dataen på best mulig måte. Dette innebærer at de har samme klasseproporsjoner som det originale datasettet, altså at forholdet mellom de ulike klassene er likt i de ulike delene som i det originale datasettet [9]. En trening, validering og testsplitt som representerer dataen på en god måte vil i tillegg føre til en mer generell modell som kan minske risikoen for overtilpasning [36].

3.1.4 Overtilpasning

Et grunnleggende problem innen maskinlæring er å finne en fin balanse mellom optimalisering og generalisering. Optimalisering refererer til prosessen med å tilpasse modellen for å få den beste ytelsen på treningsdataen. Generalisering er hvor godt den trente modellen yter på usett data. I begynnelsen av treningsprosessen er det en sammenheng mellom generalisering og optimalisering. Et lite tap på treningsdataen vil også gi et lavt tap på testdataen. På dette stadiet er det fortsatt relevante mønstre for modellen å lære, og det sies at

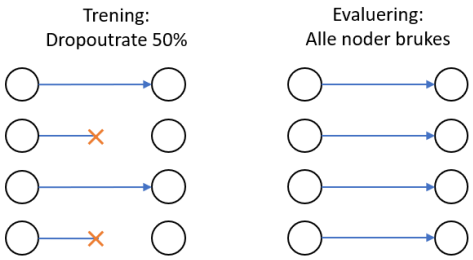
modellen undertilpasser. Etter hvert vil modellen bli overtilpasset. En overtilpasset modell har en høy ytelse på treningsdataen og en mye lavere på den usette dataen i validering- og testsettet. Modellen har da lært alle mønstrene som er spesifikke for treningsdataen, men irrelevante eller misvisende for usett data [9].

Den beste løsningen for å forhindre at en modell skal lære seg irrevante eller misvisende mønstre fra treningsdataen, er å skaffe mer treningsdata. En modell som trener på mer data vil naturlig generalisere bedre [9]. En metode som brukes for å skaffe mer treningsdata er bildeaugmentering. Her blir den allerede eksisterende treningsdataen transformert for å skape nye variasjoner av dataen [36]. Trening med mer data er ikke alltid mulig, og da finnes det andre metoder for å forhindre overtilpasning. En løsning kan være å modifisere mengden av informasjon som modellen kan lagre eller legge til begrensninger på hva slags informasjon som kan lagres. Dette gjør at modellen bare kan memorere mindre antall mønstre, og tvinges til å lagre bare de viktigste. Denne metoden for å håndtere overtilpasning kalles regularisering. Regularisering kan gjøres på ulike måter, enten ved å redusere størrelsen på nettverket, ved regularisering av vektene eller ved dropout [9].

Å redusere størrelsen på nettverket er den enkleste metoden for regularisering. Som forklart tidligere er kapasiteten til nettverket avhengig av antall lag i modellen og antall noder i lagene. Justeringen av modellens kapasitet vil være et kompromiss mellom en overtilpasset og undertilpasset modell [9].

En annen metode er regularisering av vektene. Konseptet bak vektregularisering er å gi begrensninger til vekter som har ekstremverdier. Dette fører til en mindre kompleks modell, hvor vektene vil ha en mer generell fordeling. To metoder som brukes til vektregularisering er L1- og L2-regularisering. Begrensningen som adderes til tapsfunksjonen i L1-regularisering er proporsjonal med absoluttverdien til vektkoeffisientene. I L2-regularisering er begrensningen proporsjonal med kvadratet av verdien av vektkoeffisienten [36].

Dropout er også en effektiv metode for regularisering av nevrale nettverk. I denne metoden blir en andel av tilfeldig valgte noder droppet i hver iterasjon, vist i Figur 3.6. Dropoutraten er andelen noder som blir droppet, og den ligger vanligvis på mellom 0,2 og 0,5 [9]. Dropout gjøres kun under treningsprosessen, og ikke under validering- og testprosessen. For å kompensere for at det er flere aktive noder under validerings- og testprosessen enn ved treningsprosessen, blir outputet fra hvert lag i validerings- og treningsprosessen skalert med en faktor lik dropoutraten [9].



***Figur 3.6**: Illustrasjon som viser konseptet dropout under trening og evaluering av modellen. Illustrasjon laget etter inspirasjon fra figur i Python Machine Learning av S. Raschka og V. Mirjalili (2019) [36].*

Antall oppdateringer av vektene i et dypt nevralt nettverk avhenger av batchstørrelsen («batch size») siden vektene oppdateres etter hvert batch [36]. Batchstørrelsen er hvor mange eksemplarer som nettverket bruker for å oppdatere vektene. En stor batchstørrelse vil gi en mer generell og mindre overtilpasset modell. Dersom batchstørrelsen reduseres vil modellen kunne finne og lære seg mønstre og egenskaper som ikke er relevante for prediksjonen, og dermed gi en mer overtilpasset modell.

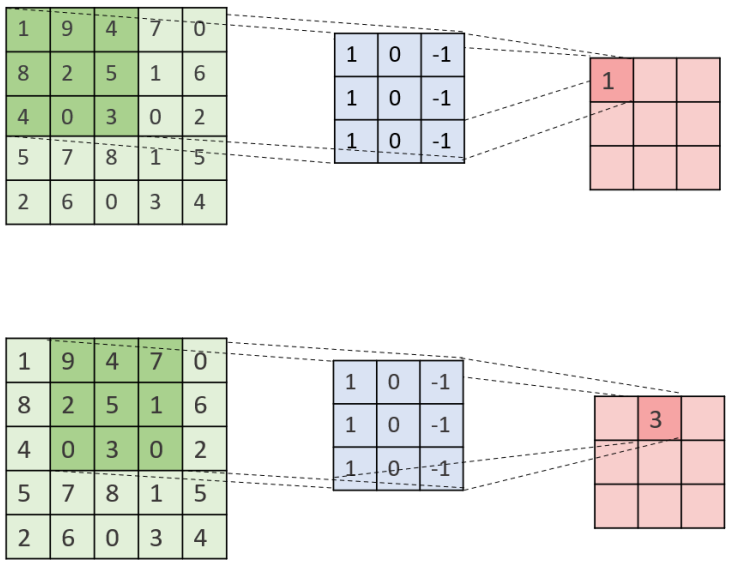
3.2 Klassifisering av bilder

Konvolusjonsnettverk («Convolutional neural networks», CNNs) er dype nevrale nettverk som brukes til å klassifisere bilder. CNNs er inspirert av hvordan hjernen prosesserer visuell informasjon ved å hierarkisk prosessere informasjonen på ulike nivåer [37]. Det er vanligvis tre ulike typer lag i CNN-arkitekturer: konvolusjonslag, samlelag og fullt koblede lag [36]. Nodene i et konvolusjonslag får bare input fra en liten andel nabo-noder i forrige lag, i motsetning til nodene i fullt koblede lag hvor hver node får input fra alle nodene i forrige lag. Disse nabo-nodene korresponderer til nabopiksler i bildet. På denne måten vil CNNs med konvolusjonslag redusere antall parametere som trengs og følgelig hjelpe for å unngå overtilpasning [37].

3.2.1 Konvolusjoner

Et konvolusjonslag kan sees på som et filter som kan utheve enkelte egenskaper i bilder, som kan være nyttige for å klassifisere bildet [37]. Filteret er en todimensjonal matrise og har vanligvis en størrelse på 3×3 eller 5×5. Filteret består av vekter som beveges over bildet. Filteret kan beveges over bildet med et hopp på en eller flere piksler, og hvor mange piksler som filteret beveges refereres til som steglengde. I hver filterposisjon blir det fortatt en transformasjon via vektmatrisen [37], vist i Figur 3.7. Et konvolusjonslag opererer på tredimensjonale inputbilder som har en høyde, bredde og dybde. For et RGB-bilde er dybden

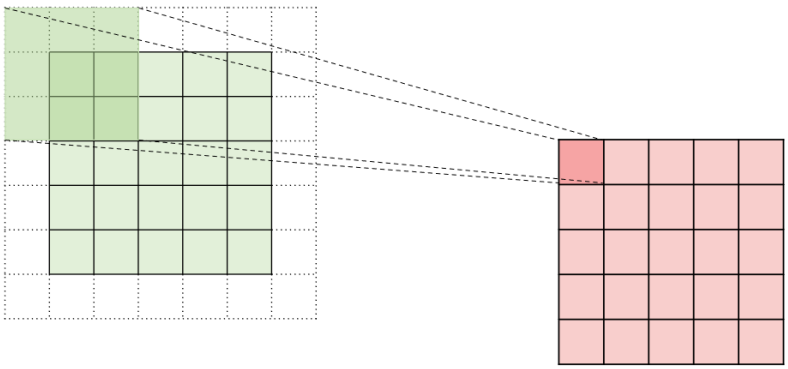
3, siden bildet har tre fargekanaler. Et svarthvitt bilde har en dybde på 1. Et konvolusjonslag kan bestå av flere ulike filtre og outputet vil bestå av et sett med bilder med ulike uthevede egenskaper. Dette settet med bilder kalles egenskapskart, og dybden til output-egenskapskartet vil ikke lengre være fargekanaler, men antall bilder. Dybden til output-egenskapskartet vil være dybden til inputbildet multiplisert med antall ulike filtre [9].



Figur 3.7: *Illustrasjon av en 2d konvolusjon. Et filter (blå) på 3×3 beveges over inputbildet (grønn) med en steglengde på 1, som gir outputbildet (rosa). Verdien øverst i venstre hjørne i outputbildet (rosa) beregnes på følgende måte:(1 · 1) + (8 · 1) + (4 · 1) + (9 · 0) + (2 · 0) + (0 · 0) + (4 · (-1)) + (5 · (-1)) +(3 · (-1)) = 1.*

3.2.2 Polstring

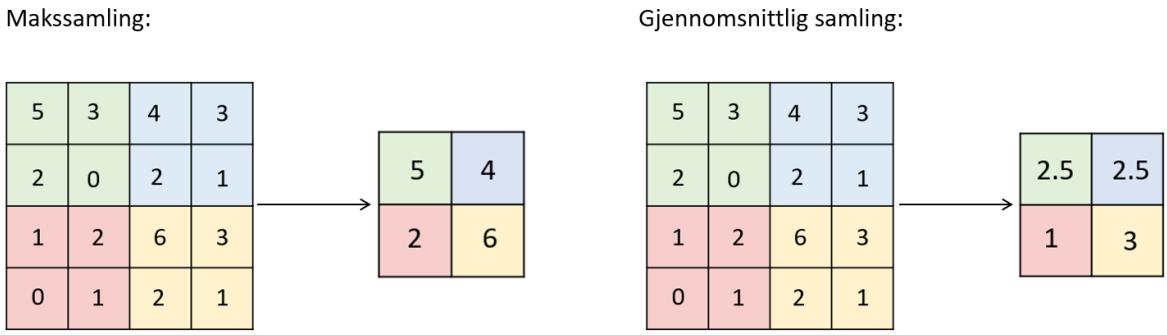
Når en konvolusjon blir påført på et bilde, vil outputstørrelsen bli mindre. Polstring er en teknikk som brukes for å kontrollere størrelsen på outputbildet. Teknikken går ut på å legge på rader og kolonner med piksler med verdien 0 rundt inputbildet [37]. Hvor mange rader og kolonner som legges til inputbildet bestemmer størrelsen på outputbildet. I full polstring øker størrelsen til bildet, samme polstring bevarer størrelsen til inputbildet, og *valid* polstring reduserer størrelsen. En illustrasjon av polstringsteknikken samme polstring er vist i Figur 3.8. Valget av polstring påvirker også viktigheten av kantpikslene i inputbildet [36].



***Figur 3.8**: Illustrasjon som viser effekten av samme polstring i en konvolusjon med et filter med størrelsen 3×3 og steglende 1. En kant med null blir lagt på rundt inputbildet (grønn), som øker størrelsen med to i hver dimensjon. Filteret kan dermed sentreres rundt hver piksel i inputbildet, noe som resulterer i et outputbilde med samme dimensjoner som inputbildet.*

3.2.3 Samlingslag

Samlelag, også kalt nedsamplingslag, brukes i CNNs for å redusere størrelsen på output-egenskapskartet ved å redusere antall piksler, og dermed redusere kapasiteten til nettverket [9, 36]. De to mest vanlige samle-metodene er makssamling og gjennomsnittssamling. I makssamling beveges et område (på f.eks 2×2 piksler) over hele bildet og beholder den høyeste verdien innenfor området i hver ny posisjon. I gjennomsnittssamling beholdes gjennomsnittet av alle pikselverdiene innenfor området i hver ny posisjon [36]. Maks- og gjennomsnittssamling er illustrert i Figur 3.9.



***Figur 3.9:** Illustrasjon av maks- og gjennomsnittssamling med filterstørrelse 2×2 og steglengde 2. Illustrasjon laget etter inspirasjon fra figur i Python Machine Learning av S. Raschka og V. Mirjalili (2019) [36].*

Inkludering av samlelag i nettverk vil føre til en høyere beregningsmessig effektivitet, samt redusere graden av overtilpassing. Makssamlingslag gir egenskaper som er mer robuste mot støy i inputdataen, siden små endringer i de omkringliggende pikslene innenfor et område ikke vil endre resultatet av makssamlingen [36]. Hvilken metode som brukes er avhengig av bildene som er i datasettet. Samlelag gjør også at påfølgende konvolusjonslag vil

undersøke/se med et økende større vindu, som medfører at nettverket vil kunne lære seg mønstre over en større del av bildet [37].

3.3 Semantisk bildesegmentering

Den typiske oppgaven for et konvolusjonsnettverk er klassifiseringsoppgaver der outputet til et bilde er en enkelt klasse. I mange visuelle problemer, slik som medisinsk bildeprosessering, er det ønskelig med informasjon om posisjonen til de ulike klassene i bildet [11]. Semantisk bildesegmentering innebærer å klassifisere alle pikslene i et bilde. Deretter blir ønskede objekter segmentert ut av bakgrunnen [43]. Semantisk segmentering kan brukes for en rekke ulike problemer. I medisinske bilder kan semantisk segmentering brukes til å skille ut svulster, organer eller andre ønskede områder. Algoritmer for selvkjørende kjøretøy bruker semantisk segmentering for å kunne skille mellom veien, en bil og en syklist eller en fotgjenger [36, 37].

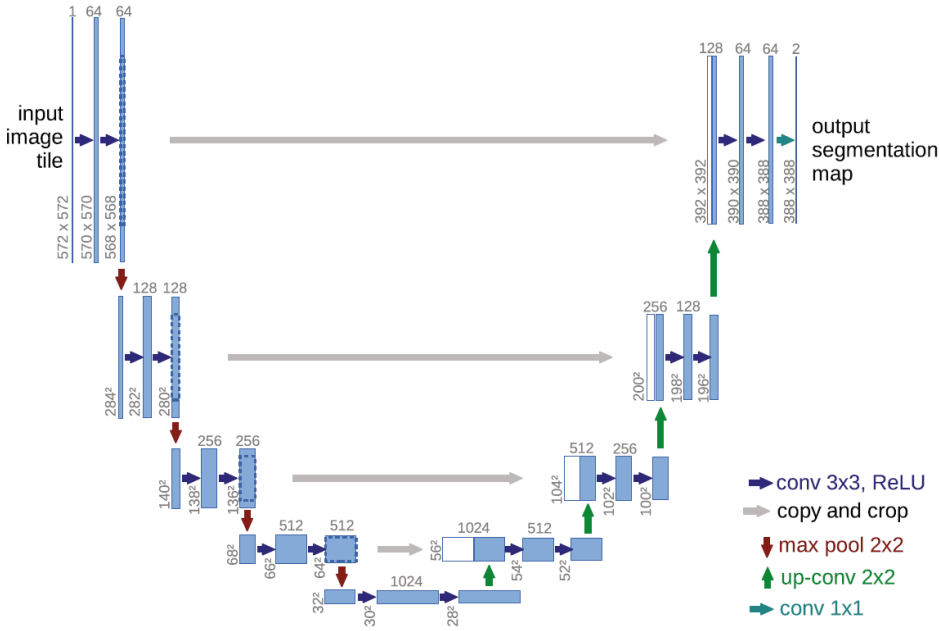
Bruken av konvolusjonsnettverk har i den senere tid vist seg å gi lovende resultater, og da spesielt fullt konvolusjonsnettverk («Fully Convolutional Network», FCN) [43]. Et FCN bytter ut alle fullt koblede lag med konvolusjonslag som har et filter på samme størrelse som inputbildet. Dette gir en output i form av et klassifiserings-*heatmap* og gjør at nettverket kan ta inn inputbilder med vilkårlige størrelser. Klassifiseringsheatmappene som blir produsert er grove, de har dårligere oppløsning enn inputbildet, og dette mister mye informasjon om detaljerte strukturer. Årsaken til disse grove heatmappene er samlelag som gir opphav til nedsamplingseffekter [44, 45]. Nøyaktig segmentering og posisjon er svært viktig i medisinsk bildeprosessering for å kunne stille en diagnose, samt sikre god behandling. For å opprettholde oppløsningen til inputbildet i klassifiseringsheatmappet har det blitt introdusert såkalte koder-dekoder arkitekturer. Disse arkitekturene gjør det mulig å lokalisere objekter samt å bruke konteksten i bildet [11].

Et FCN med samlelag begynner med en kode-del som finner de viktigste egenskapene i bildet, og som en følge av dette reduseres den romlige oppløsningen til bildet [11] [45]. Denne nedsamplingen er kjent som sammentrekningsdelen av nettverket. Så tilføyes dekoderdelen, også kalt utvidelsesdelen, av nettverket i form av oppsamplingslag. Her produseres segmenteringen av objektene og den romlige oppløsningen blir gradvis gjenopprettet. De første lagene i utvidelsesdelen finner formen til objektene, mens de senere lagene finner de små detaljene. Under oppsamplingen blir den romlige oppløsningen gradvis gjenopprettet ved hjelp av hoppforbindelser («skip connections») [11]. Hoppforbindelser er

koblinger mellom egensapskartene i sammentrekningsdelen og egenskapskartene i utvidelsesdelen [46]. Det første egenskapskartet i nedsamplingen blir slått sammen med det siste i oppsamplingen. Det andre egenskapskartet i nedsamplingen blir slått sammen med det nest siste i oppsamplingen, som vist i Figur 3.10. Ronneberger et al. [11] har utviklet en slik modell, kalt U-Net, som i 2015 utkonkurrerte tidlige modeller på segmentering av nevronstrukturer i elektronmikroskopiske bilder [11].

3.3.1 U-Net-arkitektur

Nettverket som er utviklet av Ronneberger et al. [11] består av en nedsamplingsdel og en oppsamplingsdel. Nettverket er illustrert i Figur 3.10. I sammentrekningsdelen blir to 3×3 konvolusjoner påført flere ganger, etterfulgt av aktiveringsfunkjsonen ReLU og makssamling med steglengde på 2 for nedsampling. Ved hver nedsampling blir antall egenskapskanaler doblet. Hvert steg i ekspansjonsdelen består av en oppsampling av egenskapskartet etterfulgt av en 2×2 oppkonvolusjon som halverer antall egenskapskanaler. Videre slås egenskapskartene fra sammentreknings- og ekspasjonsdelen sammen via hoppforbindelser og to 3×3 konvolusjoner blir utført med aktiveringsfunksjonen ReLU etter hver konvolusjon. I det siste laget blir en 1×1 konvolusjon brukt for klassifisering [11].



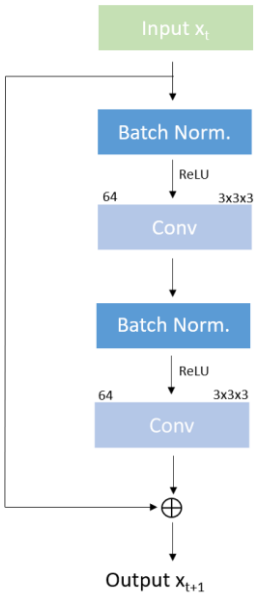
***Figur 3.10**: Illustrasjon av U-Net-arkitekturen. Gjengitt med tillatelse fra Olaf Ronneberger [11]. Hver blå boks representerer et egenskapskart med antall kanaler skrevet ovenfor. Tallet nederst til venstre på hver blå boks er oppløsningen til bildet. De blå pilene er konvolusjonslag med et 3×3-filter. De grå pilene er hoppforbindelser og de hvite boksene representerer de kopierte egenskapskartene fra tidligere lag. Videre representer de røde pilene makssamlingslag, mens de grønne pilene er oppkonvolusjonslag, begge med en størrelse på 2×2. Til slutt representerer den blågrønne pilen en 1×1 konvolusjon, som klassifiserer pikslene i bildet.*

Moe et al. [17] har undersøkt bruken av U-Net til segmentering av kreftsvulster og affiserte lymfeknuter i kontrastforsterket CT, PET og PET/CT-bilder av pasienter med hode- og halskreft. Nettverket som ble brukt er basert på arkitekturen fra Ronneberger et al. [11]. Det ble undersøkt hvilken type modalitet som ga de mest nøyaktige automatiske inntegningene. Modalitetene som ble testet var PET og CT hver for seg samt kombinasjonen PET/CT. Ytelsen under trening på valideringssettet og testing på testsettet gav en noe høyere Dice-score for PET/CT etterfulgt av PET alene og til slutt CT alene. Det ble derfor anbefalt å bruke PET/CT-bilder fremfor kun PET eller CT for automatisk inntegning med CNNs for disse pasienten med hode- og halskreft [17]. Modellen oppnådde på validerings- og testsettet en gjennomsnittlig Dice-score på henholdsvis 0,63 og 0,71.

3.3.2 VoxResNet

Dybden til nettverket har en avgjørende betydning for egenskapsrepresentasjonen. Dypere nettverk er vanligvis vanskeligere å trene sammenlignet med mindre dype nettverk, og selv om de er dypere er de nødvendigvis ikke bedre [47]. Dette er kjent som degraderingsproblemet og det blir et mer betydelig problem når ytelsen synker kraftig med økende dybde på nettverket. Dype residualnettverk med residualblokker har i den senere tid gitt gode resultater på flere store segmenteringsoppgaver i medisinske bilder, slik som i Chen et al. [47] og i Lin et al. [12].

Arkitekturen til residualnettverkene, bestående av blant annet residualblokker, har vist seg å minske degraderingsproblemet, siden residualkoblingene gjør det lettere å optimalisere modellen [47]. Residualblokkene består av to konvolusjonslag med en residualkobling over lagene. Hvert lag i modellen sender input videre til neste lag i modellen, og residualkoblingen sender input over blokken, slik som vist i Figur 3.11. Inputet x_t blir addert med den transformerte dataen med residualkoblingen (som vist i Figur 3.11), som gjør at informasjonen kan bli direkte propagert både framover og bakover i nettverket. Residualenhetene tillater propagering av signaler direkte fra en blokk til de andre ved å bruke identitetskart som hoppforbindelser og aktivering etter addering. Informasjonen som er avdekket i treningsdataen kan dermed utnyttes tilstrekkelig og effektivt for å kunne øke ytelsen. En annen fordel med disse hoppforbindelser er at de ikke tilfører flere parametere og dermed ikke gjør modellen mer beregningsmessig kompleks [47]. I Figur 3.12 er det illustrert et nettverk som tar i bruk residualblokker.

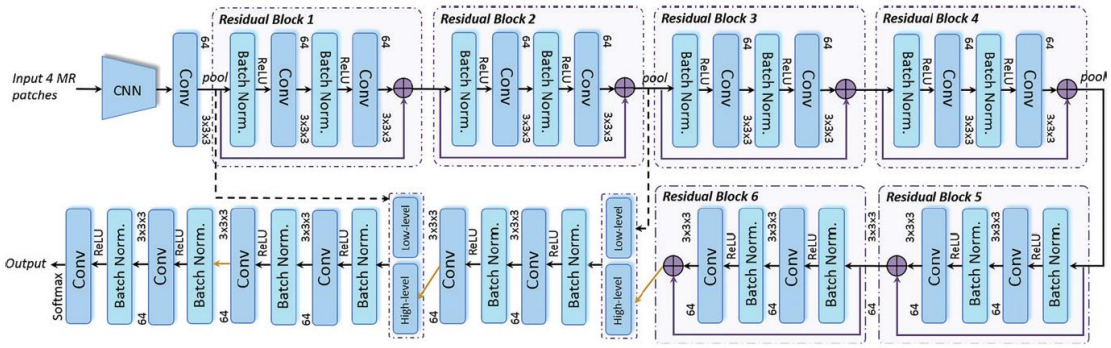


***Figur 3.11**: Residualblokk i et nettverk med batchnorm og 64 filtre. Illustrasjon laget etter inspirasjon fra figur i VoxResNet: Deep voxelwise residual networks for brain segmentation from 3D MR images av Chen et al. [47].*

Chen et al. [47] har undersøkt bruken av VoxResNet til segmentering av ulike substanser i hjernen på 3D MR-bilder. Dette brukes til diagnostisering, oppfølging og behandling av en rekke degenerative nevrologiske sykdommer. Segmentering av hvit substans, grå substans og cerebrospinalvæsken brukes blant annet til å måle og visualisere de anatomiske strukturene i hjernen, kartlegge hvordan hjernen endres over tid og planlegging av operasjoner [47]. Dette er en krevende oppgave siden hjernen har en kompleks anatomi og det er stor variasjon i hjernevev. Bruk av et residualnettverk for denne oppgaven gjør det mulig for modellen å lære flere representative egenskaper for å håndtere de store variasjonene i hjernevevet. Arkitekturen består av stablede residualblokker med totalt 25 volumetriske konvolusjons- og dekonvolusjonslag. I hver residualblokk blir dataen propagert som beskrevet tidligere. Konvolusjonslagene har filtre med størrelse $1 \times 3 \times 3$ eller $3 \times 3 \times 3$, da dette tidlige har vist seg å gi betydelige fordeler når det kommer til beregningshastighet og egenskapskapasiteten [47]. Tre av konvolusjonslagene i modellen har en steglengde lik 2, som reduserer oppløsningen til inputet med en faktor på 8. Dette gjør at nettverket kan ha et større *receptive field size*, og dermed omslutte mer informasjon som øker kapasiteten for å skille mellom ulike objekter/teksturer. Batchnormalisering er inkludert i modellen for å gjøre treningsprosessen raskere. Aktiveringsfunksjonen som brukes i nettverket er ReLU, bortsett fra i siste lag, hvor aktiveringsfunksjonen er Softmax. Denne aktiveringsfunksjonen brukes for klassifiseringsproblemer med flere enn to klasser. Nettverket ble trent ved å bruke mini-batch gradientnedstigning, hvor batchstørrelsen ble satt til 5. Det ble også sett på om bruk av ulike

MR-sekvenser (modaliteter), eller en kombinasjon av flere, vil ha en effekt på ytelsen til nettverket. Nettverket oppnådde gode resultater for segmentering av de ulike hjernesubstansene. Nettverket ble brukt i MICCAI MRBrainS (MR Brain Segmentation) utfordringen som resulterte i førsteplass ut av 37 utfordrere. Rangeringen her ble beregnet ut i fra en rekke ytelsesmål, men Dice-score for beste modell var mellom 0,84-0,89 for de ulike hjernesubstansene [47].

Lin et al. [12] har også undersøkt bruken av VoxResNet for automatisk segmentering, men har sett på 3D MR-bilder med størrelse $96 \times 96 \times 32$ av pasienter med nasofarynkskreft. Et 3D konvolusjonsnettverk basert på Chen et al. [47] sitt VoxResNet ble implementert for å kunne trekke ut representative egenskaper til de kompliserte tumorene basert på fire ulike typer MR-sekvenser. En illustrasjon av nettverket vises i Figur 3.12. Nettverket har totalt 28 lag, bestående av en nedsampling- og en oppsamplingsdel. Nedsamplingsdelen starter med tre konvolusjonslag etterfulgt av seks residualblokker. Det er makssamlingslag etter hver andre residualblokk. Nedsamplingsdelen reduserer den romlige oppløsningen i bildene for å trekke ut relevante egenskaper. Egenskapene vil ha forskjellig grad av kompleksitet. Enkle egenskaper som kanter eller linjer vil trekkes ut i de første lagene, mens mer komplekse egenskaper som struktur vil trekkes ut i de senere lagene. I oppsamplingsdelen brukes dekonvolusjonslag for å øke oppløsningen til bildene. For å kunne utnytte informasjonen fra både de enkle og mer komplekse egenskapene som trekkes ut, blir det brukt hoppforbindelser mellom de første lagene og de senere lagene. Batchnormalisering blir brukt for å gjøre treningsprosessen raskere. Aktiveringsfunksjonen som brukes er ReLU, bortsett fra i siste lag, der brukes Softmax. Nettverket ble trent ved å bruke tilbakepropagering med AdaDelta optimaliseringsalgoritme. Ytelsen på testsettet gitt ved median Dice-score per pasient ga et resultat på 0,79 [12].



***Figur 3.12**: Illustrasjon av arkitekturen av VoxResNet-nettverket som brukes i Lin et al. [12]. Nettverket består av 28 lag med seks residualblokker. Illustrasjon hentet fra Deep Learning for Automated Contouring of Primary Tumor Volumes by MRI for Nasopharyngeal Carcinoma (2019) av Lin et al. [12]. Gjengitt med tillatelse av forfatter.*

3.4 Ytelsesmål

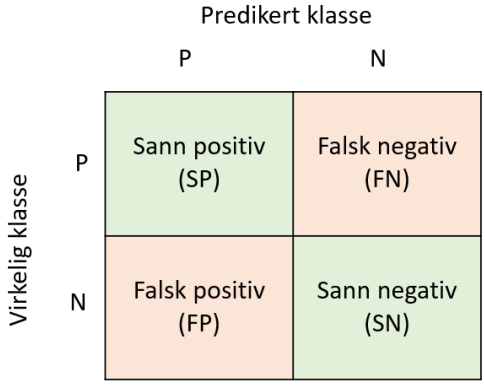
Det finnes flere ulike metoder som kan evaluere prestasjonen til en modell og til å måle modellens relevans [36]. Hvilket mål på ytelse som skal brukes for en modell avhenger av det spesifikke problemet. Et av de mest brukte målene for ytelsen til klassifiseringsmodeller er nøyaktighet:

$$Nøyaktighet = \frac{\text{Korrekte klassifiserte eksemplarer}}{\text{Totalt antall eksemplarer}} \tag{3.11}$$

Dette er en generelt nyttig metode for mål på ytelse, men sier for eksempel ikke noe om graden av feil for hver enkelt klassifisering [36]. Et datasett kan være ubalansert, det vil si at det er mange eksemplarer av en klasse og få av den andre klassen. Modellen vil da kunne få en god nøyaktighet bare ved å klassifisere alle eksemplarene til den klassen som oppstår mest hyppig. Det vil av denne grunn være nødvendig med andre mål på ytelsen som kan ta hensyn til blant annet ubalanse mellom klassene.

3.4.1 Ytelsesmål basert på forvirringsmatrisen

Flere av ytelsesmålene baserer seg på en forvirringsmatrise. En forvirringsmatrise er en matrise som angir andelen av sanne positive (SP), falske positive (FP), sanne negative (SN) og falske negative (FN) klassifiseringer for en binær klassifiseringsmodell [36]. En slik forvirringsmatrise er illustrert i Figur 3.13.



***Figur 3.13**: Oppsettet av en forvirringsmatrise. Kolonnene representerer den predikerte klassen og radene angir den virkelige klassen. . Illustrasjon laget etter inspirasjon fra figur i Python Machine Learning av S. Raschka og V. Mirjalili (2019) [36].*

For et binært klassifiseringsproblem med klassene positiv og negativ, vil sanne positive være andelen eksemplarer som er korrekt klassifisert som positiv. Falske positive vil være eksemplarene som feilaktig har blitt klassifisert som positive og falske negative vil være eksemplarene som feilaktig har blitt klassifisert som negative. Videre er sanne negative andelen av eksemplarene som er korrekt klassifisert som negative [36].

3.4.2 Ytelsesmål basert på grad av overlapp

Overlappbaserte ytelsesmål er mål på prestasjonen til modeller som beskriver forholdet mellom overlapp av den predikerte og sanne klassifiseringen. Sensitivitet og spesifisitet er to eksempler på dette. Sensitivitet er raten av sanne positive (SPR), mens spesifisitet er raten av sanne negative (SNR) [36, 48]:

$$Sensitivitet = SPR = \frac{\mathrm{SP}}{SP\ +\ FN} \tag{3.12}$$

$$Spesifisitet = SNR = \frac{\mathrm{SN}}{SN\ +\ FP} \tag{3.13}$$

Sensitivitet måler hvor mange eksemplarer som er klassifisert som positive sammenlignet med den totale andelen virkelige positive eksemplarer. På den andre siden måler spesifisitet andelen av eksemplarer som er klassifisert som negative sammenlignet med den totale andelen virkelige negative eksemplarer. Disse to målene brukes i seg selv ikke i stor grad som mål på ytelse for problemer innenfor segmentering i medisinske bilder, da de «straffer» feil i små segmenteringer mer enn i større segmenteringer [48]. Presisjon, også kalt positiv prediktiv rate (PPV), brukes derimot i segmenteringsproblemer i medisinske bilder. Dette

målet på prestasjon måler raten av sanne positive basert på den totale andelen som ble klassifisert som positive. Presisjon er definert som [48]:

$$Presisjon = PPV = \frac{SP}{SP\ +\ FP} \tag{3.14}$$

En kombinasjon av sensitivitet (SPR) og presisjon (PPV) gir et mål som kalles F1-score eller Dice-score. Dette er det målet på ytelse som er mest brukt som validering av segmenteringer i medisinske bilder og er matematisk definert som [36, 48]:

$$Dice = F1 = 2\frac{PPV\ \times\ SPR}{PPV\ +\ SPR} = \frac{2SP}{2SP\ +\ FP\ +\ FN} \tag{3.15}$$

Fullstendig overlapp mellom predikert og virkelig segmentering gir en Dice-score på 1, mens ingen overlapp vil gi en Dice-score på 0.

Ved å vekte SPR og PPV forskjellig, vil ytelsesmålet kunne vektlegge den parameteren som er viktigst for det enkelte problemet. En mer generell definisjon av F-score vil da være:

$$F_\beta = \frac{(\beta^2 + 1) \cdot PPV \cdot SPR}{\beta^2 \cdot PPV + SPR} \tag{3.15}$$

hvor β er angir vektingen. Dice-score er et spesialtilfelle av F-score hvor $\beta = 1$. Vektingen er da lik på både PPV og SPR, og de er like viktige [48].

3.4.3 Ytelsesmål basert på distanse

Distansebaserte ytelsesmål er mål på prestasjonen til modellen som tar hensyn til distansen mellom predikert inntegning og sann inntegning [48]. Avstanden kan beregnes på flere ulike måter, for eksempel Hausdorff-distanse («Hausdorff distance», HD) og median overflateavstand («median surface distance», MSD). Hausdorff-distanse er den maksimale avstanden mellom sann inntegning *G* og predikert inntegning *P*, og er uttrykt som:

$$HD(G,P) = max\big(h(G,P), h(P,G)\big) \tag{3.16}$$

hvor *h(G, P)* er definert som:

$$h(G,P) = \max_{g\,\in\,G}\min_{p\,\in\,P} \| g\ - p \| \tag{3.17}$$

$\| g - p \|$ i likning 3.17 er den euklidske avstanden mellom punktene *g* og *p* [48]. HD er sensitiv for utliggere («outliers»), og 95. persentil HD (HD_{95}) bør derfor heller brukes. Denne tar ikke med de mest ekstreme verdiene i beregningen [49]. MSD er medianavstanden mellom sann inntegning og predikert inntegning. HD_{95} viser hvor alvorlig den største

inntegningsfeilen er, mens MSD viser den totale inntegningsfeilen. Det er derfor ønskelig at disse distansebaserte ytelsesmålene er så små som mulig [17].

Kapittel 4: Metode

4.1 Datasett

Datasettet som har blitt brukt til analyse i denne oppgaven består av bilder av pasienter med hode- og halskreft behandlet ved Oslo universitetssykehus (OUS). Dataen ble hentet inn mellom januar 2007 og desember 2013. Pasientene måtte oppfylle visse krav for å bli inkludert i datasettet. Inkluderingskravene var i dette tilfellet plateepitelkreft i munnhule, oropharynx (en del av svelget), hypopharynx (en del av svelget, bak strupehodet) eller strupehodet behandlet med kombinert radio- og kjemoterapi, og tilgjengelige strålebehandlingsplaner basert på FDG-PET/CT [19]. En oversikt over pasientkarakteristikk er gitt i Tabell 4.1. Datasettet som brukes i denne oppgaven har ikke inkludert pasienter som ikke hadde en kontrastforsterket CT sammen med PET-undersøkelsen. Dette resulterte i totalt 197 pasienter i datasettet [18].

Inntegningsmaskene som ble brukt til sammenlikning bestod av *gross tumor volume* (GTV) samt affiserte lymfeknuter hos de pasienter der disse også var påvirket. Inntegningene ble først gjort av en erfaren nukleærmedisinsk lege basert på FDG-PET. Deretter ble resultatene vurdert av en eller to onkologer basert på kontrastforbedret CT samt klinisk informasjon. Som en siste kvalitetskjekk ble inntegningene vurdert av en senioronkolog. En union av disse inntegningene ble brukt som de sanne inntegningene og til å trene modellene [17].

***Tabell 4.1**: Karakteristikk over pasientene i datasettet. Her vises fordelingen av pasientene i alder, kjønn, tumorstadium, tumorplassering o.l. Hentet fra Moe et al. [17].*

Characteristic[a]	All patients ($n = 197$)	Train ($n = 142$)	Validation ($n = 15$)	Test ($n = 40$)
Age [years]				
Mean	60.3	60.7	58.8	59.4
Range	39.9–79.1	39.9–79.1	43.2–73.7	43.0–77.0
Sex				
Female	24.9 %	25.4 %	13.3 %	27.5 %
Male	75.1 %	74.7 %	86.7 %	72.5 %
TNM[b]				
T1	9.1 %	9.2 %	6.7 %	10.0 %
T2	39.6 %	39.4 %	40.0 %	40.0 %
T3	23.4 %	23.9 %	20.0 %	22.5 %
T4	27.9 %	27.5 %	33.3 %	27.5 %
N0	23.9 %	25.4 %	6.7 %	25.0 %
N1	11.7 %	12.0 %	13.3 %	10.0 %
N2	60.9 %	58.5 %	80 %	62.5 %
N3	3.6 %	4.2 %	0 %	2.5 %
AJCC/UICC[b] stage				
I	1.0 %	1.4 %	0 %	0 %
II	8.6 %	9.2 %	0 %	10.0 %
III	19.8 %	19.7 %	20.0 %	20.0 %
IV	70.1 %	69.0 %	80.0 %	70.0 %
Tumour site				
Oral cavity	8.6 %	7.0 %	26.7 %	7.5 %
Oropharynx	72.6 %	73.2 %	60.0 %	75.0 %
Hypopharynx	8.1 %	9.2 %	13.3 %	2.5 %
Larynx	10.7 %	10.1 %	0 %	15.0 %
GTV-T[c] [cm^3]				
Mean	25.0	23.9	37.3	24.3
Range	0.8–285.0	0.8–285.0	2.6–247.2	1.4–157.6
GTV-N[d] [cm^3]				
Mean	19.3	26.6	37.4	19.5
Range	0.5–276.7	0.5–276.7	2.6–247.2	0.5–76.4

[a] Percentages may not sum to exactly 100 due to rounding.
[b] 7th edition
[c] Gross primary tumour volume
[d] Involved nodal volume (for patients with nodal stage $\geq$ N1)

Bildene i datasettet består av 2D tverrsnitt av 3D PET- og kontrastforsterket CT-bilder av pasientene [18]. Det er to kanaler i bildene, en for PET og en for CT. Det ble brukt to klasser for å klassifisere pikslene i bildene: friskt vev eller berørt vev. En piksel som ble tegnet inn som enten tumor eller affisert lymfeknute ble tilegnet klassen berørt vev [40]. Datasettet har i tillegg blitt preprosessert ved å bruke Hounsfield *windowing* preprosessering på CT-bildene. Her ble vindu-senter satt til 70 HU og vindu-bredden satt til 200 HU. Dette resulterte i bilder med CT-verdiene i intervallet [-30, 170] HU. Dette gjøres fordi vi er interessert i CT-bilder av tumorer av mykt vev og involverte lymfeknuter, som kun er representert i en liten del av CT-intervallet. Det vil derfor være unødvendig å analysere hele CT-intervallet, og det kan også gjøre det mer vanskelig å finne relevante egenskaper [17]. Å bruke denne typen preprosessering har også vist å ha en positiv effekt på modellytelsen [40]. Pikselverdiene i PET- og de preprosesserte CT-bildene i datasettet er normalisert mellom 0 og 1. Ulike bilder

kan ha forskjellig pikselintervaller for ulike egenskaper, noe som kan forvirre det nevrale nettverket. Derfor normaliseres pikselverdiene i bildene, slik at de har samme fordeling, noe som også vil gjøre at treningsprosessen kan konvergere raskere [50].

4.1.1 Splitting av dataen

Datasettet er delt i tre deler for å kunne bruke det til dyp læring. Treningssettet brukes for å trene modellen, valideringssettet brukes for å evaluere modellen og optimalisere parameterne, mens testsettet brukes for å teste kvaliteten til den endelige modellen. Det er viktig å ha både et valideringssett og et treningssett. Når modellen optimaliseres basert på ytelsen på valideringssettet, vil informasjon om valideringssettet ha muligheten til å lekke inn i modellen [9]. Modellen kan derfor lett bli overtilpasset på valideringssettet, og den endelige vurderingen må derfor gjøres på et helt usett datasett. Dataen ble stratifisert etter TNM-klassifisering (tumor, lymfeknuter og metastaser) slik som beskrevet i [51]. Det vil si at dataen ble splittet slik at fordelingen av de ulike tumorstadiene er lik over de tre delene av datasettet [17]. Fordelingen av pasienter i de ulike settene kan sees i Tabell 4.2.

***Tabell 4.2**: Fordelingen av antall pasienter i datasettet til de ulike settene.*

Datasett	Antall pasienter
Trening	142
Validering	15
Test	40

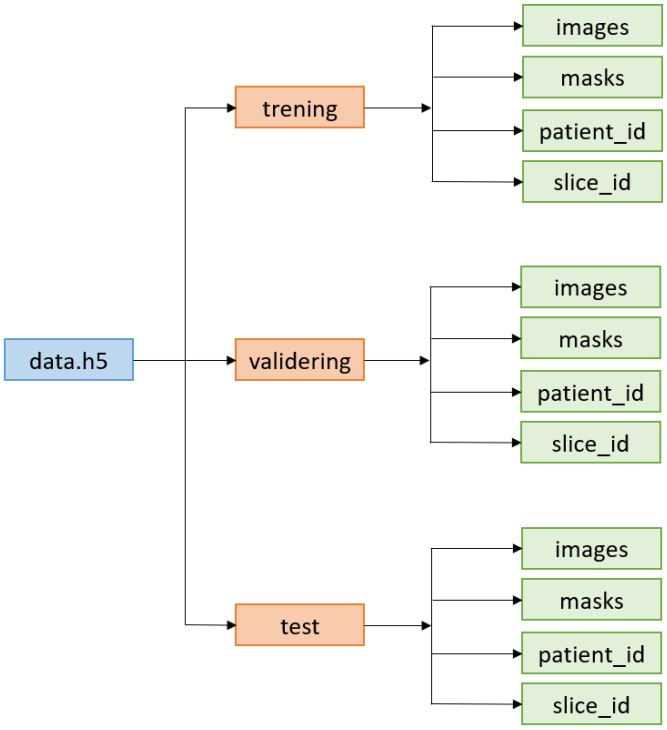
Som en ekstra evaluering skal modellen testes på et eksternt testsett. Det vil si et testsett som inneholder usett data fra en tilsvarende pasientgruppe fra et annet sykehus. Dataen til dette datasettet er hentet fra pasienter fra Maatro Clinic i Nederland. Pasientene har samme kreftdiagnoser som pasientene i datasettet fra OUS og er avbildet med PET/CT. På samme måte som i datasettet fra OUS er pasienter uten en kontrastforsterket CT ekskludert, noe som resulterer i 114 pasienter i det eksterne testsettet. Pasientene har mottatt tilsvarende type behandling som OUS-pasientene. Inntegningene er tilsvarende som for OUS-pasientene, det vil si GTV og affiserte lymfeknuter [52].

4.1.2 Organisering av dataen

Bildedataen ble organisert i en HDF5-fil før den ble brukt som input til modellen. *Hierarchical Data Format version 5* (HDF5) er en nyttig metode for å lagre store

datamengder som gjør det mulig å organisere dataen på en hierarkisk måte [53]. Dataen i en HDF5-fil ligger på disken fram til den skal brukes, noe som gjør det mulig å håndtere datasett som totalt sett overskrider datamaskinens RAM. Oppsettet til en HDF5-fil er delt opp i grupper som kan inneholde datasett eller flere subgrupper. I tillegg kan filen inneholde attributter som er tilleggsinformasjon til enten datasettet eller gruppene [53]. HDF5-filen som ble brukt i denne masteroppgaven har, som nevnt tidligere, tre grupper: trening, validering og test. Hver av disse gruppene inneholder fire datasett: *images*, *masks*, *patient_id* og *slice_id*.

En oversikt over inndelingen til HDF5-filen er gitt i Figur 4.1.



***Figur 4.1**: Illustrasjon av oppsettet til HDF5-filen som ble brukt i denne masteroppgaven. Den blå boksen til venstre er HDF5-filen, de oransje boksene i midten er gruppene og de grønne boksene til høyre er datasettene.*

Størrelsen til de ulike datasettene i HDF5-filen er beskrevet i Tabell 4.3. *n_images* er det totale antall bilder i datasettet, x er antall piksler i x-retning, y er antall piksler i y-retning. For bildene er c antall input kanaler. For dette datasettet med to kanaler, PET og CT, er c lik 2. Dimensjonen til bildene er derfor $x \times y \times c$. For bildene i OUS-datasettet er dimensjonen $191 \times 256 \times 2$. For inntegningsmaskene er m antall segmenteringsmasker. I binære segmenteringsproblemer er m lik 1. Dimensjonen til inntegningsmaskene er derfor $x \times y \times m$ [40].

Tabell 4.3: Beskrivelse av datasettene i HDF5-filen brukt i denne masteroppgaven.

Datasett	Størrelse	Datatype	Innhold
images	[n_images, x, y, c]	float32	Input bildene
masks	[n_images, x, y, m]	float32	Segmenteringsmaskene
patient_id	[n_images]	uint16	Pasientens ID-nummer
slice_id	[n_images]	uint16	Tverrsnitt-nummer i pasientens 3D-bilde

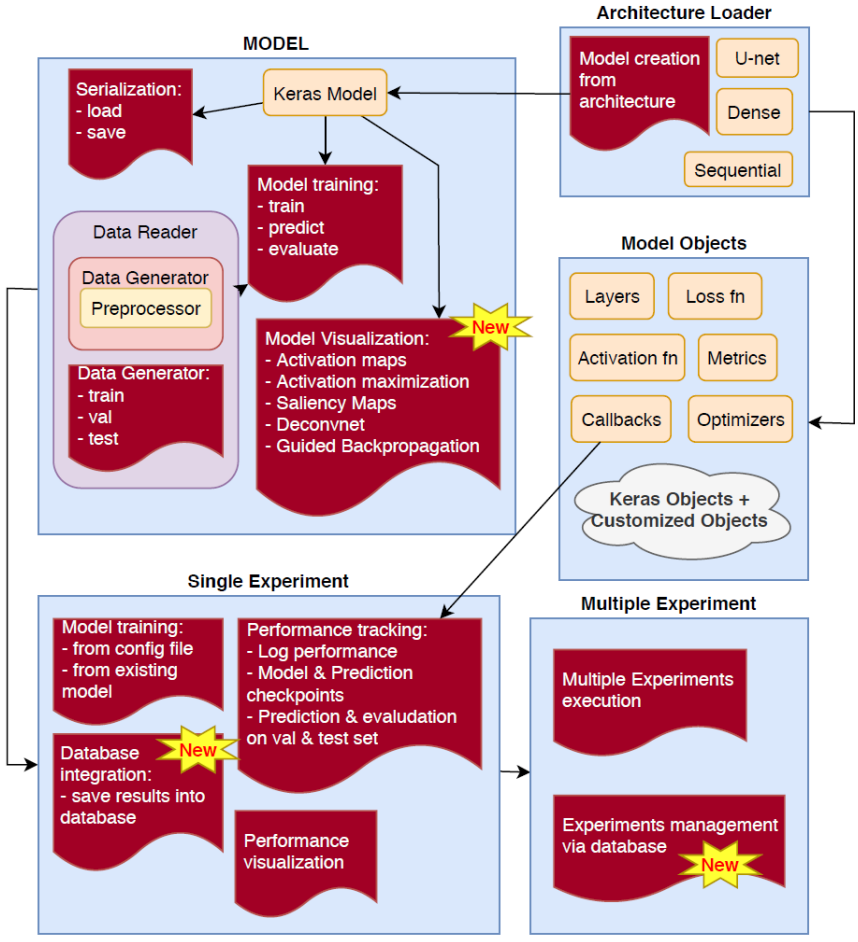
4.2 Rammeverk og programvare

Rammeverket som ble brukt for automatisk inntegning av kreftsvulster i PET/CT-bilder kalles *deoxys* og er utviklet av Bao Ngoc Huynh [54]. Rammeverket gir muligheten til å generere og trene konvolusjonsnettverk på et sett med bilder, og visualisere ytelsen til treningsprosessen samt se på prediksjonene til den trente modellen. For å lage modeller med rammeverket, må det ønskede konvolusjonsnettverket og parametere defineres i en JSON-fil (JavaScript Objection Notation), og dataen må bli lagret i et HDF5 format. I de konfigurerbare JSON-filene kan ulike typer lag, tapsfunksjoner, aktiveringsfunksjoner, ytelsesmål og andre komponenter velges. Rammeverket er basert på Keras[1], en API for dyplæring som bruker maskinlæringsplattformen Tensorflow [55]. Programvarekrav for at *deoxys* skal fungere optimalt er Python 3.7[2] og Keras 2.3.0 eller senere versjoner av disse [54]. En fullstendig forklaring og kode for *deoxys* er tilgjengelig på GitHub siden https://github.com/huynhngoc/deoxys.

Hovedkomponentene i rammeverket er dataleseren, modell, arkitektur innlastningsmodulene og eksperimentmodulen [54]. Oppsettet er vist i Figur 4.2. Dataleseren skal gi inputdata for trening og for evaluering av modellen, og gjør dette med tre hovedfunksjoner. Den første funksjonen består av å lese inn dataen fra disken. Videre kan dataleseren splitte inputdataen inn i trening-, test- og valideringsdatasett. Siden inputdataen til rammeverket er store datasett med medisinske bilder, kan også dataleseren splitte datasettet opp i mindre batcher som får plass i minnet under treningsprosessen [54]. Komponenten modell inneholder en *wrapper* av en Keras modell som inneholder metoder for trening og testing av modellen. Den inneholder også metoder for å lagre og laste opp modellen til og fra disken. Arkitektur innlastningsmodulene, *architecture loader* og *model object* i Figur 4.2, lager komponenter av

[1] https://keras.io/
[2] https://www.python.org/

modellen fra JSON konfigurasjonsfilene. Disse komponentene er enten objekter som er implementert i Keras slik som aktiveringsfunksjoner, tapsfunksjoner eller ytelsesmål, eller egne objekter definert av brukeren. Disse modulene inneholder også noen predefinerte arkitekturer, blant annet U-Net [54]. Eksperimentmodulene brukes for å kjøre ulike eksperimenter med konvolusjonsnettverk med forskjellige komponenter og arkitekturer, hvor ytelsen blir loggført og lagret til disken for hver iterasjon. *Single experiment* brukes for å utføre et enkelt eksperiment. Denne modulen kan visualisere ytelsen på trening og validering samt å visualisere prediksjonene. *Multiple experiment* brukes for å utføre flere eksperimenter, enten etter hverandre eller samtidig. Etter at alle eksperimentene er kjørt, kan modulen finne den beste modellen og bruke denne for å predikere og evaluere modellen med testsettet [54].



***Figur 4.2**: Oppsettet av rammeverket deoxys med de ulike hovedkomponentene. Gjengitt med tillatelse fra Bao Ngoc Huynh [54].*

Under prosessen med å finne den CNN-modellen som gir høyest ytelse, må brukeren kjøre mange eksperimenter for å få nok data som kan analyseres. Å analysere disse ulike dataene krever tid og ferdigheten til å kombinere dataen da de kan ligge på ulike plasser i datamaskinen. Et *Database Management System* (DBMS) gir brukeren mulighet til å sentralisere og organisere dataen som blir produsert under treningsprosessen til en modell [54]. En database gir derfor *deoxys* muligheten til å kommunisere med DBMS som kan administrere resultatene fra flere eksperimenter. DBMS administrer følgende objekter: eksperimentene, eksperiment-kjøringene, de lagrede modellene, de lagrede prediksjonene og ytelsesloggen [54].

4.2.1 Orion

Treningsprosessen av de ulike modellene innebærer mange iterasjoner og beregninger. Datasettet er relativt stort, så trening kjørt på en lokal maskin er derfor tidkrevende og i noen tilfeller ikke mulig på grunn av lagringskapasiteten. Eksperimentene ble derfor gjennomført på regneklyngen Orion[3], som er et tilbud gitt av NMBU (Norges miljø- og biovitenskapelige universitet) som driftes av Centre for Interactive Genetics (CIGENE). Regneklyngen er en ekstern server som hjelper brukeren med å kjøre eksperimenter med et stort CPU minne og GPUer for raskere prosesser. Orion er en Slurm-basert Linux regneklynge med 580 CPUer, 7 TB RAM, 550 TB lagringsplass [56] og 4 GPUer med 256 GB RAM hver [57]. Tilgang til regneklyngen fås av Orion-teamet ved NMBU. For å kunne kobles opp til Orion trengs en SSH-klient (Secure Shell Client) som kan lage en sikker kryptert kobling til regneklyngen. For gjennomføring av eksperimenter i denne masteroppgaven ble MobaXterm[4], brukt som SSH-klient. Man vil da få tilgang til en hjemmekatalog, mulighet til å redigere script, administrere dataene og sende eksperimenter til regneklyngen [57].

4.2.2 Kjøring av eksperimenter

For å kjøre et eksperiment med rammeverket *deoxys* i regneklyngen Orion, må det genereres en JSON-konfigurasjonsfil. Denne filen inneholder informasjon om de ulike parameterne som definerer de forskjellige modellene samt en del som definerer lagene i nettverket. Konfigurasjonsfilen genereres i Spyder[5] og lastes opp til GitHub. Videre lastes konfigurasjonsfilen ned til Orion fra GitHub, hvor eksperimentet kan kjøres. Et eksempel på

[3] https://nmbu-orion-support.readthedocs.io
[4] https://mobaxterm.mobatek.net
[5] https://www.spyder-ide.org/

en slik konfigurasjonsfil ligger i Vedlegg B. Konfigurasjonsfilene til alle de ulike eksperimentene som ble kjørt i Orion kan finnes på GitHub-siden https://github.com/sofiekrogstie/cnn-template.

4.3 Modeller

4.3.1 VoxResNet

Arkitekturen som skal optimaliseres er basert på et residualnettverk og kalles VoxResNet [12, 47]. Oppbygningen er basert på arkitekturen som er brukt i Lin et al. [12]. Dette er et nettverk som består av konvolusjonslag, makssamplingslag og residualblokker, slik som forklart i teorien i delkapittel 3.3.2. Bruken av residualblokker gjør det mulig å trene dypere nettverk, som også er forklart i teorien. En oversikt over de ulike lagene i arkitekturen og antall filtre i de ulike lagene er lagt ved i Tabell A.5 - Tabell A.8 i Vedlegg A.

Konvolusjonslagene har et filter på 3×3 og samme polstring. Antall filtre er det samme antallet i alle konvolusjonslag i modellen. Nettverket har tre nedsamplingslag i form av makssamlingslag og det er to residualblokker etter hvert nedsamplingslag. Makssamlingslagene har en vindusstørrelse på 2×2 og en steglengde på 2. Aktiveringsfunksjonen som brukes i modellen er ReLU, bortsett fra i siste lag hvor aktiveringsfunksjonen som brukes er Sigmoid. Adam er optimaliseringsmetoden som brukes under trening av modellen. Denne metoden er beskrevet i detalj i delkapittel 3.1.2. Hver modell ble trent i maksimalt 200 epoker. Antall epoker er hvor mange ganger modellen trener gjennom hele datasettet. Det ble lagt inn en stoppfunksjon som stoppet treningen dersom tapsverdien ikke endret seg nevneverdig på de siste 30 epokene. Batchstørrelsen angir hvor mange bilder som blir tatt inn i modellen om gangen, og vektene blir oppdatert etter hver batch. Batchstørrelsen ble satt til 16, så vektene blir oppdatert etter hvert 16. bilde. Tapsfunksjonen som brukes er DiceLoss, beskrevet i delkapittel 3.1.2.

VoxResNet-modellen som brukes i denne masteroppgaven skiller seg noe fra VoxResNet-modellen brukt i Lin et al. [12]. Det er brukt ulike optimaliseringsalgoritmer og forskjellige aktiveringsfunksjoner i det siste laget. Her bruker Lin et al. [12] sin modell AdaDelta og Softmax, mens modellen i denne masteroppgaven bruker Adam og Sigmoid. Videre er modellen i denne masteroppgaven en 2D-versjon mens Lin et al. [12] bruker en 3D-versjon. Størrelsen på inputbildene er også ulik.

4.3.2 Optimaliseringsprosess

For å optimalisere modellen ble fire ulike parametere i VoxResNet-modellen variert, og en oversikt over de ulike parameternivåene er gitt i Tabell 4.4. Parameterne som ble variert var batchnormalisering, dropoutrate, læringsraten og antall filtre. De fire parameterne ble satt sammen i alle mulige kombinasjoner, som resulterte i 36 eksperimenter. En oversikt over de ulike eksperimentene som ble gjennomført står i Tabell B.1 i Vedlegg B.

***Tabell 4.4**: Oversikt over parameternivå brukt i VoxResNet.*

Parameter	Verdier
Batchnormalisering	Med, Uten
Dropout	0, 0,5
Læringsrate	10^{-3}, 10^{-4}, 10^{-5}
Antall filtre	48, 64, 80

Batchnormalisering

Normalisering er en rekke ulike metoder som gjør at dataen som sendes inn til en maskinlæringsmodell blir mer lik hverandre, noe som gjør at modellen lærer bedre og kan generalisere bedre til ny data [9]. En mye brukt metode er standardisering, som trekker i fra gjennomsnittsverdien og deler på standardavviket til dataen. Dette gir en fordeling av dataen som er sentrert rundt gjennomsnittet [9]. Når dype nevrale nettverk blir trent vil fordelingen til inputet til hvert lag endre seg, siden parameterne til foregående lag endres. Dette gjør at treningen av modellen tar lengre tid fordi det kreves en lavere læringsrate og en nøye bestemmelse av parameterne [58]. Ioffe og Szegedy [58] introduserte i 2015 en metode for å unngå dette problemet ved å normalisere inputet til lagene for hver batch. Batchnormalisering tillater bruk av høyere læringsrater og gjør at man kan være mindre nøye i bestemmelsen av parameterne. Et batchnormaliserings-lag plasseres ofte etter et konvolusjonslag. Batchnormalisering kan også virke regulariserende på modellen og derfor eliminere behovet for dropout [58]. Under optimaliseringsprosessen til VoxResNet, ble det kjørt modeller med eller uten batchnormalisering. I Figur 3.12 vises en VoxResNet-modell med batchnormaliseringslag. For modeller uten batchnormalisering blir alle disse lagene fjernet.

Dropout

Dropout er, som nevnt i teoridelen i delkapittel 3.1.4, en regulariseringsmetode som brukes for å forhindre overtilpasning. Dropoutraten angir hvor mange tilfeldige noder som blir droppet i hver iterasjon [9]. Under eksperimentene ble dropoutraten satt til enten 0 eller 0,5. Det er altså prøvd ut uten dropout eller med dropout hvor 50 % av nodene blir droppet.

Læringsrate

Læringsraten angir steglengden som brukes i optimaliseringsalgoritmen som brukes for å minimalisere tapsfunksjonen og dermed optimalisere vektene [36]. Læringsratene som ble prøvd ut i de ulike eksperimentene var 10^{-3}, 10^{-4} og 10^{-5}.

Antall filtre

Et filter inngår i et konvolusjonslag og leter etter romlige mønstre. Dette kan for eksempel være horisontale kanter, vertikale kanter eller ulike typer skygger. Antall filtre som blir beregnet av hvert konvolusjonslag bestemmer dybden til output-egenskapskartet. Et nettverk med mange filtre blir mer komplisert [37]. I VoxResNet-arkitekturen er antall filtre likt i alle konvolusjonslagene. De ulike verdiene som ble prøvd ut av antall filtre i eksperimentene var 48, 64 og 80.

4.3.3 Sammenlikningsmodeller

U-Net

Modellen som brukes til sammenlikning er en U-Net-arkitektur fra Moe et al. [17], beskrevet i delkapittel 3.3.1. Arkitekturen består av 4 nedsamplingslag og 4 oppsamplingslag, som begynner med 64 filtre som blir doblet for hvert nedsamplingslag og halvert for hvert oppsamplingslag. Alle konvolusjonslagene har et filter på 3×3, samme polstring og aktiveringsfunksjonen ReLU. Vindustørrelsen til makssamlingslagene er 2×2. Modellen ble trent med 16 i batchstørrelse, med en læringsrate på 10^{-4} og optimaliseringsalgoritmen Adam. Tapsfunksjonen som brukes er DiceLoss.

U-Net-arkitekturen i Moe et al. [17] skiller seg litt fra den originale U-Net-arkitekturen fra Ronneberger et al. [11]. I arkitekturen til Moe et al. [17] blir konvolusjoner med polstring brukt, noe som eliminerer behovet for å beskjære bildene før makssamlingslagene og sammenslåing i hoppforbindelsene. Videre ble det også brukt oppkonvolusjoner med en

størrelse på 3×3 istedenfor 2×2, og i det siste laget ble det brukt en konvolusjon med størrelse 1 [17, 40].

CT-bildene ble preprosessert med *Hounsfield windowing* preprosessering med et vindusenter på 70 HU og en vindubredde på 200 HU. Dette resulterer i bilder med CT-verdier i intervallet [-30, 170] HU. Tabell 4.5 gir en oversikt over de ulike parameterne som ble brukt i U-Net sammenlikningsmodellen. En oversikt over de ulike lagene i arkitekturen og antall filtre i de ulike lagene er lagt ved i Tabell A.1 - Tabell A.4 i Vedlegg A.

Tabell 4.5: *Parametere brukt i U-Net modellen [17].*

Parameter	Verdi
Aktiveringsfunksjon	ReLU
Antall filtre (i første lag)	64
Læringsrate	10^{-4}
Optimaliseringsalgoritme	Adam
Batchstørrelse	16

4.3.4 Bildeaugmentering

Bildeaugmentering vil si å utføre forskjellige transformasjoner på de ulike bildene i datasettet, slik at datasettet blir større. Dette kan gjøre at modellen blir mer generell ved å forhindre at modellen lærer seg irrelevante eller misvisende mønstre i dataen [36]. Maria Ødegaard har i sin masteroppgave [59] undersøkt bildeaugmentering for å finne ut av hvilken kombinasjon av ulike metoder for bildeaugmentering som gir modeller med mest nøyaktig inntegning.

Metodene som brukes for bildeaugmentering kan deles opp i to hovedkategorier: affine transformasjoner og filter- og punkttransformasjoner [60]. I Figur 4.3 vises en oversikt over effekten til de ulike bildeaugmenterings-teknikkene. Affine transformasjoner innebærer geometriske endringer i bildet. Dette kan være rotasjon, forstørring, forminskning, flipping eller forskyvning. For rotasjon angis et tall, i grader, som angir et intervall som går fra den negative verdien av tallet til den positive verdien av tallet. Modellen vil videre velge en tilfeldig grad innenfor dette gitte intervallet som bildet roteres med. For forstørring og forminskning angis et zoom-intervall, hvor tallet 1 er originalbildet, et tall under 1 er en forminskning av bildet og et tall over 1 er en forstørring av bildet. Modellen vil velge et tilfeldig tall innenfor dette intervallet som blir påført bildet i form av en forstørring eller

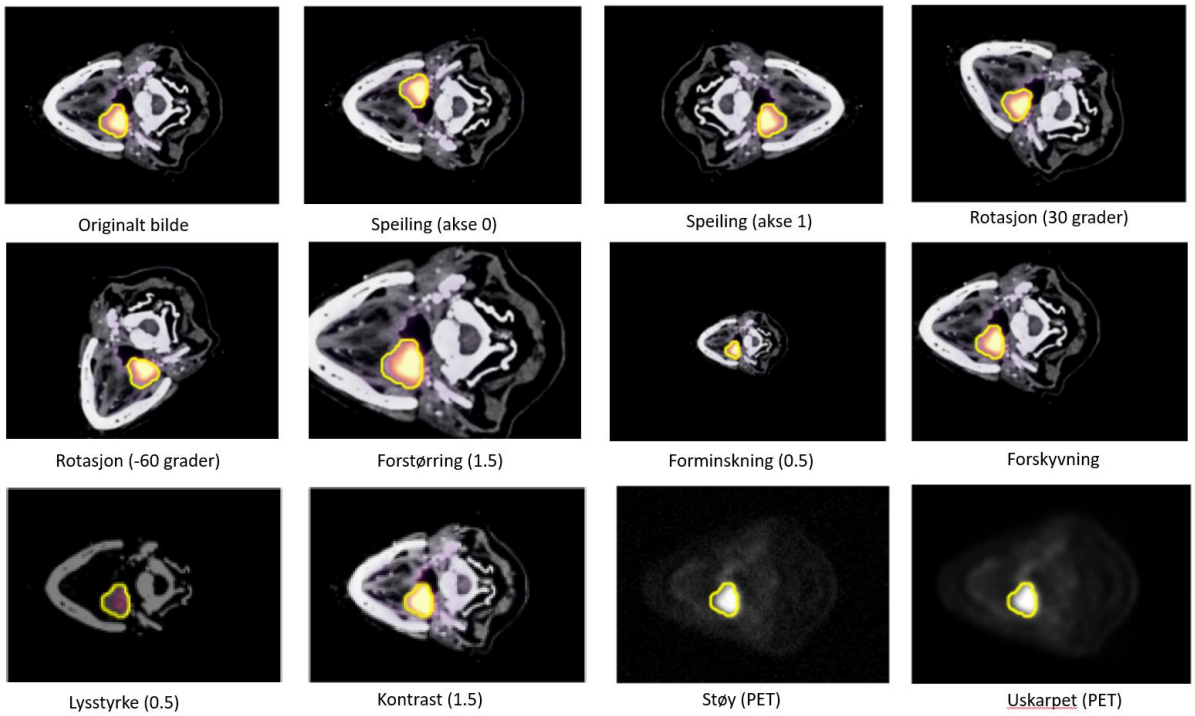
forminskning. Ved et tall på 0,8 vil bildet bli forminsket til en faktor 0,8 av originalbildet. Når et bilde blir forminsket vil det legges på en svart kant rundt bildet. Flipping vil si at bildet flippes horisontalt eller vertikalt. Her angis tallet 0 for flipping om x-aksen (vertikal flipp) og 1 for flipping om y-aksen (horisontal flipp). Forskyvning er en forskyvning av bildet langs aksene. Her angir brukeren et intervall for forskyvningen, som angir hvor mange piksler bildet forskyves med i x- og y-retning. Dette kan føre til at deler av originalbildet ikke lengre er en del av det nye bildet. For å bevare størrelsen til det transformerte bildet legges det til en svart kant der hvor det originale bildet har blitt flyttet fra.

Filter- og punkttransformasjoner innebærer metoder som endrer intensitetsverdiene til pikslene i bildene [60]. Dette kan være endringer i lysstyrke eller kontrast, eller å legge på støy eller en uskarphet. Lysstyrken endres ut ifra et intervall som defineres av brukeren. Hvilket tall som legges på pikselverdiene beregnes ut ifra den maksimale pikselverdien. I datasettet som brukes i denne masteren er pixelverdiene normalisert mellom 0 og 1. Dersom intervallet for lysstyrken defineres som [0,8, 1,2], vil tallet som legges på pixelverdiene ligge i intervallet [-0,2, 0,2]. Dette kommer fra beregningen $1 \cdot (0.8 - 1) = -0,2$ og $1 \cdot (1,2 - 1) = 0.2$. Modellen velger så et tilfeldig tall innenfor dette intervallet som adderes på pikselverdiene i bildet. Kontrast angir intensiteten til pikselverdiene i bildet [60]. Brukeren angir et intervall med verdier som kontrasten kan endres med. Modellen velger et tilfeldig tall innenfor dette intervallet som kontrasten endres med. Støy som legges til bildet vil være Gaussisk fordelt. Brukeren definerer et intervall som angir variansen til støyet. Om støyet skal legges på alle kanalene, ingen eller bare en, kan også bestemmes. For å legge på en uskarphet i bildet brukes et Gaussisk filter. Her definerer brukeren et intervall for standardavviket sigma. På samme måte som for støy kan brukeren bestemme om uskarpheten skal legges på alle kanalene, ingen eller bare en [59].

Verdiene på de ulike bildeaugmenteringsmetodene ble valgt ut ifra verdiene som ga høyest ytelse i testene gjennomført i Marias masteroppgave [59]. Bruken av kun affine transformasjoner ga høyest ytelse. Testene ble gjennomført på det samme datasettet, men med en U-Net modell tilsvarende modellen i Moe et al. [17]. En oversikt over de ulike verdier som ble brukt for de ulike bildeaugmenteringsmetodene i denne oppgaven er gitt i Tabell 4.6.

Tabell 4.6: *Oversikt over parametere brukt for bildeaugmentering i denne masteroppgaven.*

Metode	Verdi
Rotasjonsintervall	90
Zoom-intervall	[0,5, 1,5]
Flipp	0 (vertikal flipp, om x-aksen)
Forskyvingsintervall	10, 10 ([-10,10] i y-retning og [-10,10] i x-retning)



Figur 4.3*: Effekten av ulike teknikker for bildeaugmentering. Støy og uskarphet legges kun til PET-bildene, mens de resterende teknikkene gjøres på PET/CT-bildene. Det originale bildet er gitt øverst til venstre.*

4.4 Evaluering av parameternivå

For å evaluere hvilken modell som ga den høyeste ytelsen, ble gjennomsnittlig Dice-score for hvert eksperiment plottet for hvert nivå av modellparameterne gitt i Tabell 4.4. Videre ble statistiske tester brukt for å finne eventuelle signifikante forskjeller mellom de ulike nivåene for parameterne. Den statistiske analysen ble gjennomført i R[6] med et signifikansnivå på 0,05.

[6] https://www.r-project.org/

Koden brukt til den statistiske analysen ligger i Vedlegg C. Hypotesene for de statistiske testene ble definert som følgende:

H_0: Valg av parameternivå har ingen effekt på ytelsen

H_A: Valg av parameternivå har en effekt på ytelsen

For å finne de mest optimale nivåene av hver parameter samt se på interaksjonene mellom de ulike parameterne var det ønskelig å utføre en N-veis ANOVA (Analysis of Variance) [61]. For å kunne bruke en ANOVA må visse antagelser være oppfylt. Dette førte til tre ulike senarioer, basert på om antagelsene var oppfylt.

Dataen som ble brukt til den statistiske analysen er basert på Dice-score per tverrsnitt. Dice-score per tverrsnitt generes for hvert enkelt eksperiment gjennom koden *post processing* i *deoxys*. Ved å kjøre denne koden vil man få et datasett som inneholder pasientnummer, tverrsnittnummer og Dice-score per tverrsnitt, totalt 1033 rader for hvert eksperiment, siden det er totalt 1033 tverrsnitt i valideringssettet. Denne dataen for alle 36 eksperimentene ble samlet i et datasett sammen med de ulike nivåene for hver parameter. Dette resulterer i et datasett med 36 ganger 1033 rader og 8 rader (pasientnummer, tverrsnittnummer, Dice-score per tverrsnitt, parameternivå til de fire parameterne, eksperimentnummer).

Antagelsen for ANOVA som må være oppfylt, er at residualene er normalfordelt. Dette undersøkes i diagnoseplott, som er QQ-plott og residualer mot tilpasset modell-plott. Her kan det også gjennomføres en Anderson-Darling normalitetstest for å underbygge resultatene fra diagnoseplottene [62]. Hypotesene for normalitetstesten vil være:

H_0: Residualene er normalfordelt

H_A: Residualene er ikke normalfordelt

Det første senarioet vil være at antagelsen om normalfordeling av residualene er oppfylt, og ANOVA kan brukes for å evaluere modellen. ANOVA brukes da for å finne ut om det er en signifikant forskjell mellom ulike nivåer i modellen samt undersøke om det er en interaksjon mellom parameterne. I dette tilfellet vil post-hoc-analyser brukes for å finne ut hvor forskjellen i nivåene til parameterne ligger.

Det andre senarioet vil være at residualene ikke er normalfordelte. Det kan da være nyttig å forsøke å transformere dataen, for så å undersøke om residualene til den transformerte dataen er normalfordelt. Å transformere dataen vil endre referansepunktene, men ikke avstanden

mellom datapunktene. Ulike transformasjoner som kan brukes er log-transformasjon, kvadratrot-transformasjon, invers-transformasjon eller Box-Cox-transformasjon [61, 63]. Normalfordelingen til residualene blir undersøkt på samme måte som tidligere, ved å se på plottene. Dersom antagelsen nå er oppfylt, kan ANOVA og post-hoc-analysen utføres på den transformerte dataen.

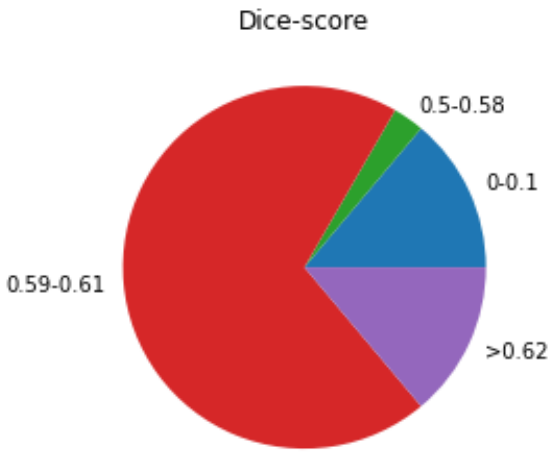
Dersom transformeringen av dataen ikke vil føre til normalitet av residualene, bør ikke-parametriske tester heller brukes [64]. I dette tilfellet kan en ikke-parametrisk Friedmantest eller en tosidig Wilcoxon signed rank-test brukes, avhengig av antallet av parameternivå [64, 65]. Dette er tester som ser på parameterne hver for seg. Friedmantesten krever at dataen er organisert i et *unreplicated complete block design* [64]. Dette kan gjøres ved å for eksempel se på gjennomsnittlig Dice-score per tverrsnitt for samme tverrsnitt med samme nivå av en av parameterne. Ved å gjøre dette får man et datasett for hver parameter som inneholder gjennomsnittlig Dice-score per tverrsnitt for samme tverrsnitt med samme nivå av parameteren for alle nivåene til parameteren. Friedmantesten brukes for parameterne med tre eller flere nivåer, mens Wilxocon signed rank-testen brukes for parameterne med to nivåer. Disse testene vil kunne si om det er en signifikant forskjell mellom nivåene av parameterne. For Friedmantesten som brukes på parameterne med tre nivåer, må det også brukes en tosidig Nemenyi post-hoc-test [65] for å avgjøre hvilket nivå av parameteren som gir mest nøyaktig modell. Effektstørrelsen kan også beregnes, for å si hvor stor innvirkning valget av nivå har. Dersom Wilcoxon signed rank-testen viser at det er en signifikant forskjell mellom nivåene for parameterne med to nivåer, kan det parameternivået som gir signifikant forskjell bestemmes ut ifra et boksplott. Effektstørrelsen kan også beregnes for Wilcoxon signed rank-testen.

Kapittel 5: Resultater

Effekten av nivåvalg for de ulike parameterne på modellytelsen blir beskrevet i dette kapittelet. Det ble trent 36 modeller med ulike nivåer for de fire parameterne, slik som beskrevet i Tabell B.1 i Vedlegg B. Det nivået av hver parameter som gir høyest gjennomsnittlige Dice-score vil til sammen gi den beste modellen. Valget av parameternivå er basert resultatene fra valideringssettet. Den beste modellen ble også trent med bildeaugmentering. I tillegg ble den beste modellen, både med og uten augmentering, vurdert på testsettet og det eksterne Maastro-testsettet.

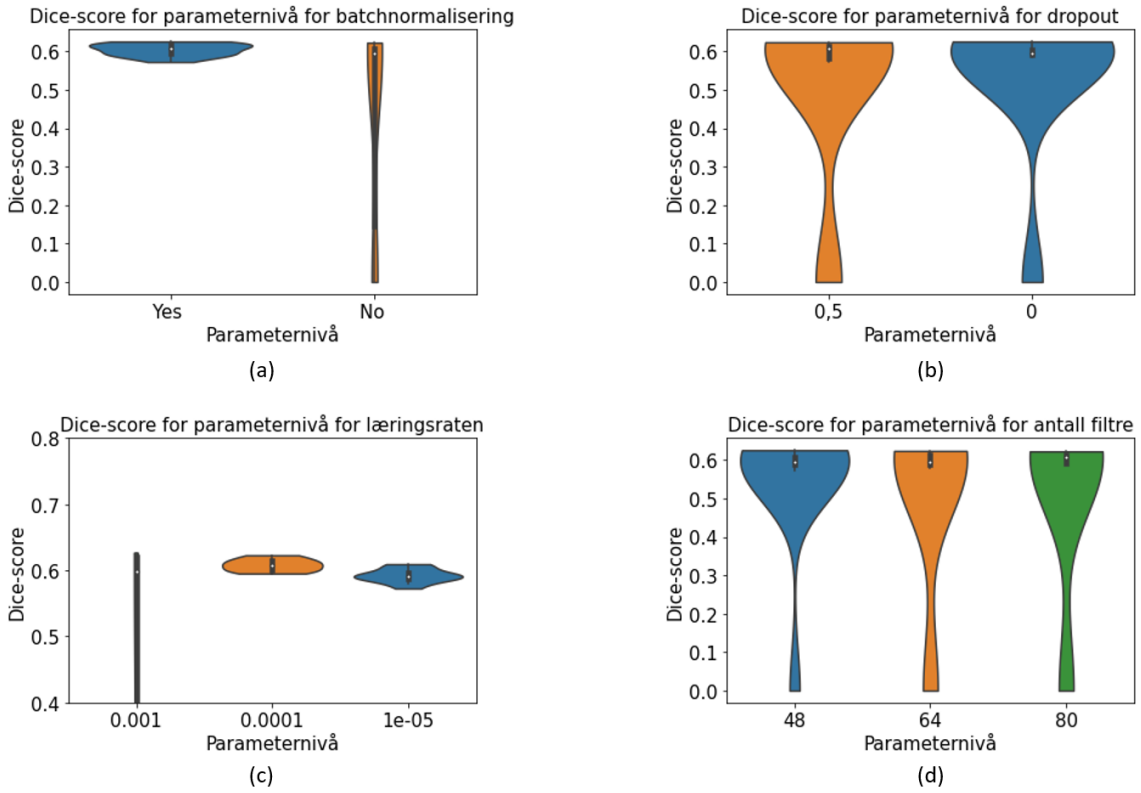
5.1 Resultater fra eksperimentene

For hver modell som ble trent, ble ytelsen vurdert for hver epoke. Ytelsen er målt i gjennomsnittlig Dice-score per tverrsnitt, slik som forklart i teorien i delkapittel 3.4.2. For å finne den epoken med de beste mulige vektene, for senere bruk på testsettet, ble den epoken med høyest ytelse valgt for hvert eksperiment. En oversikt over den høyeste gjennomsnittlig Dice-score per tverrsnitt for de ulike eksperimentene er gitt i Tabell B.1 i Vedlegg B. Fordelingen av disse Dice-scorene vises i sektordiagrammet i Figur 5.1. Det er fem eksperimenter som får en gjennomsnittlig Dice-score på over 0,62. Videre er det 25 eksperimenter som får en ytelse mellom 0,59 og 0,61 og fem eksperimenter som får en ytelse under 0,1. Et eksperiment får en ytelse på mellom 0,5 og 0,58.



Figur 5.1: *Sektordiagram som viser fordelingen av den høyeste gjennomsnittlige Dice-score for alle eksperimentene. 25 eksperimenter får en ytelse på mellom 0,59 og 0,61. 5 eksperimenter får en ytelse > 0,62. Videre får 5 eksperimenter en ytelse på under 0,1 og et eksperiment får en ytelse på mellom 0,5 og 0,58.*

For å undersøke videre hvordan valg av parameterens nivå påvirket modellens ytelse, ble gjennomsnittlig Dice-score for hver parameter plottet i fiolinplot. Disse vises i Figur 5.2. Noen av plottene har tydelige trender for hvilket nivå av parameterne som gir høyeste gjennomsnittlige Dice-score. For batchnormalisering (Figur 5.2 a) ligger alle eksperimentene som inkluderte batchnormalisering på en Dice-score på rundt 0,6, mens eksperimentene som ikke inkluderte dette ligger spredt mellom 0 til rundt 0,6, med noen flere eksperimenter liggende rundt de høyere Dice-scorene. For dropout (Figur 5.2 b) er fordelingen mellom de to parameternivåene mer jevnt fordelt. Begge fiolinene er bredest på en Dice-score på rundt 0,6. Det ser likevel ut som at en dropoutrate på 0 har en litt bredere fordeling rundt Dice-score på ca. 0,6 og har en litt smalere hale, sammenliknet med dropoutrate 0,5. For læringsraten (Figur 5.2 c) oppnår eksperimentene med en læringsrate på 10^{-4} høyest gjennomsnittlig Dice-score på rundt 0,6. Like under ligger ansamlingen av eksperimenter med læringsraten 10^{-5} med en Dice-score på rett under 0,6. Eksperimentene med læringsrate 10^{-3} har oppnådd Dice-scorer som ligger spredt mellom 0 og rett over 0,6. For eksperimentene med ulike antall filtre (Figur 5.2 d) har generelt alle parameternivåene eksperimenter som har oppnådd høye Dice-scorer. Det er likevel 48 i antall filtre som har den bredeste fiolinen for høyest Dice-score og samtidig smalest hale. Det vil derimot totalt sett være vanskelig å visuelt bestemme hvilket nivå for de ulike parameterne som er den beste. Det ble derfor utført statistiske tester for å underbygge observasjonene fra fiolinplottene i Figur 5.2 og for å se om det var en signifikant forskjell mellom de ulike nivåene.



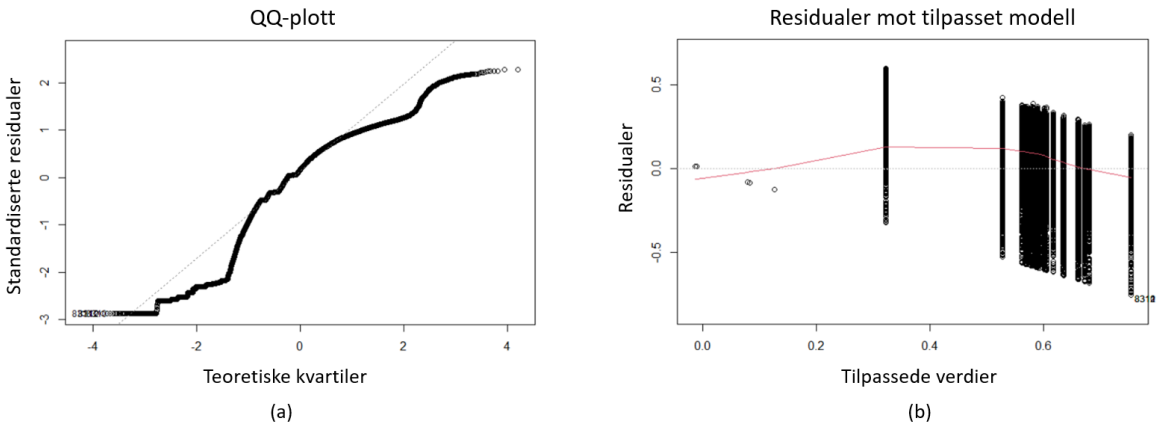
Figur 5.2: *Fiolinplot som viser gjennomsnittlig Dice-score for (a) Batchnormalisering, (b) Dropout, (c) Læringsrate, (d) Antall filtre. Boksen i hver fiolin representerer observasjonene mellom første og tredje kvartil, og prikken er medianen. Lengden på fiolinene representerer variasjonen i gjennomsnittlig Dice-score for de ulike parameternivåene, mens tykkelsen representerer hvor mange eksperiperimenter som får denne Dice-scoren.*

5.2 Statistiske tester

5.2.1 Normalfordeling

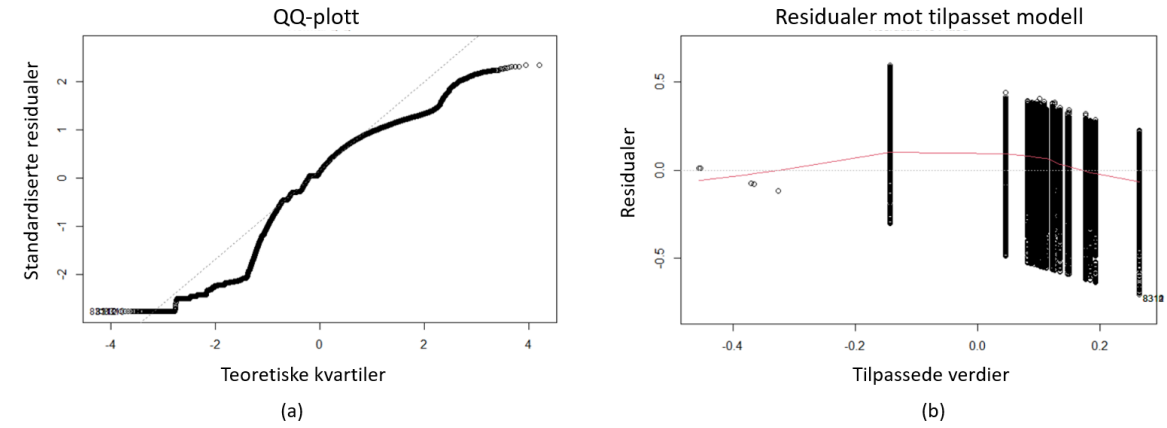
Som forklart i delkapittel 4.4, må visse betingelser være oppfylt for å kunne gjennomføre en ANOVA-analyse. I Figur 5.3 vises diagnoseplottene (QQ-plot og residualer mot tilpasset modell) til residualene til dataen. Det ble også gjennomført en Anderson-Darling normalitetstest for å underbygge resultatene fra diagnoseplottene. Hypotesene for denne testen er definert i delkapittel 4.4.

Residualene er normalfordelt når dataen i QQ-plottet følger den diagonale stiplede linjen i plottet (Figur 5.3 a), og i plottet for residualene mot tilpasset modell (Figur 5.3 b) vil den røde linjen legge seg på null samt at datapunktene vil legge seg jevnt fordelt over og under denne linjen. Ut ifra Figur 5.3 vises det at residualene ikke er normalfordelte, og med en p-verdi < 0,001 fra Anderson-Darling-testen, vil nullhypotesen forkastes og kravet om normalfordeling av residualene er altså ikke oppfylt.



Figur 5.3: *Diagnoseplott for den originale dataen. Figur (a) viser QQ-plottet og figur (b) viser plottet for residualer mot tilpasset modell.*

Det ble videre undersøkt om transformeringer av datasettet kunne gjøre residualene normalfordelt. De ulike transformasjonene som ble testet var logaritmisk-transformasjon, invers-transformasjon, kvadratrot-transformasjon og Box-Cox-transformasjon. I Figur 5.4 vises QQ-plottet og residualer mot tilpasset modell-plottet for Box-Cox-transformasjonen. Tilsvarende plott for de øvrige transformasjonene er vist i Vedlegg C. Det ble også gjennomført Anderson-Darling normalitetstester for den transformerte dataen, resultatene vises i Tabell 5.1.



Figur 5.4: *Diagnoseplott for Box-Cox-transformert data. Figur (a) viser QQ-plottet og figur (b) viser plottet for residualer mot tilpasset modell.*

Tabell 5.1: Oversikt over p-verdier fra Anderson-Darling normalitetstest for transformert data

Transformert data	P-verdi
Logaritmisk	< 0,001
Invers	< 0,001
Kvadratrot	< 0,001
Box-Cox	< 0,001

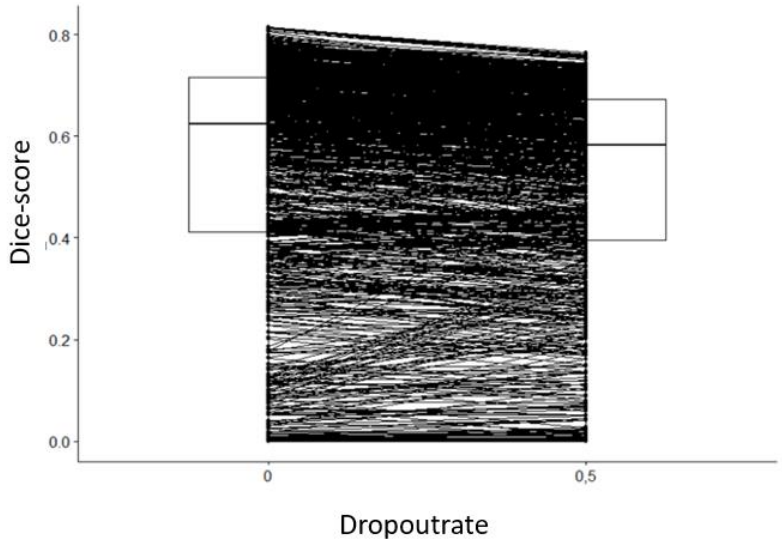
Med den samme nullhypotesen som tidligere, vil nullhypotesen forkastes for alle de transformerte datasettene også. Residualene er altså fortsatt ikke normalfordelt, og denne ANOVA-betingelsen kan derfor ikke oppfylles. Andre statistiske tester ble derfor tatt i bruk.

5.2.2 Effekt av parametere

Effekten av hver parameter ble videre undersøkt hver for seg, slik som forklart i delkapittel 4.4. Ulike tester ble brukt avhengig av antallet nivåer i parameteren. Wilcoxon signed rank-test ble brukt for parameterne med to nivåer. Friedmantesten og Nemenyi post-hoc-test ble brukt for parameterne med tre nivåer. Hypotesene som ble brukt i testene var de samme for alle parameterne og er definert i delkapittel 4.4.

Dropout

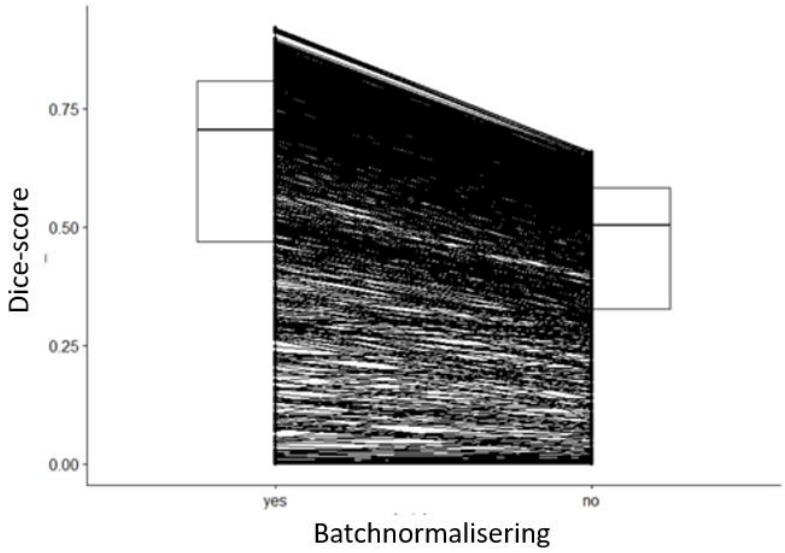
P-verdien for Wilcoxon signed rank-testen var mindre enn 0,001. Dette vil si at nullhypotesen kan forkastes med et signifikansnivå på 0,05. Det vil altså si at det er en forskjell mellom valg av nivå av denne parameteren. De to nivåene som ble testet ut, var en dropoutrate på 0 eller 0,5. For å finne ut hvilken av disse som ga den høyeste ytelsen kan man se på boksplottet i Figur 5.5. Boksplottet viser at ytelsen ligger høyere for modellene med 0 i dropoutrate. Effektstørrelsen for dette parameternivået var stor.



Figur 5.5: *Boksplott for de to parameternivåene til parameteren dropout. Boksene representerer resultatene som ligger mellom første og tredje kvartil, mens streken i hver boks viser medianen. Streken mellom de to boksene viser sammenhengen mellom samme tverrsnitt til samme pasient med ulikt parameternivå.*

Batchnormalisering

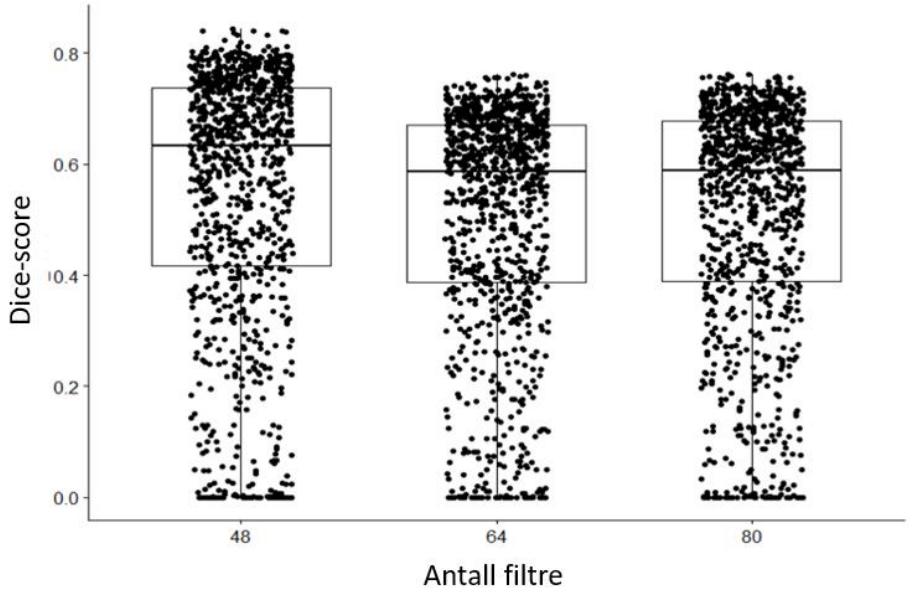
Wilcoxon signed rank-testen ga en p-verdi < 0,001. Nullhypotesen om ingen effekt i valg av parameternivå vil derfor kunne forkastes med et signifikansnivå på 0,05. Også her kan man ut ifra boksplottet avgjøre hvilket nivå av parameteren som gir høyest ytelse. Ut ifra Figur 5.6 vil nivået som gir den høyeste ytelsen være *yes*, altså med batchnormalisering. Effektstørrelsen ble også undersøkt, og effekten var stor.



Figur 5.6: *Boksplott for de to parameternivåene til parameteren batchnormaliseing. Boksene representerer resultatene som ligger mellom første og tredje kvartil, mens streken i hver boks viser medianen. Streken mellom de to boksene viser sammenhengen mellom samme tverrsnitt til samme pasient med ulikt parameternivå.*

Antall filtre

Friedmantesten ga en p-verdi < 0,001. Med denne lave p-verdien kan nullhypotesen forkastes med et signifikansnivå på 0,05, og det vil derfor være en signifikant forskjell mellom hvilket parameternivå som velges. For å velge det nivået som gir høyest ytelse, må det gjøres en post-hoc-test. Nemenyi post-hoc-analyse ga p-verdier < 0,001 for kombinasjonene mellom alle de ulike parameternivåene. Nullhypotesen som sier at det ikke er en forskjell mellom nivåene, kan dermed forkastes. For å avgjøre hvilket nivå som ga den høyeste gjennomsnittlige ytelsen kan man se på boksplottet i Figur 5.7. Det er 48 filtre som totalt gir modeller med høyest ytelse. Effektstørrelsen for valg av parameternivå var moderat effekt.

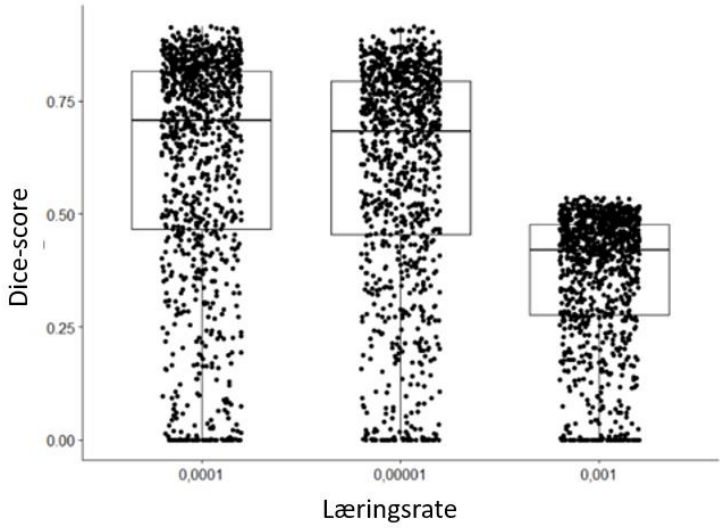


Figur 5.7: *Boksplott for de tre parameternivåene til parameteren antall filtre. Boksene representerer resultatene som ligger mellom første og tredje kvartil, mens streken i hver boks viser medianen. Punktene i plottene representerer hver Dice-scoren til hver modell for hvert enkelt parameternivå.*

Læringsrate

For læringsraten resulterte Friedmantesten i en p-verdi < 0,001. Nullhypotesen kan dermed forkastes med et signifikansnivå på 0,05, og det vil dermed være en signifikant forskjell mellom valget av de ulike læringsratene. Nemenyi-testen ble brukt som post-hoc-test for å avgjøre hvilken læringsrate som ga den høyeste ytelsen. Post-hoc-testen ga p-verdier < 0,001 for alle kombinasjonene mellom de ulike parameternivåene. Post-hoc-testen kunne dermed ikke avgjøre mellom hvilke nivåer av læringsraten den signifikante forskjellen ligger. Det vil derimot være mulig å se på boksplottet hvilket nivå for læringsraten som gir den høyeste ytelsen. I Figur 5.8 vises boksplottet for de forskjellige nivåene til læringsraten. For 10^{-4}

ligger boksen og medianen høyere enn for de andre parameternivåene, og det kan ut ifra dette se ut til at det dette parameternivået som gir modellene med høyest Dice-score. Effektstørrelsen ble også her vurdert, og resulterte i stor effekt.



Figur 5.8: *Boksplott for de tre parameternivåene til parameteren læringsrate. Boksen i representerer resultatene som ligger mellom første og tredje kvartil, mens streken i hver boks viser medianen. Punktene i plottene representerer hver Dice-scoren til hver modell for hvert enkelt parameternivå.*

5.3 Beste modell

Ut ifra de statistiske testene ble nivået av hver parameter som ga den høyeste gjennomsnittlige ytelsen bestemt. Modellen som har disse parameternivåene, er definert som den beste modellen. I Tabell 5.2 er en oversikt over parameternivåene til den beste modellen gitt. Videre ble det også undersøkt hvordan den beste modellen presterte ved å legge til bildeaugmentering.

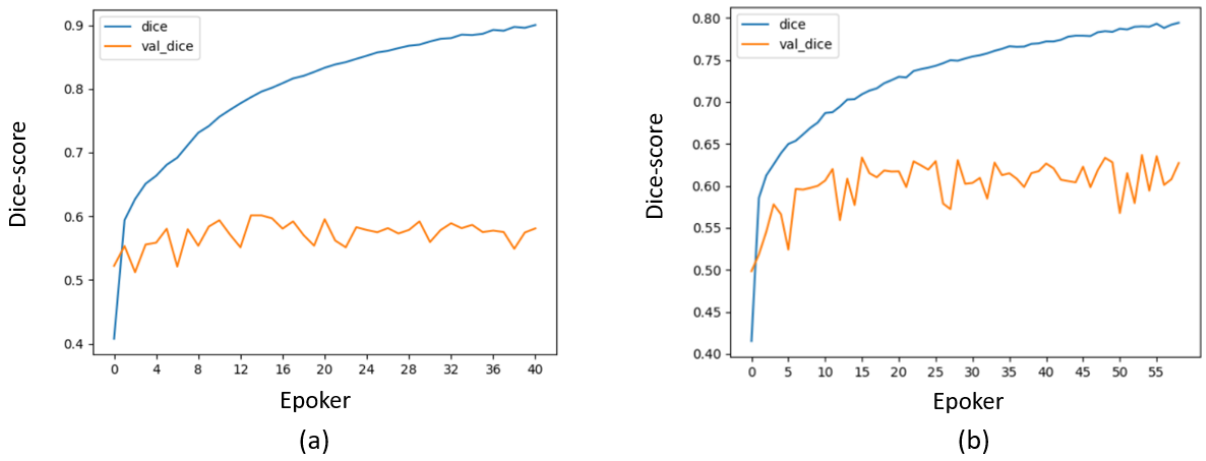
Tabell 5.2*: Oversikt over det nivået av hver parameter som ga høyest modellytelse. Disse parameterne definerer den beste modellen.*

Parameter	Verdi
Dropout	0 (uten)
Batchnormalisering	Ja
Læringsrate	10^{-4}
Antall filtre	48

5.3.1 Valideringssettet

I Figur 5.9 vises grafene som viser utviklingen av ytelsen over epokene på treningssettet sammenliknet med valideringssettet for den beste modellen uten og med augmentering. Modellene trente i henholdsvis 40 og 55 epoker, utstrekningen på x-aksen er derfor ulik. Her vises det at Dice-scoren for treningssettet fortsetter å øke, mens valideringskurven er relativ flat. Ved å fortsette treningen vil trolig treningskurven også flate ut. For modellen uten bildeaugmentering (Figur 5.9 a) stabiliserer valideringskurven seg på mellom 0,5 og 0,6, og for modellen med bildeaugmentering (Figur 5.9 b) stabiliserer valideringskuven seg på rundt 0,6. I valideringskurvene kan det observeres noen tagger, noe som kan tyde på at modellene er sensitive for noen spesifikke oppdateringer av vektene.

Det er verdt å legge merke mellomrommet mellom validering- og treningskurvene. Ideelt sett bør valideringskurven følge treningskurven i en større grad. Mellomrommene tyder på at informasjon fra valideringssettet kan ha lekket inn i modellen. Størrelsen på mellomrommet tyder også på at modellen er godt tilpasset treningssettet, men sliter imidlertid litt med å generalisere til dataen i valideringssettet. Det at modellen er godt tilpasset treningssettet og ikke valideringssettet kan tyde på at modellen er overtilpasset.



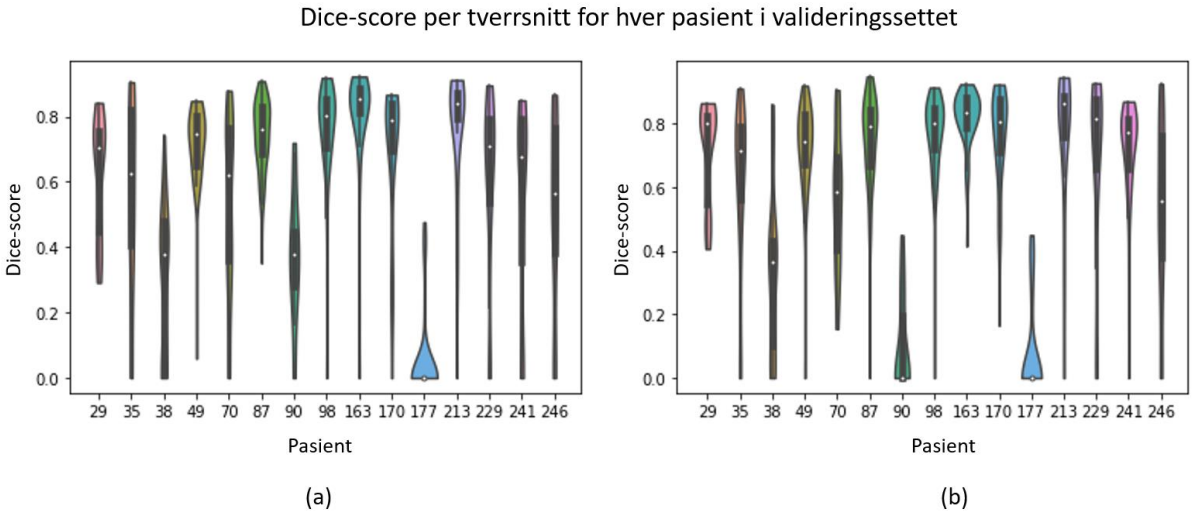
Figur 5.9: *Trening- og valideringskurve for (a) den beste modellen uten bildeaugmentering og (b) den beste modellen med bildeaugmentering. De blå grafene representerer treningskurvene, og de oransje grafene representer valideringskurvene.*

Den beste modellen uten og med bildeaugmentering fikk en gjennomsnittlig Dice-score per tverrsnitt på henholdsvis 0,601 ± 0,28 og 0,637 ± 0,27 på valideringssettet. For å vurdere hvordan modellen presterte på en pasient som en helhet og til bruk for sammenlikning senere, ble det beregnet en Dice-score per pasient. I Tabell 5.3 vises en oversikt over Dice-score per pasient for den beste modellen uten og med bildeaugmentering for valideringssettet. For

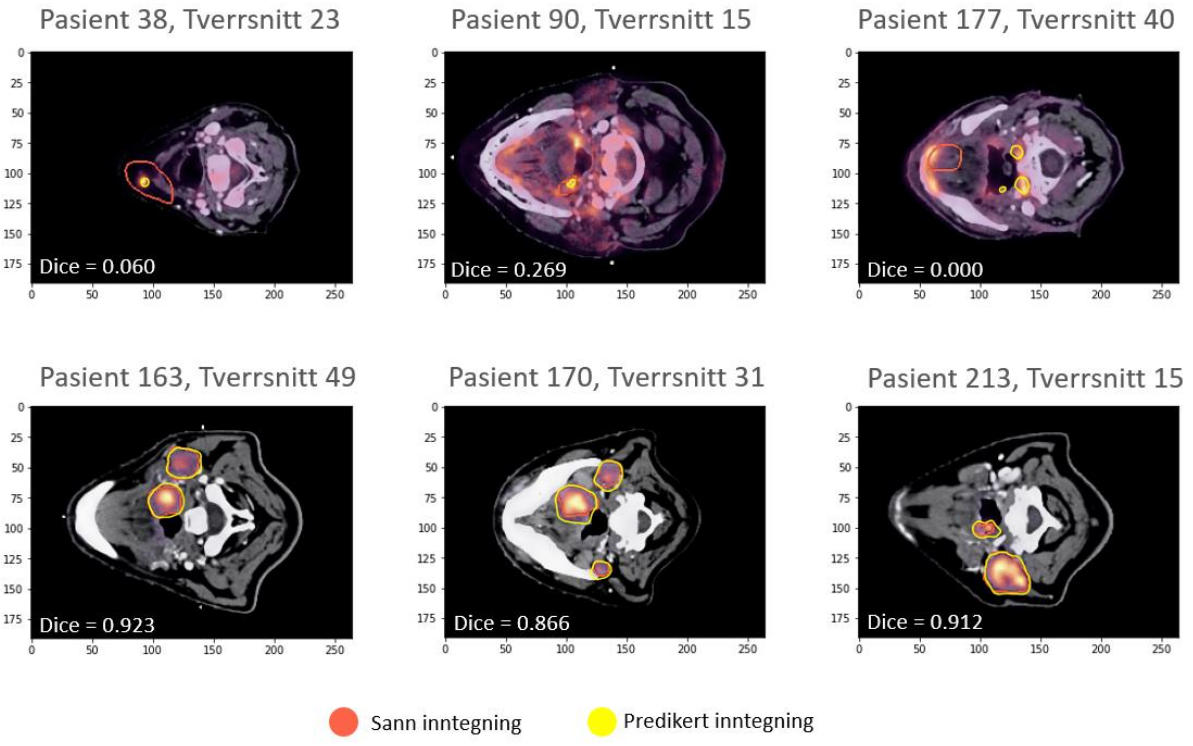
modellene uten og med bildeaugmentering ble gjennomsnittlig Dice-score per pasient beregnet til henholdsvis 0,639 ± 0,22 og 0,647 ± 0,26. I Figur 5.10 vises fiolinplott som illustrerer Dice-score per tverrsnitt for hver pasient for modellene vurdert på valideringssettet. Tabell 5.3 og fiolinplottene i Figur 5.10 viser at modellene har en høy ytelse på 12 av totalt 15 pasienter. Totalt sett har modellen med bildeaugmentering en høyere ytelse. Det er de samme tre pasientene (pasient 38, 90 og 177) som modellen presterer dårligere på. Av disse gjør modellen med bildeaugmentering det dårligere på to av tre pasienter. I Figur 5.11 og Figur 5.12 vises eksempler på inntegninger gjort av begge modellene på disse pasientene samt noen inntegninger med høy Dice-score fra andre pasienter. For modellene med lav Dice-score er det en liten eller ingen grad av overlapp mellom sann inntegning og predikert inntegning, mens for modellene med høy Dice-score er det en høy overlapp mellom sann og predikert inntegning.

***Tabell 5.3**: Dice-score per pasient for den beste modellen med og uten bildeaugmentering, evaluert på pasientene i valideringssettet.*

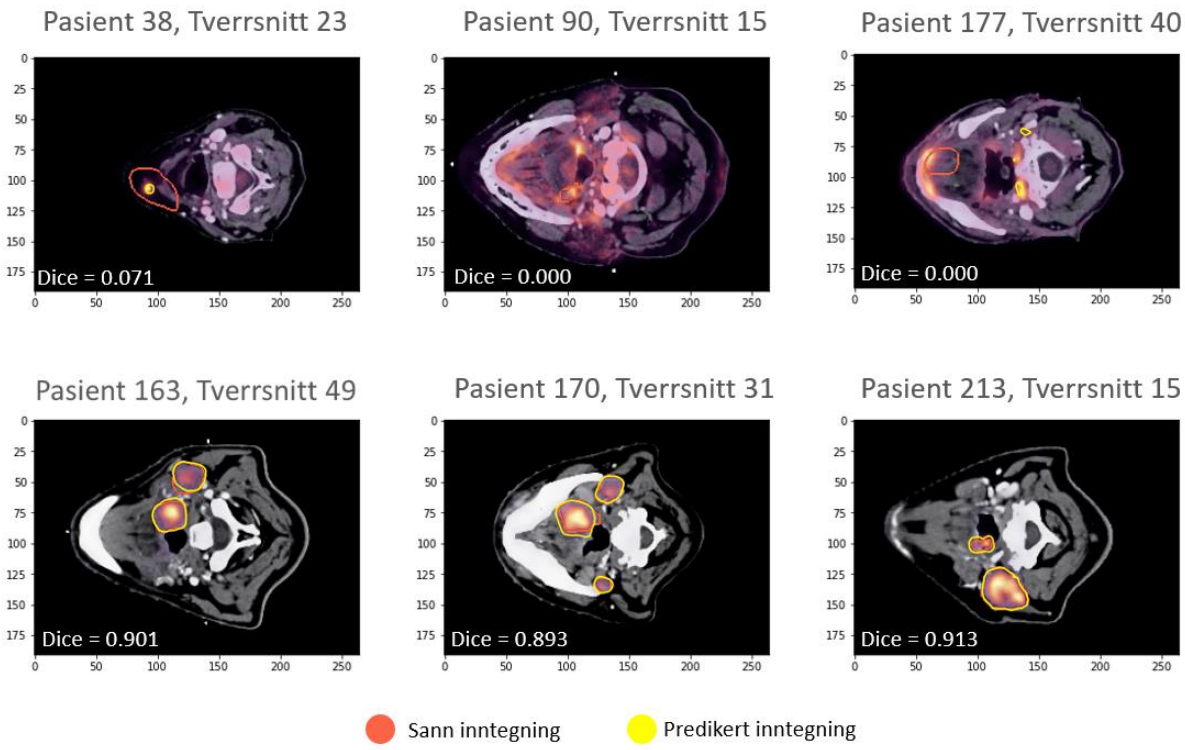
Pasient	Dice-score per pasient (beste modell uten augmentering)	Dice-score per pasient (beste modell med augmentering)
29	0,680	0,754
35	0,669	0,706
38	0,357	0,346
49	0,724	0,749
70	0,589	0,581
87	0,760	0,802
90	0,393	0,147
98	0,776	0,786
163	0,852	0,838
170	0,782	0,812
177	0,031	0,058
213	0,831	0,836
229	0,745	0,832
241	0,719	0,771
246	0,600	0,689



***Figur 5.10**: Fiolinplott som illustrerer fordelingen av Dice-score per tverrsnitt for hver pasient i valideringssettet for a) den beste modellen uten bildeaugmentering og b) den beste modellen med augmentering. Boksen i hver fiolin representerer observasjonene mellom første og tredje kvartil, og den hvite prikken angir medianen. De svarte vertikale strekene er utstikkere, observasjoner utenfor disse utstikkerne er utliggere (outliers). Lengden på fiolinene representerer variasjonen i gjennomsnittlig Dice-score hos hver enkelt pasient, mens tykkelsen representerer hvor mange bildetverrsnitt som får denne Dice-scoren.*

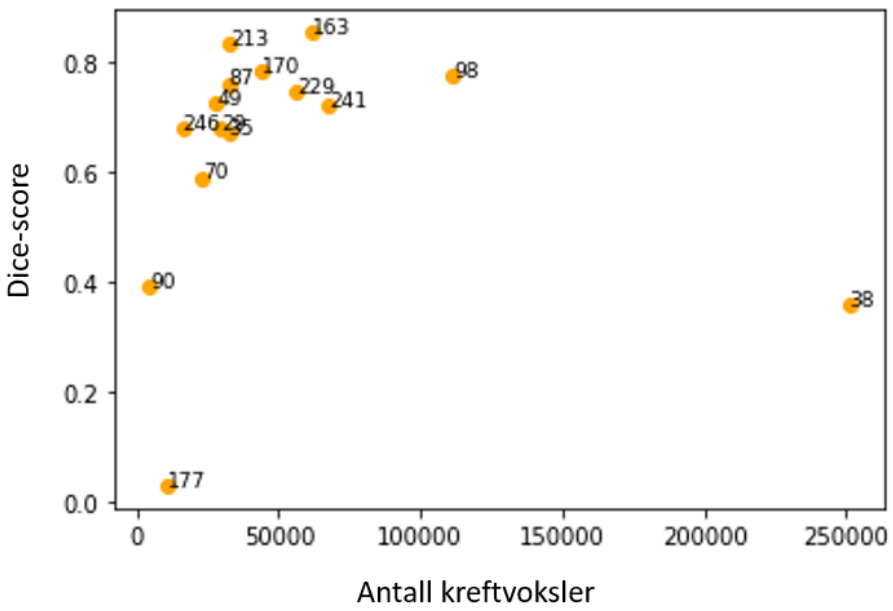


***Figur 5.11**: Eksempler på inntegninger for den beste modellen uten bildeaugmentering på valideringssettet. Den røde inntegninger er den sanne inntegningen og den gule inntegningen er den predikerte. I øverste rad vises inntegninger med lav Dice-score, mens den nederste raden viser inntegninger med høy Dice-score.*



***Figur 5.12**: Eksempler på inntegninger fra den beste modellen med bildeaugmentering på valideringssettet. Den røde inntegningen er den sanne inntegningen og den gule inntegningen er den predikerte. I øverste rad vises inntegninger med lav Dice-score, mens i den nederste raden vises inntegninger med høy Dice-score.*

For å undersøke om det var en sammenheng mellom ytelsen på modellen og størrelsen på påvirket vev (tumor og affiserte lymfeknuter), ble antall voksler klassifisert som kreft (fra sann inntegning) for hver pasient plottet mot Dice-score for tilhørende pasient for valideringssettet for den beste modellen uten augmentering. Dette spredningsplottet er vist i Figur 5.13. Her kan man se at for pasientene hvor modellen gir lav ytelse (pasient 38, 90 og 177) er tumorstørrelsen enten veldig liten eller veldig stor. For de små tumorstørrelsen er variasjonen i Dice-score størst.



Figur 5.13: *Spredningsplottet viser sammenhengen mellom påvirket vev (antall kreftvoksler for hver pasient) og Dice-score per pasient for den beste modellen uten bildeaugmentering vurdert på valideringssettet. En voksel har størrelsen 1×1×1 mm^3.*

5.3.2 Testsett

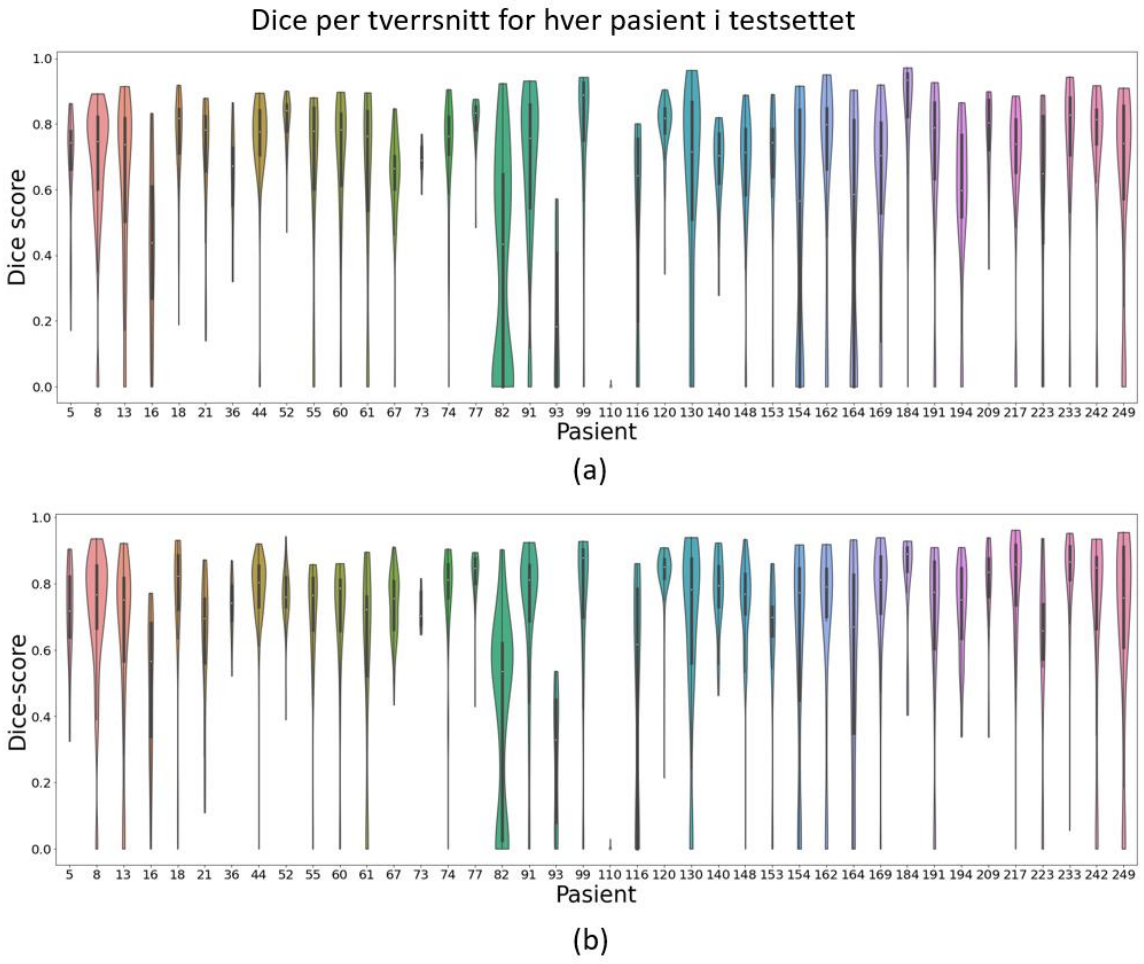
Den beste modellen, både med og uten bildeaugmentering ble kjørt på testsettet. Modellene uten og med bildeaugmentering fikk en gjennomsnittlig Dice-score per tverrsnitt på henholdsvis 0,643 ± 0,27 og 0,683 ± 0,25 vurdert på testsettet.

For testsettet ble det vurdert hvordan modellenes prestasjon var per pasient. I Vedlegg D i Tabell D.1 er det en oversikt over Dice-score per pasient for modellen med og uten bildeaugmentering for testsettet. For modellen uten og med bildeaugmentering ble gjennomsnittlig Dice-score per pasient beregnet til henholdsvis 0,714 ± 0,16 og 0,731 ± 0,16. I Tabell 5.4 vises en oversikt over de ulike ytelsesmålene per pasient for begge modellene. Sammenliknet med modellen uten bildeaugmentering får modellen med bildeaugmentering en høyere HD_{95} og Dice-score samt en lavere MSD. Dette indikerer at modellen med bildeaugmentering gjør færre alvorlige feil, men de alvorlige feilene som blir gjort er større, altså at den lengste avstanden mellom sann og predikert inntegning er større.

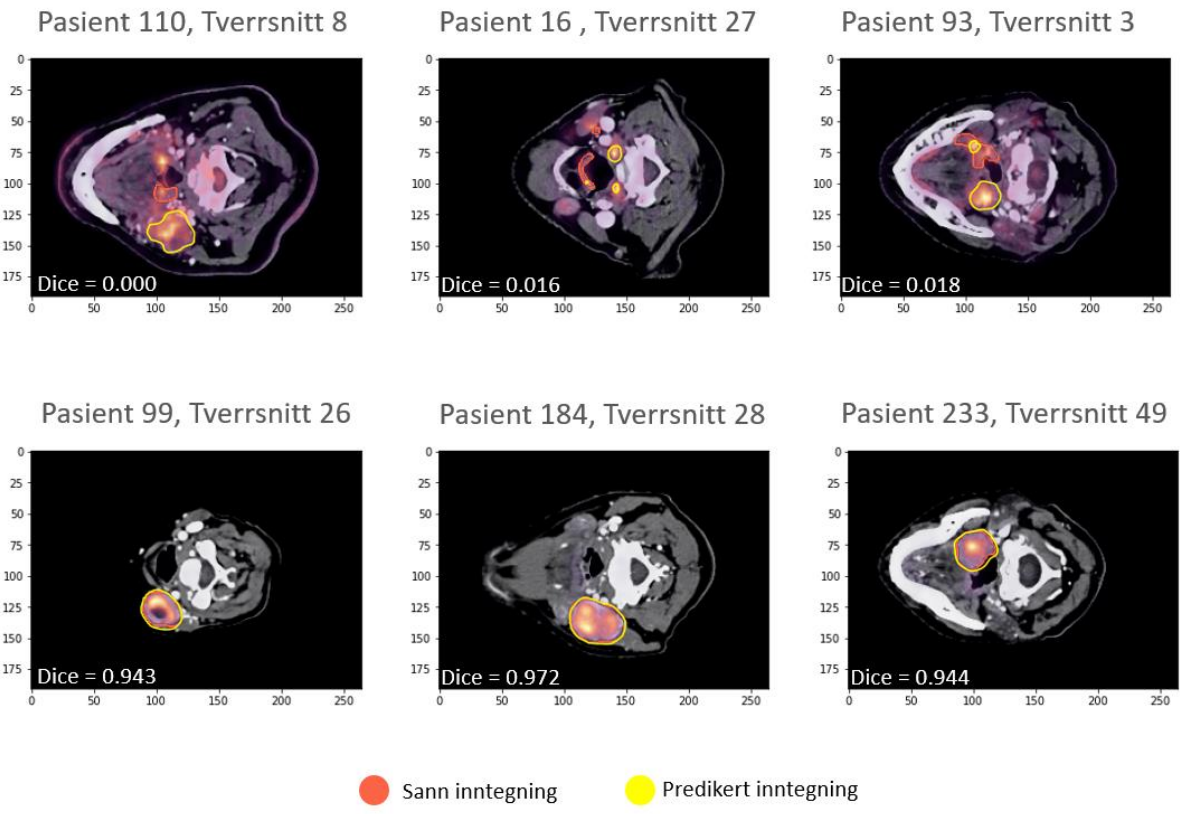
Tabell 5.4*: Ytelsesmål per pasient for de beste modellene med og uten bildeaugmentering vurdert på testsettet. Formatet er gjennomsnitt ± standardavvik.*

Modell	Dice	HD_{95} [mm]	MSD [mm]
Uten bildeaugmentering	0,714 ± 0,16	19,0 ± 13	1,94 ± 4,5
Med bildeaugmentering	0,731 ± 0,16	20,2 ± 15	1,68 ± 4,3

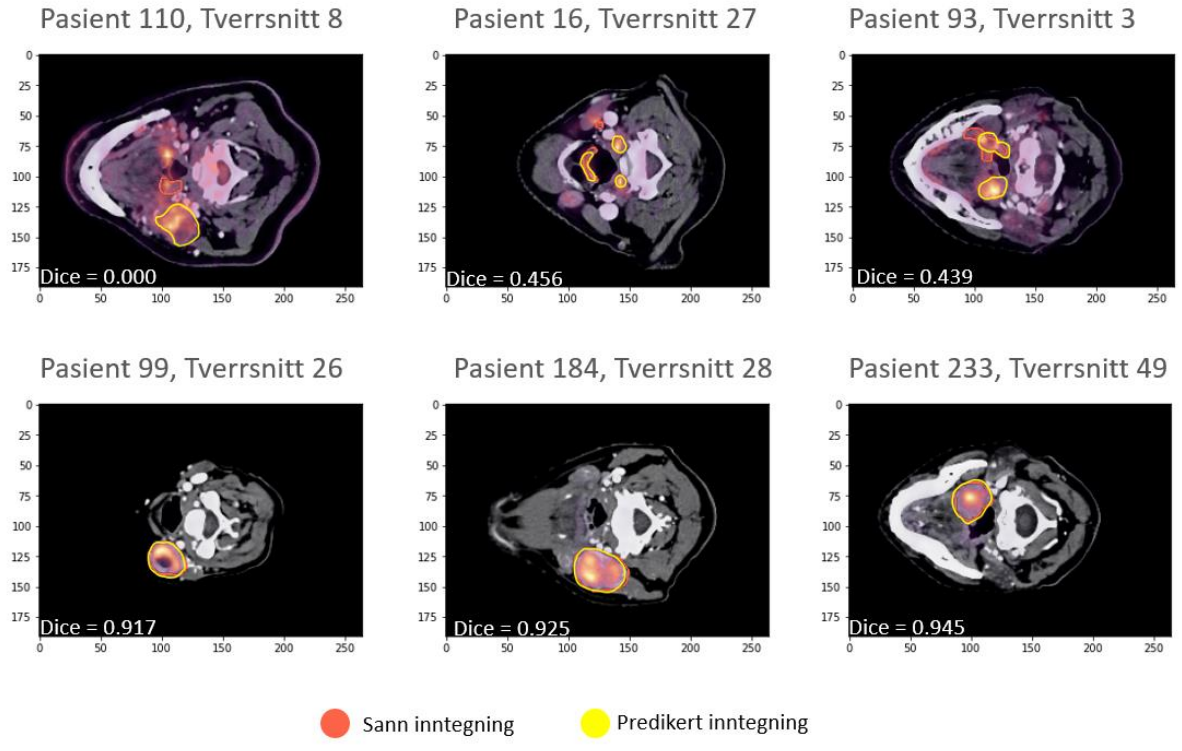
Figur 5.14 viser fiolinplot som illustrerer fordelingen av Dice-score per pasient for testsettet. Tabell D.1 og fiolinplottene i Figur 5.14 viser at modellene har en høy Dice-score for flere av pasientene. Begge modellene har lav ytelse på de samme pasientene (pasient 16, 93 og 110). I Figur 5.15 og Figur 5.16 vises eksempler på inntegninger gjort av modellen på disse pasientene, samt noen inntegninger fra andre pasienter hvor modellen gir en høy gjennomsnittlig Dice-score per pasient. For pasientene 16, 93 og 110 får modellen med bildeaugmentering en høyere Dice-score enn modellen uten bildeaugmentering. Modellene med lav ytelse har en liten eller ingen grad av overlapp mellom sann og predikert inntegning, mens modellene med høy ytelse har en stor grad av overlapp.



Figur 5.14: *Fiolinplott som illustrerer fordelingen av Dice-score per tverrsnitt for hver pasient i testsettet for a) den beste modellen uten bildeaugmentering og b) den beste modellen med augmentering. Boksen i hver fiolin representerer observasjonene mellom første og tredje kvartil, og den hvite prikken angir medianen. De svarte vertikale strekene er utstikkere, observasjoner utenfor disse utstikkerne er utliggere (outliers). Lengden på fiolinene representerer variasjonen i gjennomsnittlig Dice-score for hver enkelt pasient, mens tykkelsen representerer hvor mange bildetverrsnitt som får denne Dice-scoren.*

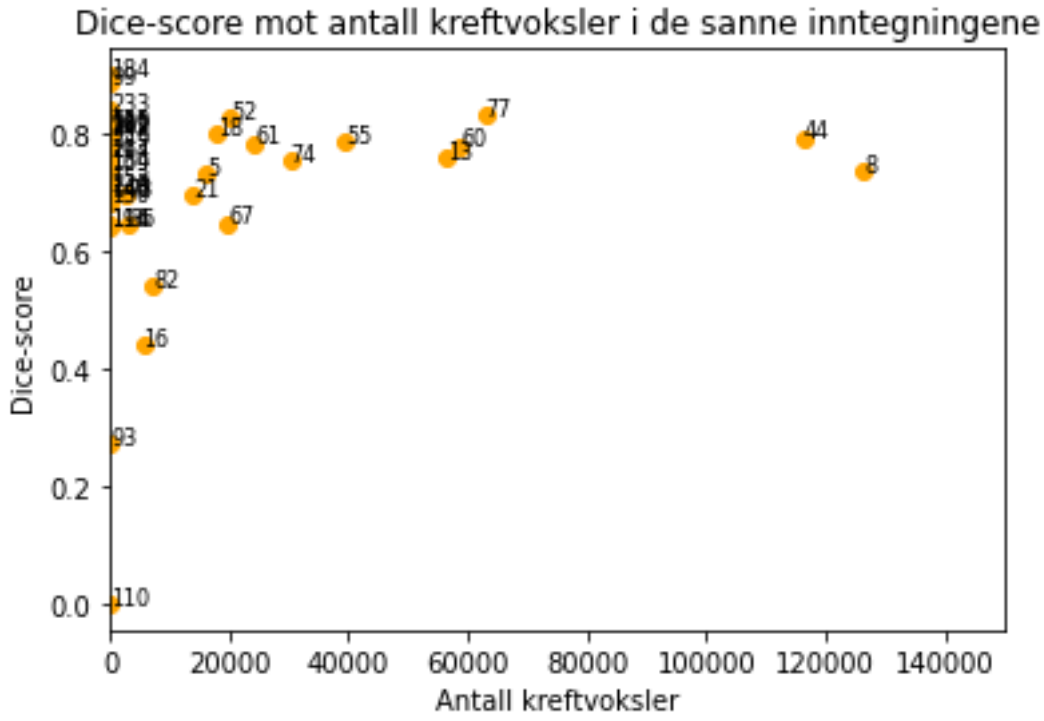


***Figur 5.15**: Eksempler på inntegninger av den beste modellen uten bildeaugmentering på testsettet. Den røde inntegningen er den sanne inntegningen og den gule inntegningen er den predikerte. I øverste rad vises inntegninger med lav Dice-score mens i nederste rad vises inntegninger med høy Dice-score.*



***Figur 5.16**: Eksempler på inntegninger av den beste modellen med bildeaugmentering på testsettet. Den røde inntegningen er den sanne inntegningen og den gule inntegningen er den predikerte. I øverste rad vises inntegninger med lav Dice-score mens i nederste rad vises inntegninger med høy Dice-score.*

Det ble for testsettet også sett på sammenhengen mellom Dice-score og størrelsen av påvirket vev (tumor og affiserte lymfeknuter) i sann inntegning hos hver pasient. I Figur 5.17 vises spredningsplottet for antall påvirkede voksler (fra sann inntegning) mot Dice-score per pasient for den beste modellen uten bildeaugmentering på testsettet. Her vises det at variasjonen i Dice-score per pasient er størst for små tumorvolum. Segmenteringen av større tumorvolum gir god nøyaktighet og høy Dice-score.



Figur 5.17: *Spredningsplott som viser sammenhengen mellom påvirket vev (antall kreftvoksler for hver pasient) og Dice-score per pasient for beste modell uten bildeaugmentering vurdert på testsettet. En voksel har størrelsen 1×1×1 mm³.*

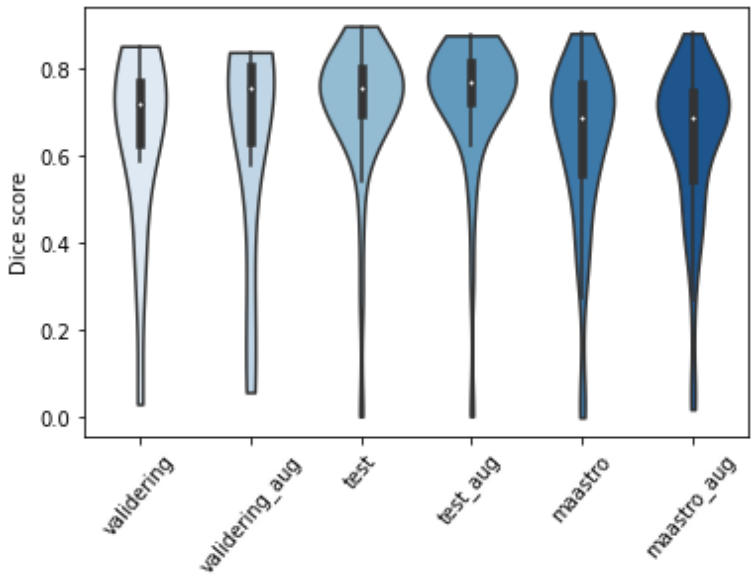
5.3.3 Maastro-testsett

Som forklart i delkapittel 4.1.1, ble den beste modellen (med og uten bildeaugmentering) også vurdert på det eksterne Maastro-testsettet. En oversikt over Dice-score per pasient for dette testsettet ligger i Vedlegg D i Tabell D.2. Gjennomsnittlig Dice-score per tverrsnitt for modellen uten og med bildeaugmentering ble henholdsvis 0,545 ± 0,31 og 0,564 ± 0,30. Gjennomsnittlig Dice-score per pasient ble 0,629 ± 0,19 og 0,635 ± 0,18 for henholdsvis modellen uten og med bildeaugmentering. I Tabell 5.5 vises en fullstendig oversikt over ytelsesmål per pasient for begge modellene fra inntegningen på Maastro-testsettet. Modellen med bildeaugmentering har en høyere Dice-score og HD_{95} og en lavere MSD, sammenliknet med modellen uten bildeaugmentering. Dette indikerer også her at modellen med bildeaugmentering gjør færre alvorlige feil, men at de alvorlige feilene som blir gjort er større.

I Figur 5.18 vises en sammenlikning av Dice-score per pasient for begge modellene vurdert på validerings- og testsettet fra OUS og Maastro-testsettet. Her vises det at fordelingen for Maastro-testsettet ligger lavere i Dice-score enn for validering- og testsettet fra OUS. Størrelsen til fiolinene er ulike fordi de forskjellige settene består av et ulikt antall pasienter. Størrelsen på boksene inne i fiolinene er ulike, noe som indikerer at variansen i Dice-score til de ulike settene er forskjellig. Resultatene fra Maastro-testsettet har større varians enn de andre settene.

***Tabell 5.5**: Ytelsesmål per pasient for de beste modellene uten og med bildeaugmentering vurdert på Maastro-testsettet. Formatet er gjennomsnitt ± standardavvik.*

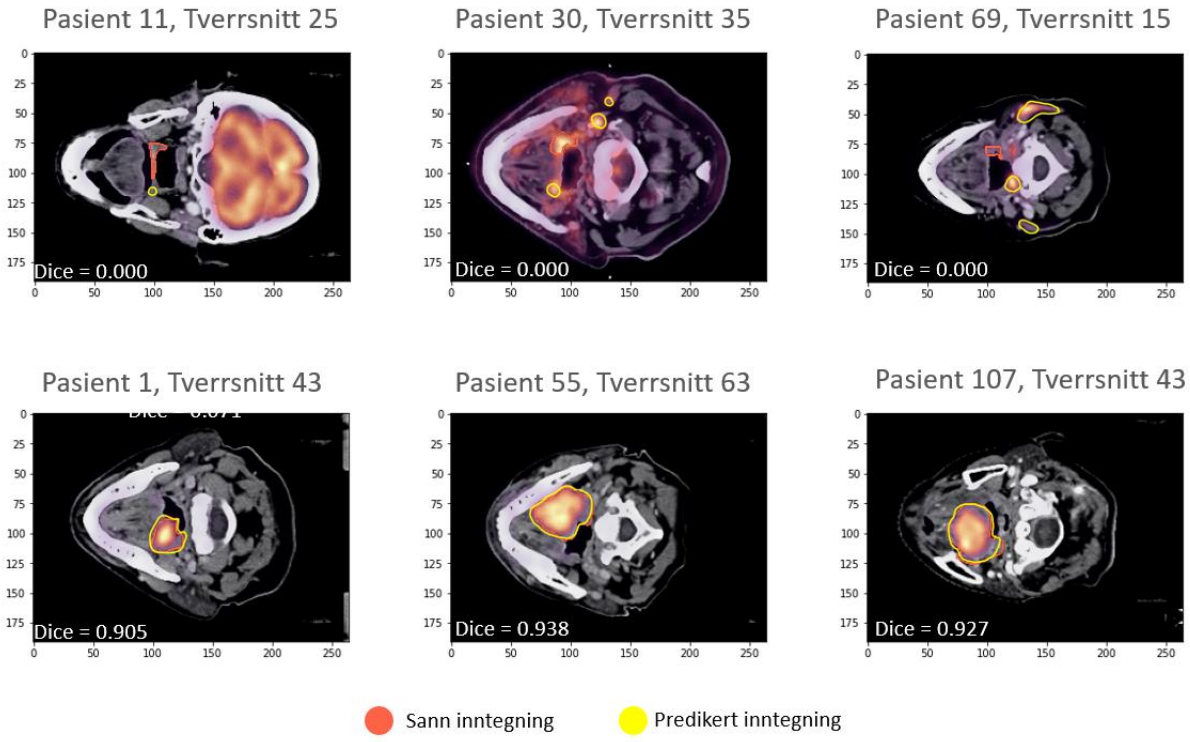
Modell	Dice	HD_{95} [mm]	MSD [mm]
Uten bildeaugmentering	0,629 ± 0,19	22,6 ± 16	2,23 ± 4
Med bildeaugmentering	0,635 ± 0,18	25,1 ± 19	2,10 ± 4,3



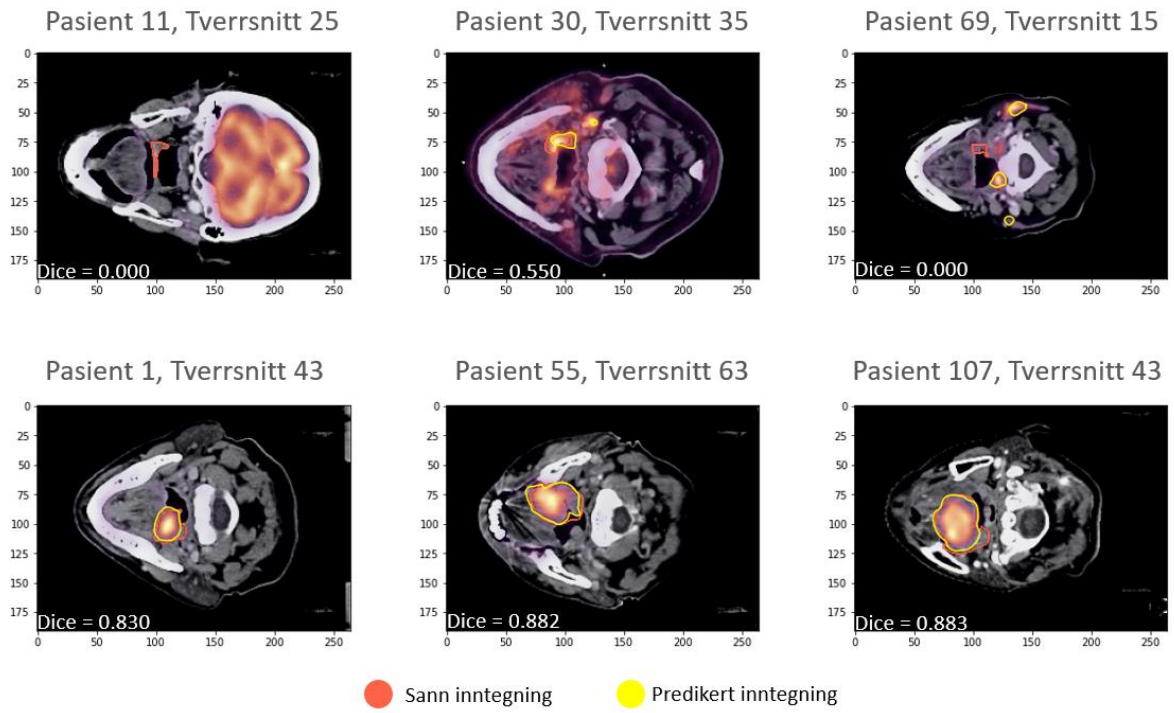
***Figur 5.18:** Fiolinplott som viser Dice-score per pasient for de beste modellene uten og med bildeaugmentering på validerings- og testsettet fra OUS og testsettet fra Maastro. Boksen i hver fiolin representerer observasjonene mellom første og tredje kvartil, og den hvite prikken angir medianen. De svarte vertikale strekene er utstikkere, observasjoner utenfor disse utstikkerne er utliggere (outliers). Lengden på fiolinene representerer variasjonen i Dice-score per pasient for de ulike settene, mens tykkelsen representerer hvor mange pasienter som har denne Dice-scoren. Tykkelsen er her forskjellig siden det er ulikt antall pasienter i de forskjellige settene. Valideringssettet har 15 pasienter, testsettet har 40 pasienter og Maastro-testsettet har 114 pasienter.*

I Figur D.1 og Figur D.2 i Vedlegg D vises fiolinplottene for Dice-score per pasient for henholdsvis den beste modellen uten og med bildeaugmentering vurdert på Maastro-testsettet. Lengden til fiolinene indikerer at det er stor varians i Dice-score for hver enkelt pasient. Her ser vi at modellene presterer generelt godt på de fleste pasientene, men svært dårlig på noen. Begge modellene presterer dårlig på blant annet pasient 5, 11, 18, 30, 45 og 69. Eksempler på

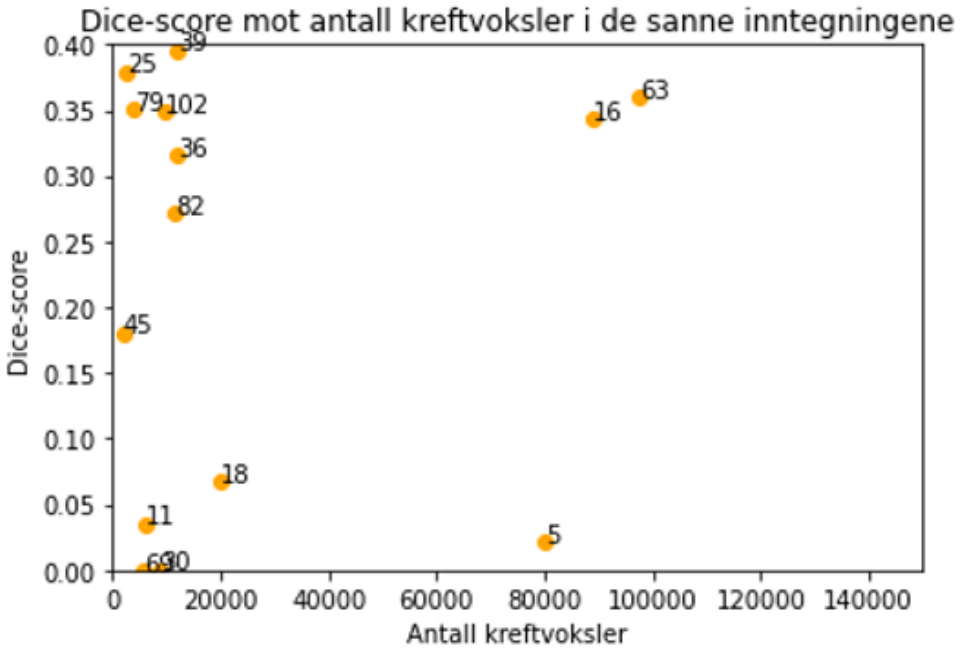
inntegninger for noen av disse pasientene kan sees i Figur 5.19 og Figur 5.20. Modellene som har en lav ytelse har ingen overlapp mellom sann og predikert inntegning, mens modellen som presterer godt har stor overlapp mellom sann og positiv inntegning. I Figur 5.21 vises spredningsplottet for størrelsen av påvirket vev (tumor og affiserte lymfeknuter) for et utvalg av pasientene mot Dice-score per pasient. Her kan man se at for små tumorstørrelser gir modellen enten en lav eller høy ytelse. Etter hvert som tumorstørrelsen øker, øker også Dice-scoren.



Figur 5.19: *Eksempler på inntegninger av den beste modellen uten bildeaugmentering på Maastro-testsettet. Den røde inntegningen er den sanne inntegningen og den gule inntegningen er den predikerte. Den øverste raden viser inntegninger med lav Dice-score, mens den nederste raden viser inntegninger med høy Dice-score.*



***Figur 5.20**: Eksempler på inntegninger av den beste modellen med bildeaugmentering på Maastro-testsettet. Den røde inntegninger er den sanne inntegningen og den gule inntegningen er den predikerte. Den øverste raden viser inntegninger med lav Dice-score, mens den nederste raden viser inntegninger med høy Dice-score.*



***Figur 5.21**: Spredningsplott som viser sammenhengen mellom påvirket vev (antall kreftvoksler for hver pasient) og Dice-score per pasient for et utvalg av pasienter for beste modell uten bildeaugmentering vurdert på Maastro-testsettet. En voksel har størrelsen 1×1×1 mm^3.*

Kapittel 6: Diskusjon

Målet med denne masteroppgaven var å undersøke og optimalisere konvolusjonsnettverket VoxResNet for bruk til automatisk segmentering av kreftsvulster i PET/CT-bilder for pasienter med hode- og halskreft. Optimaliseringen ble gjort ved å undersøke fire parametere, og finne de nivåene av parameterne som sammen ga høyest Dice-score. I følge Zijdenbos et al. [66] kan en Dice-score på 0,7 eller høyere bli ansett som en god overlapp og dermed en god modell. Modellen med høyest Dice-score oppnådde en score på 0,714 per pasient, som er et lovende resultat, men med rom for forbedringer. Det ble også sett på effekten av bildeaugmentering ved å trene modellen på datasettet preprosessert med ulike bildeaugmenteringsteknikker. Denne modellen oppnådde en Dice-score per pasient på 0,731. Videre blir nettverket sammenliknet med U-Net-arkitekturen brukt i Moe et al. [17] og Groendahl et al. [18]. Dette kapittelet vil diskutere resultatene fra optimaliseringsprosessen med valg av parametere, modellens ytelse, sammenlikning med tidligere arbeid og forslag til videre arbeid.

6.1 Valg av parametere

Fra resultatene av de statistiske testene i delkapittel 5.2.2, er det tydelig at valget av nivået til de ulike parameterne har en betydelig effekt på modellens ytelse. I utgangspunktet var planen å velge de beste parameterne ved å bruke ANOVA for å se om det var signifikante forskjeller mellom nivåene, etterfulgt av post-hoc-tester for å finne ut hvor forskjellen eventuelt lå. Etter å ha undersøkt datamaterialet, og prøvd ulike transformasjoner, var det klart at residualene ikke var normalfordelt. På grunn av dette var ikke antagelsene for ANOVA oppfylt, og kunne dermed ikke brukes. Hver parameter ble derfor vurdert individuelt.

6.1.1 Batchnormalisering

I følge Wilcoxon signed rank-testen vil modellen få en høyere Dice-score ved å inkludere batchnormaliseringslag i modellen. Nettverk som trener på data som er skalert vil lære bedre og gjør at modellen kan generalisere til ny data bedre. For dype nevrale nettverk vil fordelingen til inputet endres under treningsprosessen, fordi parameterne fra foregående lag endres. Det vil derfor være en fordel å bruke batchnormaliseringslag i tillegg til eller fremfor skalering av dataen på forhånd. I følge Ioffe og Szegedy [58], vil bruken av batchnormalisering også føre til at nettverket kan trene raskere. Dette kommer av at det tillates bruk av en høyere læringsrate, noe som gjør at modellen trenes med færre treningssteg.

Siden effektstørrelsen for batchnormalisering var stor, vil inkluderingen av dette i modellen ha en betydelig effekt på ytelsen.

6.1.2 Dropout

Dropoutraten som ble valgt å bruke i modellen var 0 eller 0,5. Det vil altså si at enten så ble ingen eller 50 prosent av nodene droppet. Valg av dropoutrate som ble testet ble satt til 0,5 fordi en for lav rate vil trolig ikke ha en stor nok regulariserende effekt, mens en for høy rate vil føre til at modellen ikke lærer egenskapene godt nok.

Å inkludere dropout vil som sagt ha en regulariserende effekt på modellen, noe som kan hjelpe mot overtilpassing. Aktiveringsfunksjonen ReLU har en tendens til å overtilpasse, så i tilfeller hvor denne aktiveringsfunksjonen brukes, vil det være en fordel å inkludere regulariseringsteknikker, som f.eks. dropout [37]. Friedmanntesten viste derimot at modellene som ikke inkluderte dropout presterte generelt høyere enn modellene med dropout. Årsaken til dette kan være, som Ioffe og Szegedy [58] viste, at når batchnormaliseringslag er inkludert i modellen, vil det i noen tilfeller ikke være nødvendig med dropout, da batchnormaliseringslagene i seg selv har en regulariserende effekt. Effektstørrelsen for dropout var her stor, så valget vil ha en betydelig effekt på ytelsen.

6.1.3 Antall filtre

Friedmantesten for antall filtre viste at det var det nivået med færrest antall filtrene som ga den høyeste ytelsen. Dette var 48 filtre. Effektstørrelsen var her moderat, så valget av antall filtre som brukes vil ha en moderat betydning på ytelsen. En modell med få filtre vil være mindre kompleks, noe som kan redusere risikoen for overtilpassing. Valg av antall filtre vil også påvirke treningstiden, en modell med færre filtre vil trene raskere. Med en Dice-score på henholdsvis 0,714 og 0,731 for modellene uten og med augmentering, ser det imidlertid ut som at VoxResNet-modellene med 48 antall filtre klarer å trekke ut de mest relevante egenskapene.

6.1.4 Læringsrate

Læringsraten som ga den høyeste gjennomsnittlige Dice-scoren var 10^{-4}. Her var også effektstørrelsen stor, så forskjellen i ytelse vil altså være markant for de ulike nivåene av læringsraten. Når læringsraten skal bestemmes må to senarioer tas i betraktning. En for liten læringsrate kan føre til unødvendig mange iterasjoner og minimumet som nås kan være et lokalt minimum [36]. Dersom læringsraten er for stor kan det hende at man ikke finner et

minimum, og punktet bare er et tilfeldig punkt langs tapsfunksjons-kurven. Valg av læringsrate påvirker også tiden modellen bruker på trening. En høy læringsrate vil tillate at modellen kan lære raskere, på bekostning av at de endelige vektene ikke er fullstendig optimale. En lav læringsrate tillater derimot modellen å lære gunstigere vekter, men vil bruke signifikant lengre tid på treningen [67]. Og som forklart tidligere vil ifølge Ioffe og Szegedy [58] inkluderingen av batchnormaliseringslag i modellen tillate bruk av en høyere læringsrate.

6.1.5 Begrensinger ved valg av beste modell

En begrensning i den statistiske analysen foretatt i denne masteroppgaven er at hver parameter ble sett på individuelt, og at det ikke ble undersøkt interaksjoner mellom de ulike parameterne. Dette ble ikke gjort da den opprinnelige planen med en N-veis ANOVA ikke kunne gjennomføres grunnet brudd på betingelsene. Interaksjoner mellom parameterne bør undersøkes da det er en tydelig sammenheng mellom flere av parameterne, slik som at batchnormalisering i noen tilfeller kan eliminere behovet for dropout og tillater å ta i bruk en høyere læringsrate sammenliknet med en modell uten batchnormalisering.

En annen faktor som kan ha hatt en påvirkning av resultatene i de statistiske testene, er at noen av kombinasjonene av parameternivå fikk ekstremt lave verdier. Noen av eksperimentene fikk en Dice-score på tilnærmet lik null. Dette kan skyldes at kombinasjonen av parameternivå rett og slett ikke fungerte sammen eller at det har vært en feil i konfigurasjonsfilen. Alle eksperimentene som fikk denne lave scoren, hadde en læringsrate på 10^{-3}. Dette har trukket ned gjennomsnittet av Dice-scoren til eksperimentene med denne læringsraten som ble brukt i de statistiske testene for å bestemme beste nivå av læringsrate. Dersom det viser seg at grunnen til at disse modellen fikk så lav Dice-score var en feil i eksperimentet, tyder dette på at resultatet for valg av beste læringsrate ville vært annerledes.

De statistiske testene ble gjennomført på Dice-score per tverrsnitt som datagrunnlag. Siden det er mer interessant å se på hvordan modellene presterer på hver hele pasient, ville det vært nyttig å vurdere de statistiske testene med Dice-score per pasient også. Å vurdere de statistiske testene basert på Dice-score per pasient vil mulig gi en modell som er bedre tilpasset vurdering per pasient. Da kan det hende at utfallet av valget av parameternivå ville vært annerledes. Det er derimot kun 15 pasienter i valideringssettet, noe som resulterer i få datapunkter for Dice-score per pasient sammenliknet med Dice-score per snitt. Potensielle utliggere («outliers») vil i da påvirke resultatet i større grad.

6.1.6 Bildeaugmentering

Bildeaugmentering ble lagt til for å øke mengden treningsdata ved å legge til transformerte versjoner av den eksisterende dataen, slik at det skapes mer variasjon i datasettet. Medisinske bilder vil i utgangspunktet allerede være varierte, siden ingen pasienter er like og av den grunn vil ingen bilder heller være like. Det vil uansett være en fordel å legge til flere varianter av bildene da dette kan øke sannsynligheten for at nettverket er kjent med ny, usett data.

Resultatene over Dice-score per pasient for testsettet og Maastro-testsettet i Tabell D.1 og Tabell D.2, viste at å utvide datasettet med bildeaugmentering hadde en positiv effekt på ytelsesmålet. Ved å gjøre datasettet større og skape mer variasjon vil modellen bli mer generell, og i større grad forhindre overtilpassing. Figur 5.9, av trening- og valideringskurvene til den beste modellen med og uten bildeaugmentering, viser at modellen trent med bildeaugmentering er noe mindre overtilpasset. Å bruke bildeaugmentering til å utvide datasettet er en effektiv, lite tidkrevende og lite kostbar metode, sammenliknet med utvidelse ved å ta bilder av flere pasienter.

Valg av teknikker som brukes for bildeaugmentering avhenger av hvilken type data som brukes [68]. I noen tilfeller vil ikke alle transformasjoner gi anatomisk mening for modellen, noe som heller kan gjøre modellen forvirret. Årsaken til at modellen kan blir forvirret av dette er fordi det er bilder som ikke vil forekomme naturlig, og er derfor ikke tilfeller som modellen vil komme over senere. Så det at modellen lærer seg egenskapene til transformasjoner med en unaturlig orientering vil også være til lite nytte. Eksempler på dette for datasettet med hode- og halskreft er horisontal flipp (om y-aksen) eller en rotasjonsgrad på over 90 grader. Dette gir bildene en unaturlig orientering, noe som kan gjøre at modellen blir forvirret. Det er derfor viktig å ha innblikk i hva datasettet består av, og velge transformasjoner ut ifra hvilken oppgave modellen skal brukes til.

I HECKTOR-utfordringen [14] er det flere av deltagerne som tar i bruk bildeaugmenteringsteknikker. Zhu et al. [69] presenterte et bidrag bestående av to steg. Først brukes et klassifiseringsnettverk basert på ResNet til å velge ut tverrsnittene som kan inneholde kreftsvulster. Deretter brukes en 2D U-Net-arkitektur til selve inntegningen. Videre tok de i bruk standard bildeaugmentering som forminskning og tilfeldig forskyvning på 10 mm i *x*- og *y*-retning, og oppnådde med dette en gjennomsnittlig Dice-score per pasient på 0,644. Yuan [70] presenterte et *Scale Attention Network* (SA-Network) basert på U-Net-arkitekturen. Augmenteringsteknikkene som ble tatt i brukt i dette bidraget var forskyvning til

høyre og venstre, speiling om aksene, rotasjon og skalering. Dette resulterte i en Dice-score per pasient på 0,732 testsettet.

6.2 Valg av ytelsesmål

Bildene i datasettet har en ubalanse mellom klassefordelingen, da det er en stor andel piksler som ikke er kreft og en mindre andel av pikslene som er kreft. Det vil derfor være gunstig å velge Dice-score for å vurdere ytelsen til modellene, da denne tar hensyn til ubalanse mellom klassene. Dice-score tar hensyn til klassebalansen ved å ikke ta med sanne negative i beregningen. Ved å vurdere Dice-score per pasient vil ytelsen legge mer vekt på den totale inntegningen per pasient, istedenfor små tumorområder per tverrsnitt som modellen ikke klarer å tegne inn. Flere tidligere studier rapporterer ytelse i Dice-score per pasient, så for å vurdere resultatene fra denne masteroppgaven opp mot tidligere arbeid ble Dice-score per pasient beregnet.

En begrensning med bruken av Dice-score er at den er avhengig av størrelsen på den sanne inntegningen og at den ikke tar hensyn til avstanden mellom den predikerte og den sanne inntegningen [16]. Små tumorer har et lavt antall sanne positive piksler, og det ser ut som at disse ofte får en lav Dice-score. Siden antallet sanne positive er lav, vil forholdet mellom predikert inntegning og sann inntegning ha en større negativ effekt på Dice-score enn for større tumorer. For inntegninger som inneholder både tumorer og affiserte lymfeknuter, vil typisk tumoren være stor og lymfeknutene være mindre. Inntegningen av tumoren vil bestå av en stor andel av de sanne positive pikslene. En modell som klarer å tegne inn tumoren, men ikke lymfeknutene vil da fortsatt oppnå en høy Dice-score. På denne måten kan modellen få en høy ytelse, selv om små strukturer ikke er tegnet inn. I tilfellene der målet er å segmentere både kreftsvulster og affiserte lymfeknuter bør man på bakgrunn av dette vurdere ytelsen basert på flere ytelsesmål enn bare Dice-score. Det vil da være hensiktsmessig å i tillegg bruke et ytelsesmål som tar hensyn til distansen mellom sann inntegning og predikert inntegning, slik som for eksempel Hausdorff-distanse eller MSD forklart i delkapittel 3.4.3. En svakhet med Hausdorff-distanse er at det er sensitivt for små utliggere [16] og 95. persentil Hausdorff-distanse bør derfor brukes. Bruken av Hausdorff-distanse eller MSD som ytelsesmål er ment for å skaffe ytterligere informasjon om kvaliteten på den automatiske inntegningen, og er ikke ment for å erstatte Dice-score.

6.3 Modellytelse

6.3.1 Begrensninger

Både modellen med og uten bildeaugmentering har en relativt lav Dice-score for enkelte pasienter i både validerings- og testsettet samt det eksterne Maastro-testsettet. Årsaken til dette kan blant annet skyldes ulike PET-signaler i bildene, altså opptak av FDG, hos hver enkelt pasient eller unormale Hounsfield verdier innenfor den sanne inntegningen. Dette fører til atypiske bilder, som modellen vil ha problemer med å kjenne igjen. I tilfeller der kreftsvulsten har lavt PET-signal innenfor den sanne inntegningen, vil det være vanskelig for modellen å detektere at det er en tumor eller lymfeknute der. Fra eksemplene som er gitt på inntegninger i Figur 5.11, Figur 5.12, Figur 5.15, Figur 5.16, Figur 5.19, Figur 5.20 kan dette tydelig observeres. Modellene har en nøyaktig segmentering der PET-signalet er sterkt (lyst), og en dårlig segmentering der PET-signalet er svakt (rød/rosa). Det er tydelig å se at det er likheter mellom bildene som modellene presterer dårlig eller nøyaktig på. En likhet mellom bildene som modellen prestere dårlig på, er det svake PET-signalet over store områder i bildet. I mange av bildene som modellen presterer godt på, er det derimot et sterkt PET-signal som kun er konsentrert til området med kreftsvulst. For Maastro-testsettet kan en årsak til at modellen feiler være at PET/CT-bildene eller de sanne inntegningene er atypiske for modellene, som igjen gjør at modellene predikerer dårlig.

Videre er det også tydelig fra Figur 5.13, Figur 5.17 og Figur 5.21, at størrelsen på sann inntegning har en innvirkning på hvor nøyaktig segmentering modellen kan gi. Årsaken til at pasientene med lave tumorvolum kan gi lav ytelse, skyldes at små tumorvolum ofte er affiserte lymfeknuter. Små lymfeknuter kan ofte ha et lavt FDG-opptak, og som forklart tidligere er det ofte her modellen feiler.

6.3.2 Eksternt testsett – Maastro

Modellen ble testet på et eksternt testsett for å vurdere hvordan modellene ville prestere på helt usett data og finne ut hvor generelle modellene var. Gjennomsnittlig Dice-score per pasient ble beregnet til 0,629 for modellen uten bildeaugmentering og 0,635 for modellen med bildeaugmentering på Maastro-testsettet. Denne ytelsen er lavere enn ytelsen på OUS-testsettet. Dette kan komme av små variasjoner mellom OUS-datasettet og Maastro-testsettet. Disse små variasjonene kan skyldes at dataen er hentet inn fra to ulike sykehus som kan ha noe forskjellige prosedyrer rundt billedtakningen og inntegningen av kreftsvulstene. Selv om det er de samme type bilder, så kan det være små forskjeller som f.eks. plasseringen av

pasienten, som kan gjøre at bildene i datasettene blir litt ulike. Noen av pasientene skiller seg fra pasientene i OUS-dataen ved at de har en liten, karakteristisk sann inntegning som har et lavt PET-signal. Retningslinjene som radiologene bruker for inntegning kan også variere fra sykehus til sykehus, noe om også bidrar til at datasettene blir litt forskjellige. Resultatene av testingen på det eksterne testsettet er som forventet, da det ikke er mulig at noe av informasjonen fra det eksterne testsettet kan ha lekket over i modellen under trening.

En metode som kan brukes for å mulig øke ytelsen ytterligere på det eksterne testsettet, er å legge noen bildetversnitt fra det eksterne Maastro-testsettet inn i OUS-datasettet for kalibrering av modellen. Da vil modellen trene og validere på noen av pasientene fra det eksterne testsettet, og vil potensielt lære seg egenskapene som er spesielle for det eksterne testsettet. Dette vil gjøre at modellen mulig kan lære seg de små variasjonene som skiller datasettene og dermed kunne gjøre at modellen presterer enda bedre på testsettet.

6.4 Sammenlikning med baselinemodell

Sammenlikningsmodellen har i Moe et al. [17] oppnådd en gjennomsnittlig Dice-score per pasient på 0,71 og i Groendahl et al. [18] en gjennomsnittlig Dice-score per pasient på 0,75. Datasettet brukt i Moe et al. [17] er det samme datasettet som bukes i denne masteroppgaven, mens i Groendahl et al. [18] brukes en noe modifisert versjon av datasettet. Modellen undersøkt i denne masteroppgaven oppnådde en gjennomsnittlig Dice-score per pasient på 0,714. Ved å trene modellen med bildeaugmentering oppnådde den en gjennomsnittlig Dice-score per pasient på 0,731. Modellen i Moe et al. oppnådde en HD_{95} på 21,2 mm og en MSD på 1,8. For VoxResNet-modellene får begge modellene en lavere HD_{95} på henholdsvis 19,0 mm og 20,2 mm for modellen uten og med augmentering. MSD er derimot høyere for modellen uten bildeaugmentering og lavere for modellen med augmentering, sammenliknet med MSD for modellen i Moe et al.[17].

De to sammenlikningsmodellene fra Moe et al. [17] og Groendahl et al. [18] har oppnådd forskjellige ytelser, selv om de har benyttet samme U-Net-arkitektur. Dette kan komme av flere grunner, men i hovedsak at de har benyttet ulike nivåer for ulike parameterer og ulik preprosessering av bildene. De to modellene har benyttet ulike læringsrater, forskjellige intensitetsvindu for CT-bildene og ulik beskjæring av bildene. I Moe et al. [17] brukes en læringsrate på 10^{-4} og et intensitetsvindu med en vindu-bredde på 200 med vindu-senter i 70, mens i Groendahl et al. [18] brukes 10^{-5} og et intensitetsvindu med en vindu-bredde på 100 og vindu-senter i 60. Videre bukte Groendahl er al. [18] 5-foldig kryssvalidering for å trene

modellen, noe som gir forskjellige bilder i hvert trenings- og valideringssett. I Groendahl et al. [18] ble bildene beskjært slik at bildene bare inneholder kreftsvulstene og affiserte lymfeknuter. Dette vil gjøre inntegningsoppgaven mindre krevende, og kan dermed sammen med bruken av andre parametere, intensitetsvindu og kryssvalidering, resultere i en høyere ytelse. Av denne grunn vil det ikke være helt rettferdig å direkte sammenlikne ytelsen for modellen i denne masteroppgaven med ytelsen oppnådd i Groendahl et al. [18]. Ut over dette er det det samme datasettet som brukes i sammenlikningsmodellene, og i denne masteroppgaven brukes en normalisert versjon av samme datasett.

VoxResNet-modellen oppnår en ytelse som kan måle seg med ytelsen for U-net-arkitekturen, som regnes som state-of-the-art innenfor segmentering i medisinske bilder [47]. VoxResNet-arkitekturen vil med sine residualkoblinger tillate at informasjon kan direkte propagere fremover og bakover i nettverket [47]. En informasjonsflyt som denne kan forhindre degraderingsproblemet og overtilpassing. En annen fordel med residualkoblingene i VoxResNet-modellen er at de ikke øker antallet trenbare parametere til modellen.

6.5 Tidligere arbeid

Det har de siste årene vært flere ulike tilnærminger og forsøk på automatisk inntegning av GTV og affiserte lymfeknuter i medisinske bilder for bruk i planleggingen av kreftbehandling. Gou et al. [16] har undersøkt bruken av 3D Dense-Net for automatisk segmentering av hode- og halskreft GTV i PET/CT-bilder med størrelse $128 \times 128 \times 48$. Modellen oppnådde en gjennomsnittlig Dice-score per pasient på 0,71. Sammenliknet med modellen i denne masteroppgaven har Gou et al. [16] en annen størrelse på inputbildene samt et annet intensitetsvindu. Størrelsen til inputbildene kan også ha en innvirkning på ytelsen. Lin et al. [12] har også som forklart tidligere oppnådd gode resultater med sitt 3D VoxResNet, med en Dice-score per pasient på 0,79. 2D VoxResNet-modellen som brukes i denne masteroppgave skiller seg noe fra modellen brukt i Lin et al. [12]. Det er blant annet ikke brukt den samme optimaliseringsalgoritmen. I modellen som brukes i denne masteroppgaven ble AdaDelta erstattet med Adam. Adam er en utvidelse av AdaDelta, som også inkluderer momentum. Dette er en fordel som gjør at modellen kan konvergere raskere [42].

I HECKTOR-utfordringen oppnådde flere av deltagernes bidrag lovende resultater. Dataen som ble brukt i utfordringen bestod av PET/CT-bilder med størrelse $144 \times 144 \times 144$, samlet inn fra fem sykehus og bestod av 254 pasienter. Dette ble delt opp i en treningskohort på 201 pasienter og en testkohort på 53 pasienter. Dice-score, presisjon og sensitivitet ble brukt for å

sammenlikne de ulike bidragene i konkurransen. 21 deltagere leverte inn ulike modeller som oppnådde Dice-scorer som varierte fra 0,561 til 0,759. Iantsen et al. [71] har i sitt bidrag til utfordringen utviklet en modell basert på U-Net-arkitekturen med residualkoblinger og *Squeeze and Excitation* (SE) normalisering. SE-normaliseringen ble tidligere utviklet av samme forfatter for segmentering av hjernetumorer. Bildeaugmentering ble ikke tatt i bruk. Resultatet på testsettet ble oppnådd med et ensemble på åtte modeller, hvor treningen og valideringen var på ulike delinger av treningssettet. Fire modeller ble trent med 4-foldet kryssvalidering og de resterende fire modellene ble trent med en tilfeldig trening/validering-splitt. Resultatet ble beregnet som et gjennomsnitt over de åtte modellene. Modellen oppnådde en Dice-score per pasient på 0,759 på testsettet og fikk med dette det høyeste ytelsesmålet blant 21 lag [71].

Det ble i HECKTOR-utfordringen kun sett på en type hode- og halskreft (oropharynx), og området hvor denne befinner seg er avgrenset med en boks i bildene. Det var kun tumorvolum som skulle tegnes inn, og affiserte lymfeknuter ble ikke tatt hensyn til. Det samme gjelder for VoxResNet-modellen brukt i Lin et al. [12]. Modellen i denne masteroppgaven har ikke en slik avgrensningsboks for området med kreft og ser på både tumor og affiserte lymfeknuter, noe som gjør inntegningsproblemet mer komplisert. Det vil av denne grunn ikke være rettferdig å direkte sammenlikne modellene fra denne masteroppgaven med modellen trent på bilder med avgrensningsbokser og som kun tegnet inn tumorvolum. Det anbefales derfor at direkte sammenlikning gjøres med modeller som har like forutsetninger og som ser på både tumorvolum og affiserte lymfeknuter, slik som i modellen i Moe et al. [17]. Til tross for at det ikke er helt rettferdig å sammenlikne direkte med modellene nevnt i dette delkapittelet, gav VoxResNet-modellene uten og med bildeaugmentering Dice-scorer på henholdsvis 0,714 og 0,731, noe som kan måle seg med de høye resultatene i HECKTOR, Gou et al. [16] og Lin et al. [12].

Å finne den mest optimale kombinasjonen av parameternivå for en dyplæringsmodell kan være en tidkrevende oppgave. Isensee et al. [72] har i *nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation* utviklet en dyplæringsbasert segmenteringsmetode som konfigureres automatisk. Dette innebærer automatisk konfigurering av preprosessering, nettverksarkitektur, trening og post-prosessering, slik at segmenteringsmodellen kan tilpasse seg enhver biomedisinsk segmenteringsoppgave. Allsidigheten til nnU-Net ble undersøkt ved å bruke modellen på 23 ulike datasett fra 11 internasjonale segmenteringsutfordringer, med til sammen 53 segmenteringsoppgaver.

nnU-Net oppnår høyeste ytelse i 33 av 53 segmenteringsoppgaver, og på lik linje eller i nærheten av nivået for de høyest presterende modellene på de resterende oppgavene [72].

6.6 Kunstig intelligens innen radiologi

En modell for automatisk inntegning av kreftsvulster vil potensielt være tidsbesparende. En ideell modell vil ha muligheten til å ta inn hvilket som helst bilde og foreta en inntegning med en nøyaktighet som er like god som en inntegning gjort av en radiolog. For inntegning av kreftsvulster i hode- og halsregionen er det viktig at modellen har nøyaktige inntegninger, da anatomien i hode- og halsregionen er kompleks og konsekvensene av en feilaktig inntegning vil kunne føre til senskader og skader på nærliggende risikoorganer [3]. Det vil på grunn av dette være viktig med en lav inter- og intravariabilitet, og ifølge arbeidet til Lin et al. [12] kan en automatisk inntegningsmodell brukt som et hjelpemiddel for radiologene redusere inter- og intravariabiliteten med henholdsvis 55 og 36 prosent.

6.6.1 Tidsbruk

Å bruke et konvolusjonsnettverk for å klassifisere bilder er en oppgave som krever en stor mengde prosesseringskraft. Bildene i denne oppgaven består av 191 x 256 piksler som skal analyseres og evalueres. Dette gjør inntegningen til en tidkrevende oppgave, som er avhengig av tilgjengelig prosesseringskraft. Lin et al. [12] fant i sitt arbeid ut at å bruke en modell for automatisk inntegning som et hjelpemiddel for radiologene, reduserte tidsforbruket med 39 prosent. For de ulike modellene i denne masteroppgaven varierte tiden som gikk med på å trene og validere fra 4 til 32 timer. Tiden brukt til inntegningene i bildene for alle 40 pasientene i testsettet varierte mellom 6-7 minutter. For Maastro-testsettet gikk omkring 43 minutter med til inntegning i bildene til alle 114 pasienter.

6.6.2 Utfordringer knyttet til bruken av kunstig intelligens

En utfordring knyttet til bruken av dyplæringsalgoritmer er risikoen for overtilpassing av modellen. På generell basis vil modeller som er overtilpasset, tilpasse seg og gjøre det bra på treningsdataen, mens på usett data vil den prestere dårlig [36]. Som forklart tidligere kan det ut ifra Figur 5.9 fra resultatet tyde på at modellene brukt i denne masteroppgaven er noe overtilpasset.

Modellen krever manuelle inntegninger som brukes som sann inntegning i trenings- og valideringsprosessen, noe som gjør modellen bias ovenfor radiologen som gjorde inntegningene. En automatisk segmenteringsmodell kan kun bli like nøyaktig som det de

sanne inntegningene er. Dersom en modell med inntegninger fra en radiolog får en høy nøyaktighet på et testsett som har inntegninger fra samme radiolog, vil det være rimelig å tenke at modellen ikke ville prestert like bra på et testsett med inntegninger fra en annen radiolog. Man bør derfor vurdere om en modell som presterer godt på inntegninger fra flere radiologer er en bedre metode enn en modell som presterer veldig bra, men som er bias ovenfor en radiolog.

Et annet problem som må tas i betraktning når det kommer til bruk av kunstig intelligens innenfor helsesektoren er hvorvidt modellene er troverdige. I de fleste tilfeller er det vanskelig å forklare modellens grunnlag for inntegningen som den har gitt. Juridiske aspekter ved bruken av medisinske bilder og hvem som har ansvaret for de predikerte inntegningen bør også diskuteres. Konsekvensene av en feilaktig inntegning er store, og det er derfor viktig å ha klart hvem ansvaret ligger hos. I følge «Etiske retningslinjer for troverdig kunstig intelligens» [73] utarbeidet av EU-kommisjonen, kreves et kontrollsystem for modeller som skal brukes for medisinsk data, for eksempel menneskelig validering. For automatisk inntegning av kreftsvulster kan dette være en radiolog som bruker den automatiske inntegningen som et hjelpemiddel for å foreta en inntegning, og har ansvaret for den endelige inntegningen.

Personvern og sikkerhet rundt medisinske data er også et tema som bør diskuteres. Helseregisterloven [74] regulerer behandlingen av helseopplysninger som ikke er behandlingsrettet, men som skal brukes til for eksempel forskning. Loven fastslår at behandlingen av dataen må skje på en etisk forsvarlig måte, som fremmer helse og ivaretar den enkeltes personvern. Bruken av kunstig intelligens på sensitiv medisinsk data må derfor bare skje dersom de strenge kravene for blant annet taushetsplikt og informasjonssikkerhet er oppfylt.

6.7 Videre arbeid

6.7.1 Interaksjoner mellom parametere

Som forklart tidligere var en begrensning av den statistiske analysen at det ikke ble sett på interaksjoner mellom de ulike parameterne. Med bakgrunn i teorien er det en sammenheng mellom flere av parameterne, og interaksjoner mellom disse bør i videre arbeid undersøkes. Interaksjoner mellom de ulike parameterne vil potensielt kunne påvirke valget av parameternivå for de ulike parameterne.

Det ble som forklart i delkapittel 4.4 foretatt ulike transformasjoner av dataen for å kunne oppfylle kravet til en N-veis ANOVA om normalfordelte residualer. Her ble det undersøkt et utvalg av transformasjoner, men ingen av transformasjonene klarte å gjøre residualene mer normalfordelt. Videre arbeid kan være å undersøke om ytterligere transformasjoner av dataen kan gjøre residualene til dataen normalfordelt slik at en N-veis ANOVA kan brukes til for å undersøke interaksjoner mellom parameterne.

6.7.2 Videre justering av parametere

Andre endringer i parametere eller parameternivå er en modifikasjon av modellen som kan gi en enda høyere ytelse. Et begrenset antall av parameternivåer ble undersøkt i denne masteroppgaven, så videre justering av parametere er noe som bør undersøkes. Videre justering av parametere kan innebære å prøve ulike dropoutrater og læringsrater, en dypere modell ved å øke antall nedsamplingslag eller en mer kompleks modell ved å øke antallet residualblokker etter hvert nedsamplingslag. Det har både i resultatene i denne masteroppgaven og i [59] vist seg at bildeaugmentering har en positiv effekt på ytelsen. I videre arbeid bør derfor flere bildeaugmenteringsteknikker undersøkes. Det vil også være interessant å se hvordan VoxResNet-modellen hadde prestert ved å bruke en metode for automatisk konfigurering slik som vist i [72].

6.7.3 Kjøring av modellen

Når et eksperiment blir kjørt i Orion, hvor en modell blir trent, vil vektene få en bestemt verdi. Vektene bestemmer hvordan dataen blir prosessert gjennom nettverket og vil dermed være med på å påvirke ytelsen. Disse vektene tas med videre når modellen skal vurderes på validering- og testsettet. Dersom samme modell kjøres flere ganger, vil ikke disse vektene bli like hver gang. I denne masteroppgaven ble hver modell kjørt en gang, som kan gi noe tilfeldige resultater. Videre arbeid kan derfor være å kjøre de ulike modellene flere ganger, og dermed ta gjennomsnittet av ytelsen for å få et mer nøyaktig estimat.

6.7.4 Kryssvalidering

Valideringssettet som brukes i denne masteroppgaven er på kun 15 pasienter. Ytelsen til modellen vil kunne bli påvirket av hvilke pasienter som plasseres i valideringssettet. Som sett oppnår modellen i Groendahl et al. [18], trent med kryssvalidering, en høy ytelse. Kryssvalidering er en metode for evaluering av nettverket som kan være en fordel å bruke i tilfeller der valideringssettet er lite. Kryssvalidering innebærer å dele den tilgjengelige dataen

i *k* antall folder, for så å velge en av foldene til validering og resten til treningssett. Dette gjentas *k* antall ganger, helt til alle foldene har vært valideringssett. Valideringsscoren beregnes deretter som gjennomsnittet av scorene fra de ulike foldene [9]. Valideringssettet fra OUS-datasettet er relativ lite, så til videre arbeid bør det undersøkes om bruken av kryssvaliderings vil gjøre ytelsen bedre.

Kapittel 7: Konklusjon

I denne masteroppgaven ble et konvolusjonsnettverk undersøkt for bruk til automatisk inntegning av kreftsvulster og affiserte lymfeknuter i PET/CT-bilder hos pasienter med hode- og halskreft. Nettverket som ble undersøkt var en 2D VoxResNet-arkitektur, basert på arkitekturen fra Lin et al. [12]. Videre ble fire av parameterne i nettverket undersøkt, for å finne kombinasjonen av de parameterne som ga den største graden av overlapp med korresponderende sann inntegning gjort av radiologer. For å finne den beste kombinasjonen av parameternivå, ble statistisk testing utført. Deretter ble det undersøkt hvordan bildeaugmentering påvirket ytelsen til nettverket samt hvordan modellen presterte på det eksterne testsettet fra Maastro Clinic, Nederland.

Eksperimentene viste et generelt godt potensiale for automatisk inntegning i PET/CT-bilder i hode- og halsregionen. Den statistiske testingen viste at det var signifikante forskjeller for valg av parameternivå for de ulike parameterne. Kombinasjonen av parameternivåene som ga høyest Dice-score, ga en beste modell med læringsrate på 10^{-4}, 48 filtre, batchnormaliseringslag og ingen dropoutrate. Denne modellen oppnådde en gjennomsnittlig Dice-score per pasient på 0,714 ± 0,16 på testsettet. Ved å inkludere bildeaugmentering i den beste modellen økte ytelsen. Modellen med bildeaugmentering fikk en gjennomsnittlig Dice-score per pasient på 0,731 ± 0,16 på testsettet. Den beste modellen kan måle seg med tidligere arbeid gjort på tilsvarende segmenteringsproblem, slik som i Moe et al. [17]. Testingen på det eksterne Maastro-testsettet resulterte i forventede resultater for modellene uten og med bildeaugmentering med gjennomsnittlig Dice-score per pasient på henholdsvis 0,629 ± 0,19 og 0,635 ± 0,18.

Til tross for gode resultater ser det imidlertid ut som at modellen presterer dårlig på bildetversnitt der tumorstørrelsen er liten eller der bildene er atypiske sammenliknet med majoriteten av bildene. Dette gjør at modellen bør ha en form for menneskelig validering av inntegningen før den brukes, eller at den brukes som et inntegningshjelpemiddel for radiologene og onkologene. På den måten kan den bidra til å redusere tiden som brukes på inntegningene. For å konkludere, gir modellen for automatisk inntegning av kreftsvulster og lymfeknuter for pasienter med hode- og halskreft med en 2D VoxResNet-arkitektur lovende resultater og bør derfor undersøkes videre. For videre utvikling og forskning bør blant annet interaksjoner mellom parameterne og flere justeringer av parameternivå undersøkes, samt at flere bildeaugmenteringsteknikker bør utforskes.

Kapittel 8: Referanser

[1] Kreftforeningen. "Hva er kreft?" Hentet fra: https://kreftforeningen.no/om-kreft/hva-er-kreft/ (lest 08.03.2021).

[2] World Health Organization. "All cancers fact sheet." Hentet fra: https://gco.iarc.fr/today/data/factsheets/cancers/39-All-cancers-fact-sheet.pdf (lest 08.05.2021).

[3] Kreftforeningen. "Hode- og halskreft." Hentet fra: https://kreftforeningen.no/om-kreft/kreftformer/hode-og-halskreft/ (lest 08.03.2021).

[4] Helsedirektoratet, *Nasjonalt handlingsprogram med retningslinjer for diagnostikk, behandling og oppfølging av hode-/haslskreft* Oslo: Helsedirektoratet 2020.

[5] Kreftforeningen. "PET/CT." Hentet fra: https://kreftforeningen.no/om-kreft/undersokelser/pet-ct/ (lest 08.02.2021).

[6] L. V. M. Loureiro *et al.*, "Waiting time to radiotherapy as a prognostic factor for glioblastoma patients in a scenario of medical disparities," *Arquivos de neuro-psiquiatria*, Online vol. 73, no. 2, s. 104-110, feb. 2015, doi: 10.1590/0004-282X20140202.

[7] E. Weiss and C. F. Hess, "The impact of gross tumor volume (GTV) and clinical target volume (CTV) definition on the total accuracy in radiotherapy," *Strahlentherapie und Onkologie*, Online vol. 179, no. 1, s. 21-30, jan. 2003, doi: 10.1007/s00066-003-0976-5.

[8] C. F. Njeh, "Tumor delineation: The weakest link in the search for accuracy in radiotherapy," *Journal of medical physics*, Online vol. 33, no. 4, s. 136-140, 2008, doi: 10.4103/0971-6203.44472.

[9] F. Chollet, *Deep learning with Python*, 1. utg. New York: Manning Publications Co., 2018.

[10] D. Ravì *et al.*, "Deep learning for health informatics," *IEEE journal of biomedical and health informatics*, Online vol. 21, no. 1, s. 4-21, jan. 2017, doi: 10.1109/JBHI.2016.2636665.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," i *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Munich, Germany, nov. 2015: Springer, s. 234-241, doi: 10.1007/978-3-319-24574-4_28.

[12] L. Lin *et al.*, "Deep learning for automated contouring of primary tumor volumes by MRI for nasopharyngeal carcinoma," *Radiology*, Online vol. 291, no. 3, s. 677-686, juni 2019, doi: 10.1148/radiol.2019182012.

[13] J. Bresnick. "Arguing the Pros and Cons of Artificial Intelligence in Healthcare." Hentet fra: https://healthitanalytics.com/news/arguing-the-pros-and-cons-of-artificial-intelligence-in-healthcare (lest 08.04.2021).

[14] V. Andrearczyk *et al.*, "Overview of the HECKTOR challenge at MICCAI 2020: automatic head and neck tumor segmentation in PET/CT," i *3D Head and Neck Tumor Segmentation in PET/CT Challenge*, Cham, jan. 2020: Springer, s. 1-21, doi: 10.1007/978-3-030-67194-5 1.

[15] S. van den Bosch *et al.*, "18F-FDG-PET/CT-based treatment planning for definitive (chemo) radiotherapy in patients with head and neck squamous cell carcinoma improves regional control and survival," *Radiotherapy and Oncology*, vol. 142, s. 107-114, 2020, doi: 10.1016/j.radonc.2019.07.025.

[16] Z. Guo, N. Guo, K. Gong, and Q. Li, "Gross tumor volume segmentation for head and neck cancer radiotherapy using deep dense multi-modality network," *Physics in*

Medicine & Biology, Online vol. 64, no. 20, okt. 2019, doi: 10.1088/1361-6560/ab440d.
[17] Y. M. Moe, A. R. Groendahl, O. Tomic, E. Dale, E. Malinen, and C. M. Futsaether, "Deep learning-based auto-delineation of gross tumour volumes and involved nodes in PET/CT images of head and neck cancer patients," *European journal of nuclear medicine and molecular imaging*, Online s. 1-11, feb. 2021, doi: 10.1007/s00259-020-05125-x.
[18] A. R. Groendahl *et al.*, "A comparison of fully automatic segmentation of tumors and involved nodes in PET/CT of head and neck cancers," *Physics in Medicine & Biology*, vol. 66, no. 6, mars 2021, doi: 10.1088/1361-6560/abe553.
[19] J. M. Moan, C. D. Amdal, E. Malinen, J. G. Svestad, T. V. Bogsrud, and E. Dale, "The prognostic role of 18F-fluorodeoxyglucose PET in head and neck cancer depends on HPV status," *Radiotherapy and Oncology*, vol. 140, s. 54-61, nov. 2019, doi: 10.1016/j.radonc.2019.05.019.
[20] J. K. Shultis and R. E. Faw, *Fundementals of Nuclear Science and Engineering*, , 3. utg. Boca Raton: CRC Press, 2016.
[21] J. G. Webster, *Medical Instrumentation: Application and Design*, 4. utg. New York: John Wiley & Sons, INC., 2009.
[22] S. A. Kane and B. A. Gelman, *Introduction to Physics in Modern Medicine*, 3. utg. Boca Raton: CRC Press, 2020.
[23] F. Gaillard and D. A. Goel. "Photoelectic effect " Hentet fra: https://radiopaedia.org/articles/photoelectric-effect (lest 02.02.2021).
[24] A. P. F. Gaillard and D. A. Goel. "Compton effect " Hentet fra: https://radiopaedia.org/articles/compton-effect?lang=us (lest 02.02.2021).
[25] A. P. F. Gaillard and D. A. Goel. "Pair production." Hentet fra: https://radiopaedia.org/articles/pair-production?lang=us (lest 03.02.2021).
[26] J. Lilley, *Nuclear physics - Principles and Applications*, 1. utg. Chichester: John Wiley & Sons Ltd., 2001.
[27] M. J. McKenzie and P. S. Goergen. "Computed Tomography (CT)." Hentet fra: https://www.insideradiology.com.au/computed-tomography/ (lest 29.01.2021).
[28] Direktoratet for strålevern og atomsikkerhet. "Ord og uttrykk for strålefysikk i stråleterapi." Hentet fra: https://dsa.no/laboratoriene/ord-og-uttrykk-for-stralefysikk-i-straleterapi (lest 04.02.2021).
[29] D. Bell and K. Greenway. "Hounsfield unit " Hentet fra: https://radiopaedia.org/articles/hounsfield-unit (lest 04.02.2021).
[30] A. Murphy. "Windowing (CT)." Hentet fra: https://radiopaedia.org/articles/windowing-ct (lest 04.02.2021).
[31] R. L. Wahl, *Principles and Practice of PET and PET/CT*, 2. utg. Philadelphia: Wolters Kluwer, 2009.
[32] M. E. Phelps, *PET Physics, instrumentation, and scanners*, 1. utg. Los Angeles: Springer, 2006.
[33] D. Bell and F. Deng. "F-18 fluorodeoxyglucose." Hentet fra: https://radiopaedia.org/articles/f-18-fluorodeoxyglucose/revisions?lang=us (lest 08.02.2021).
[34] I. Bickle and A. Shetty. "Positron emission tomography." Hentet fra: https://radiopaedia.org/articles/positron-emission-tomography?lang=us (lest 08.02.2021).
[35] P. E. Kinahan and J. W. Fletcher, "Positron emission tomography-computed tomography standardized uptake values in clinical practice and assessing response to

therapy," *Seminars in Ultrasound, CT and MRI*, Online vol. 31, no. 6, s. 496-505, des. 2010, doi: 10.1053/j.sult.2010.10.001.
[36] S. Raschka and V. Mirjalili, *Python Machine Learning*, 3. utg. Birmingham: Packt Publishing Ltd., 2019.
[37] V. Zocca, G. Spacagna, D. Slater, and P. Roelants, *Python Deep Learning*, 1. utg. Birmingham: Packt Publishing Ltd., 2017.
[38] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, 2. utg. New York: Springer, 2013.
[39] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," i *2016 fourth international conference on 3D vision (3DV)*, Stanford, California, juni 2016: IEEE, s. 565-571, doi: 10.1109/3DV.2016.79.
[40] Y. M. Moe, "Deep learning for automatic delineation of tumours from PET/CT images," Masteroppgave, Faculty of Science and Technology, Norwegian University of Life Science (NMBU), Ås, 2019.
[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2015. [Online]. Hentet fra: arXiv:1412.6980v9 [cs.LG].
[42] S. Ruder, "An overview of gradient descent optimization algorithms," 2017. [Online]. Hentet fra: arXiv:1609.04747v2 [cs.LG].
[43] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015. [Online]. Hentet fra: arXiv:1411.4038v2[cs.CV].
[44] H. Noh, S. Hong, and B. Han, "Learning Deconvolution Network for Semantic Segmentation," i *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015: IEEE Computer Society, s. 1520–1528, doi: 10.1109/iccv.2015.178.
[45] Q. Ren and R. Hu, "Multi-scale deep encoder-decoder network for salient object detection," *Neurocomputing*, Online vol. 316, d. 95-104, nov. 2018, doi: 10.1016/j.neucom.2018.07.055.
[46] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, "The importance of skip connections in biomedical image segmentation," i *Deep Learning and Data Labeling for Medical Applications*, Cham, 2016: Springer, s. 179-187, doi: 10.1007/978-3-319-46976-8_19.
[47] H. Chen, Q. Dou, L. Yu, J. Qin, and P.-A. Heng, "VoxResNet: Deep voxelwise residual networks for brain segmentation from 3D MR images," *NeuroImage*, Online vol. 170, s. 446-455, april 2018, doi: 10.1016/j.neuroimage.2017.04.041.
[48] A. A. Taha and A. Hanbury, "Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool," *BMC Medical Imaging*, Online vol. 15, no. 1, s. 1-28, aug. 2015, doi: 10.1186/s12880-015-0068-x.
[49] D. Zhang and G. Lu, "Review of shape representation and description techniques," *Pattern recognition*, Online vol. 37, no. 1, s. 1-19, 2004, doi: 10.1016/j.patcog.2003.07.008.
[50] B. N. Huynh, "Personlig kommunikasjon," 2021.
[51] UICC global cancer control. "TNM - What is the TNM cancer staging system?" Hentet fra: https://www.uicc.org/resources/tnm (lest 05.05.2021).
[52] A. R. Grøndahl, "Personlig kommunikasjon," 2021.
[53] A. Collette, *Python and HDF5: Unlocking Scientific Data*, 1. utg. Sebastopol: O'Reilly Media, Inc., 2013.
[54] B. N. Huynh, "Visualization of deep learning in auto-delineation of cancer tumors," Masteroppgave, Faculty of Science and Technology, Norwegian University of Life Science (NMBU), Ås, 2020.

[55] M. Abadi *et al.*, "Tensorflow: A System for Large-Scale Machine Learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, Savannah, USA, nov. 2016, s. 265-283.
[56] CIGENE. "CIGENE computational unit " Hentet fra: https://cigene.no/lab-and-infrastructure/cigene-computational-unit/ (lest 04.03.2021).
[57] The NMBU Orion team. "NMBU Orion user support." Hentet fra: https://nmbu-orion-support.readthedocs.io/en/latest/index.html (lest 04.03.2021).
[58] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," 2015. [Online]. Hentet fra: arXiv:1502.03167v3[cs.LG].
[59] M. Ødegaard, "Effekt av bildeaugmentering på dyp læring-basert segmentering av hode- og halskreft i PET/CT-bilder," Masteroppgave, Fakultet for realfag og tekonologi, Norges miljø- og biovitenskapelige universitet (NMBU), Ås, 2021.
[60] W. Burger and M. J. Burge, *Digital Image Processing: An Algorithmic Introduction Using Java*, 2. utg. London: Springer, 2016.
[61] D. C. Montgomery, *Design and Analysis of Experiments*, 9. utg. John wiley & sons, 2017.
[62] N. M. Razali and Y. B. Wah, "Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests," *Journal of Statistical Modeling and Analytics*, Online vol. 2, no. 1, s. 21-33, 2011.
[63] A. Grafen and R. Hails, *Modern Statistics for the Life Sciences*, 1. utg. Oxford Univesity Press, 2002.
[64] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, s. 675-701, des. 1937, doi: 10.2307/2279372.
[65] M. Hollander, D. A. Wolfe, and E. Chicken, *Nonparametric statistical methods*, 3. utg. Hoboken: John Wiley & Sons, 2013.
[66] A. P. Zijdenbos, B. M. Dawant, R. A. Margolin, and A. C. Palmer, "Morphometric analysis of white matter lesions in MR images: method and validation," *IEEE Transactions on Medical Imaging*, Online vol. 13, no. 4, s. 716-724, 1994, doi: 10.1109/42.363096.
[67] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, 1. utg. Cambridge: MIT Press 2016.
[68] A. Kleppe, O.-J. Skrede, S. De Raedt, K. Liestøl, D. J. Kerr, and H. E. Danielsen, "Designing deep learning studies in cancer diagnostics," *Nature Reviews Cancer*, Online vol. 21, no. 3, s. 199-211, mars 2021, doi: 10.1038/s41568-020-00327-9.
[69] S. Zhu, Z. Dai, and N. Wen, "Two-stage approach for segmenting gross tumor volume in head and neck cancer with CT and PET imaging," i *3D Head and Neck Tumor Segmentation in PET/CT Challenge*, 2020: Springer, s. 22-27, doi: 10.1007/978-3-030-67194-5_2.
[70] Y. Yuan, "Automatic head and neck tumor segmentation in PET/CT with scale attention network," i *3D Head and Neck Tumor Segmentation in PET/CT Challenge*, 2020: Springer, s. 44-52, doi: 10.1007/978-3-030-67194-5_5.
[71] A. Iantsen, D. Visvikis, and M. Hatt, "Squeeze-and-Excitation Normalization for Automated Delineation of Head and Neck Primary Tumors in Combined PET and CT images," i *3D Head and Neck Tumor Segmentation in PET/CT Challenge*, 2020: Springer, s. 37-43, doi: 10.1007/978-3-030-67194-5_4.
[72] F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, and K. H. Maier-Hein, "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation,"

Nature Methods, Online vol. 18, no. 2, s. 203-211, feb. 2021, doi: 10.1038/s41592-020-01008-z.
[73] Independent High-Level Expert Group on Artificial Intelligence, "Ethics guidelines for trustworthy AI," European Commission, Guidelines april 2019. [Online]. Hentet fra: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai
[74] *Lov om helseregistre og behandling av helseopplysninger* 2014.

Vedlegg A Modellarkitektur

Dette vedlegget inneholder en detaljert oversikt over sammenlikningsmodellen U-Net, brukt i Moe et al. [17], samt VoxResNet-modellen som er trent, validert og testet i denne masteroppgaven.

Detaljert oversikt over U-Net-arkitekturen

Tabell A.1: *Oversikt over sammenlikningsmodellen U-Net. Her vises nedsamplingsdelen. Hentet fra [59].*

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(191, 265, 2)	0	
conv2d (Conv2D)	(191, 265, 64)	1216	input_1
batch_normalization (BatchNormalization)	(191, 265, 64)	256	conv2d
conv2d_1 (Conv2D)	(191, 265, 64)	36928	batch_normalization
batch_normalization_1 (BatchNormalization)	(191, 265, 64)	256	conv2d_1
max_pooling2d (MaxPooling2D)	(95, 132, 64)	0	batch_normalization_1
conv2d_2 (Conv2D)	(95, 132, 128)	73856	max_pooling2d
batch_normalization_2 (BatchNormalization)	(95, 132, 128)	512	conv2d_2
conv2d_3 (Conv2D)	(95, 132, 128)	147584	batch_normalization_2
batch_normalization_3 (BatchNormalization)	(95, 132, 128)	512	conv2d_3
max_pooling2d_1 (MaxPooling2D)	(47, 66, 128)	0	batch_normalization_3
conv2d_4 (Conv2D)	(47, 66, 256)	295168	max_pooling2d_1
batch_normalization_4 (BatchNormalization)	(47, 66, 256)	1024	conv2d_4
conv2d_5 (Conv2D)	(47, 66, 256)	590080	batch_normalization_4
batch_normalization_5 (BatchNormalization)	(47, 66, 256)	1024	conv2d_5
max_pooling2d_2 (MaxPooling2D)	(23, 33, 256)	0	batch_normalization_5
conv2d_6 (Conv2D)	(23, 33, 512)	1180160	max_pooling2d_2
batch_normalization_6 (BatchNormalization)	(23, 33, 512)	2048	conv2d_6
conv2d_7 (Conv2D)	(23, 33, 512)	2359808	batch_normalization_6
batch_normalization_7 (BatchNormalization)	(23, 33, 512)	2048	conv2d_7
max_pooling2d_3 (MaxPooling2D)	(11, 16, 512)	0	batch_normalization_7

Tabell A.2: *Fortsettelse for Tabell A.1. Oversikt over sammenlikningsmodellen U-Net. Her vises flaskehalsen. Hentet fra [59].*

conv2d_8 (Conv2D)	(11, 16, 1024)	4719616	max_pooling2d_3
batch_normalization_8 (BatchNormalization)	(11, 16, 1024)	4096	conv2d_8
conv2d_9 (Conv2D)	(11, 16, 1024)	9438208	batch_normalization_8
batch_normalization_9 (BatchNormalization)	(11, 16, 1024)	4096	conv2d_9

Tabell A.3*: Fortsettelse fra Tabell A.2. Oversikt over sammenlikningsmodellen U-Net. Her vises oppsamplingsdelen. Hentet fra [59].*

conv2d_transpose (Conv2DTranspose)	(11, 16, 512)	4719104	batch_normalization_9
tf.image.resize (TFOpLambda)	(23, 33, 512)	0	conv2d_transpose
concatenate (Concatenate)	(23, 33, 1024)	0	batch_normalization_7 tf.image.resize
conv2d_10 (Conv2D)	(23, 33, 512)	4719104	concatenate
batch_normalization_10 (BatchNormalization)	(23, 33, 512)	2048	conv2d_10
conv2d_11 (Conv2D)	(23, 33, 512)	2359808	batch_normalization_10
batch_normalization_11 (BatchNormalization)	(23, 33, 512)	2048	conv2d_11
conv2d_transpose_1 (Conv2DTranspose)	(23, 33, 256)	1179904	batch_normalization_11
tf.image.resize_1 (TFOpLambda)	(47, 66, 256)	0	conv2d_transpose_1
concatenate_1 (Concatenate)	(47, 66, 512)	0	batch_normalization_5 tf.image.resize_1
conv2d_12 (Conv2D)	(47, 66, 256)	1179904	concatenate_1
batch_normalization_12 (BatchNormalization)	(47, 66, 256)	1024	conv2d_12
conv2d_13 (Conv2D)	(47, 66, 256)	590080	batch_normalization_12
batch_normalization_13 (BatchNormalization)	(47, 66, 256)	1024	conv2d_13
conv2d_transpose_2 (Conv2DTranspose)	(47, 66, 128)	295040	batch_normalization_13
tf.image.resize_2 (TFOpLambda)	(95, 132, 128)	0	conv2d_transpose_2
concatenate_2 (Concatenate)	(95, 132, 256)	0	batch_normalization_3 tf.image.resize_2

Tabell A.4: *Fortsettelse fra Tabell A.3. Hentet fra [59].*

conv2d_14 (Conv2D)	(95, 132, 128)	295040	concatenate_2
batch_normalization_14 (BatchNormalization)	(95, 132, 128)	512	conv2d_14
conv2d_15 (Conv2D)	(95, 132, 128)	147584	batch_normalization_14
batch_normalization_15 (BatchNormalization)	(95, 132, 128)	512	conv2d_15
conv2d_transpose_3 (Conv2DTranspose)	(95, 132, 64)	73792	batch_normalization_15
tf.image.resize_3 (TFOpLambda)	(191, 265, 64)	0	conv2d_transpose_3
concatenate_3 (Concatenate)	(191, 265, 128)	0	batch_normalization_1 tf.image.resize_3
conv2d_16 (Conv2D)	(191, 265, 64)	73792	concatenate_3
batch_normalization_16 (BatchNormalization)	(191, 265, 64)	256	conv2d_16
conv2d_17 (Conv2D)	(191, 265, 64)	36928	batch_normalization_16
batch_normalization_17 (BatchNormalization)	(191, 265, 64)	256	conv2d_17
conv2d_18 (Conv2D)	(191, 265, 1)	577	batch_normalization_17

Total params: 34,536,833
Trainable params: 34,525,057
Non-trainable params: 11,776

Detaljert oversikt over VoxResNet-arkitekturen

Tabell A.5: Detaljert oversikt over VoxResNet-arkitekturen.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(191, 265, 2)	0	
conv2d (Conv2D)	(191, 265, 48)	912	input_1
batch_normalization (BatchNormalization)	(191, 265, 48)	192	conv2d
conv2d_1 (Conv2D)	(191, 265, 48)	20784	batch_normalization
batch_normalization_1 (BatchNormalization)	(191, 265, 48)	192	conv2d_1
max_pooling2d (MaxPooling2D)	(95, 132, 48)	0	batch_normalization_1
batch_normalization_2 (BatchNormalization)	(95, 132, 48)	192	max_pooling2d
activation (Activation)	(95, 132, 48)	0	batch_normalization_2
conv2d_2 (Conv2D)	(95, 132, 48)	20784	activation
batch_normalization_3 (BatchNormalization)	(95, 132, 48)	192	conv2d_2
conv2d_3 (Conv2D)	(95, 132, 48)	20784	batch_normalization_3
add (Add)	(95, 132, 48)	0	max_pooling2d conv2d_3
batch_normalization_4 (BatchNormalization)	(95, 132, 48)	192	add
activation_1 (Activation)	(95, 132, 48)	0	batch_normalization_4
conv2d_4 (Conv2D)	(95, 132, 48)	20784	activation_1
batch_normalization_5 (BatchNormalization)	(95, 132, 48)	192	conv2d_4
conv2d_5 (Conv2D)	(95, 132, 48)	20784	batch_normalization_5
add_1 (Add)	(95, 132, 48)	0	add conv2d_5

***Tabell A.6**: Fortsettelse av Tabell A.5.*

max_pooling2d_1 (MaxPooling2D)	(47, 66, 48)	0	add_1
batch_normalization_6 (BatchNormalization)	(47, 66, 48)	192	max_pooling2d_1
activation_2 (Activation)	(47, 66, 48)	0	batch_normalization_6
conv2d_6 (Conv2D)	(47, 66, 48)	20784	activation_2
batch_normalization_7 (BatchNormalization)	(47, 66, 48)	192	conv2d_6
conv2d_7 (Conv2D)	(47, 66, 48)	20784	batch_normalization_7
add_2 (Add)	(47, 66, 48)	0	max_pooling2d_1 conv2d_7
batch_normalization_8 (BatchNormalization)	(47, 66, 48)	192	add_2
activation_3 (Activation)	(47, 66, 48)	0	batch_normalization_8
conv2d_8 (Conv2D)	(47, 66, 48)	20784	activation_3
batch_normalization_9 (BatchNormalization)	(47, 66, 48)	192	conv2d_8
conv2d_9 (Conv2D)	(47, 66, 48)	20784	batch_normalization_9
add_3 (Add)	(47, 66, 48)	0	add_2 conv2d_9
max_pooling2d_2 (MaxPooling2D)	(23, 33, 48)	0	add_3
batch_normalization_10 (BatchNormalization)	(23, 33, 48)	192	max_pooling2d_2
activation_4 (Activation)	(23, 33, 48)	0	batch_normalization_10
conv2d_10 (Conv2D)	(23, 33, 48)	20784	activation_4
batch_normalization_11 (BatchNormalization)	(23, 33, 48)	192	conv2d_10

***Tabell A.7**: Fortsettelse av Tabell A.6.*

conv2d_11 (Conv2D)	(23, 33, 48)	20784	batch_normalization_11
add_4 (Add)	(23, 33, 48)	0	max_pooling2d_2 conv2d_11
batch_normalization_12 (BatchNormalization)	(23, 33, 48)	192	add_4
activation_5 (Activation)	(23, 33, 48)	0	batch_normalization_12
conv2d_12 (Conv2D)	(23, 33, 48)	20784	activation_5
batch_normalization_13 (BatchNormalization)	(23, 33, 48)	192	conv2d_12
conv2d_13 (Conv2D)	(23, 33, 48)	20784	batch_normalization_13
add_5	(23, 33, 48)	0	add_4 conv2d_13
conv2d_transpose (Conv2DTranspose)	(46, 66, 48)	20784	add_5
tf.image.resize (TFOpLambda)	(47, 66, 48)	0	conv2d_transpose
concatenate (Concatenate)	(47, 66, 96)	0	max_pooling2d_1 tf.image.resize
conv2d_14 (Conv2D)	(47, 66, 48)	41520	concatenate
batch_normalization_14 (BatchNormalization)	(47, 66, 48)	192	conv2d_14
conv2d_15 (Conv2D)	(47, 66, 48)	20784	batch_normalization_14
batch_normalization_15 (BatchNormalization)	(47, 66, 48)	192	conv2d_15
conv2d_transpose_1 (Conv2DTranspose)	(94, 132, 48)	20784	batch_normalization_15
tf.image.resize_1 (TFOpLambda)	(95, 132, 48)	0	conv2d_transpose_1
concatenate_1 (Concatenate)	(95, 132, 96)	0	max_pooling2d tf.image.resize_1

Tabell A.8: *Fortsettelse av Tabell A.7.*

conv2d_16 (Conv2D)	(95, 132, 48)	41520	concatenate_1
batch_normalization_16 (BatchNormalization)	(95, 132, 48)	192	conv2d_16
conv2d_17 (Conv2D)	(95, 132, 48)	20784	batch_normalization_16
batch_normalization_17 (BatchNormalization)	(95, 132, 48)	192	conv2d_17
conv2d_transpose_2 (Conv2DTranspose)	(190, 264, 48)	20784	batch_normalization_17
tf.image.resize_2 (TFOpLambda)	(191, 265, 48)	0	conv2d_transpose_2
conv2d_18 (Conv2D)	(191, 265, 48)	20784	tf.image.resize_2
batch_normalization_18 (BatchNormalization)	(191, 265, 48)	192	conv2d_18
conv2d_19 (Conv2D)	(191, 265, 48)	20784	batch_normalization_18
batch_normalization_19 (BatchNormalization)	(191, 265, 48)	192	conv2d_19
conv2d_20 (Conv2D)	(191, 265, 1)	433	batch_normalization_19

Total params: 503,905
Trainable params: 501,985
Non-trainable params: 1,920

Vedlegg B Eksperimentplan og konfigurasjonsfil

Vedlegg B inneholder en oversikt over de ulike eksperimentene som ble gjennomført i denne masteroppgaven, og et eksempel på en konfigurasjonsfil som ble brukt for å kjøre eksperimenter i Orion. De resterende konfigurasjonsfilene kan finnes her: https://github.com/sofiekrogstie/cnn-template/tree/master/config

Oversikt over eksperimentene

Tabell B.1: *Oversikt over de ulike eksperimentene som ble gjennomført med tilhørende gjennomsnittlig Dice-score per tverrsnitt.*

Batchnormalisering	Dropout	Læringsrate	Antall filtre	Dice-score
Yes	0	10^{-4}	48	0,6014
No	0	10^{-4}	48	0,5986
Yes	0,5	10^{-4}	48	0,6218
No	0,5	10^{-4}	48	0,6052
Yes	0	10^{-5}	48	0,5880
No	0	10^{-5}	48	0,5921
Yes	0,5	10^{-5}	48	0,5726
No	0,5	10^{-5}	48	0,5886
Yes	0	10^{-3}	48	0,6259
No	0	10^{-3}	48	0,5844
Yes	0,5	10^{-3}	48	0,6110
No	0,5	10^{-3}	48	4,48E-11
Yes	0	10^{-4}	64	0,5953
No	0	10^{-4}	64	0,5974
Yes	0,5	10^{-4}	64	0,6124
No	0,5	10^{-4}	64	0,6229
Yes	0	10^{-5}	64	0,5940
No	0	10^{-5}	64	0,5911
Yes	0,5	10^{-5}	64	0,5806
No	0,5	10^{-5}	64	0,6072
Yes	0	10^{-3}	64	0,6167
No	0	10^{-3}	64	4,48E-11
Yes	0,5	10^{-3}	64	0,6240
No	0,5	10^{-3}	64	4,48E-11
Yes	0	10^{-4}	80	0,6065
No	0	10^{-4}	80	0,6150
Yes	0,5	10^{-4}	80	0,6092
No	0,5	10^{-4}	80	0,6146
Yes	0	10^{-5}	80	0,5943
No	0	10^{-5}	80	0,6039
Yes	0,5	10^{-5}	80	0,5878
No	0,5	10^{-5}	80	0,6094
Yes	0	10^{-3}	80	0,6194
No	0	10^{-3}	80	4,48E-11
Yes	0,5	10^{-3}	80	0,6227
No	0,5	10^{-3}	80	4,48E-11

Eksempel på konfigurasjonsfil

Konfigurasjonsfilen for beste modell med batchnormalisering, 0 i dropoutrate, 10^{-4} i læringsrate og 48 filtre:

```
{
    "dataset_params": {
        "class_name": "H5Reader",
        "config": {
            "filename": "/home/work/sofiek/hn_delin/full_dataset_singleclass.h5",
            "batch_size": 16,
            "x_name": "input",
            "y_name": "target",
            "batch_cache": 10,
            "shuffle": true,
            "fold_prefix": "",
            "train_folds": [
                "train"
            ],
            "val_folds": [
                "val"
            ],
            "test_folds": [
                "test"
            ],
            "preprocessors": [
                {
                    "class_name": "HounsfieldWindowingPreprocessor",
                    "config": {
                        "window_center": 70,
                        "window_width": 200,
                        "channel": 0
                    }
                },
                {
                    "class_name": "ImageNormalizerPreprocessor",
                    "config": {
                        "vmin": [
                            -100,
                            0
                        ],
                        "vmax": [
                            100,
                            25
                        ]
                    }
                }
            ]
        }
    },
    "train_params": {
        "epochs": 200,
        "callbacks": [
            {
                "class_name": "EarlyStopping",
                "config": {
                    "monitor": "val_loss",
                    "patience": 30
                }
            }
        ]
    },
    "input_params": {
        "shape": [
            191,
            265,
            2
        ]
    },
```

```
            }
        },
        {
            "res_block": 2,
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            }
        },
        {
            "name": "pool2",
            "class_name": "MaxPooling2D"
        },
        {
            "res_block": 2,
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            }
        },
        {
            "res_block": 2,
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            }
        },
        {
            "name": "pool3",
            "class_name": "MaxPooling2D"
        },
        {
            "res_block": 2,
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            }
```

```
            },
            {
                "res_block": 2,
                "class_name": "Conv2D",
                "config": {
                    "filters": 48,
                    "kernel_size": 3,
                    "activation": "relu",
                    "kernel_initializer": "he_normal",
                    "padding": "same"
                },
                "normalizer": {
                    "class_name": "BatchNormalization"
                }
            },
            {
                "name": "upconv2",
                "class_name": "Conv2DTranspose",
                "config": {
                    "filters": 48,
                    "kernel_size": 3,
                    "strides": 2,
                    "kernel_initializer": "he_normal",
                    "padding": "same"
                }
            },
            {
                "class_name": "Conv2D",
                "config": {
                    "filters": 48,
                    "kernel_size": 3,
                    "activation": "relu",
                    "kernel_initializer": "he_normal",
                     "padding": "same"
                },
                "normalizer": {
                    "class_name": "BatchNormalization"
                },
                "inputs": [
                    "pool2",
                    "upconv2"
                ]
            },
            {
                "class_name": "Conv2D",
                "config": {
                    "filters": 48,
                    "kernel_size": 3,
                    "activation": "relu",
                    "kernel_initializer": "he_normal",
                    "padding": "same"
                },
                "normalizer": {
                    "class_name": "BatchNormalization"
                }
            },
            {
                "name": "upconv1",
                "class_name": "Conv2DTranspose",
                "config": {
                    "filters": 48,
                    "kernel_size": 3,
                    "strides": 2,
                    "kernel_initializer": "he_normal",
                    "padding": "same"
```

```
            }
        },
        {
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            },
            "inputs": [
                "pool1",
                "upconv1"
            ]
        },
        {
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            }
        },
        {
            "class_name": "Conv2DTranspose",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "strides": 2,
                "kernel_initializer": "he_normal",
                "padding": "same"
            }
        },
        {
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
            "normalizer": {
                "class_name": "BatchNormalization"
            },
            "resize_inputs": true
        },
        {
            "class_name": "Conv2D",
            "config": {
                "filters": 48,
                "kernel_size": 3,
                "activation": "relu",
                "kernel_initializer": "he_normal",
                "padding": "same"
            },
```

```
                "normalizer": {
                    "class_name": "BatchNormalization"
                }
            },
            {
                "class_name": "Conv2D",
                "config": {
                    "filters": 1,
                    "kernel_size": 3,
                    "activation": "sigmoid",
                    "kernel_initializer": "he_normal",
                    "padding": "same"
                }
            }
        ]
    }
}
```

Vedlegg C Statistisk analyse

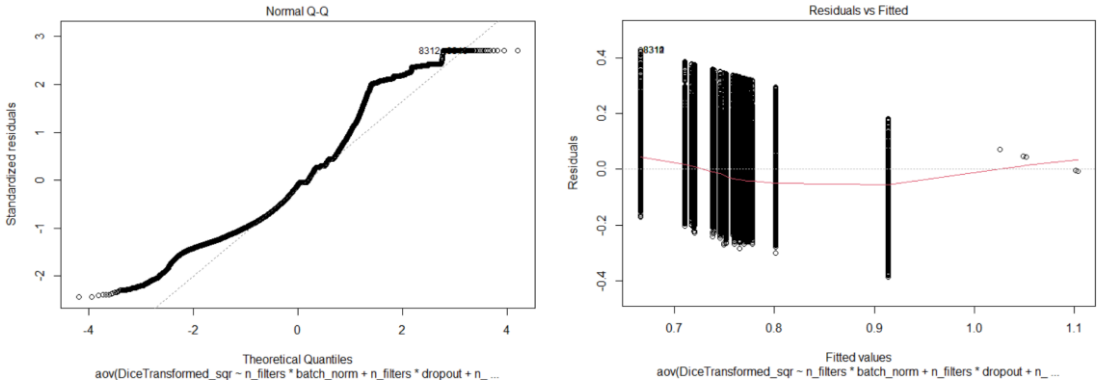
Vedlegg C inneholder en oversikt over koden brukt til den statistiske analysen gjennomført i denne masteroppgaven. Vedlagt ligger koden fra RStudio og plottene fra den statistiske analysen.

Undersøkelse av kravet om normalfordeling samt utførte transformasjoner

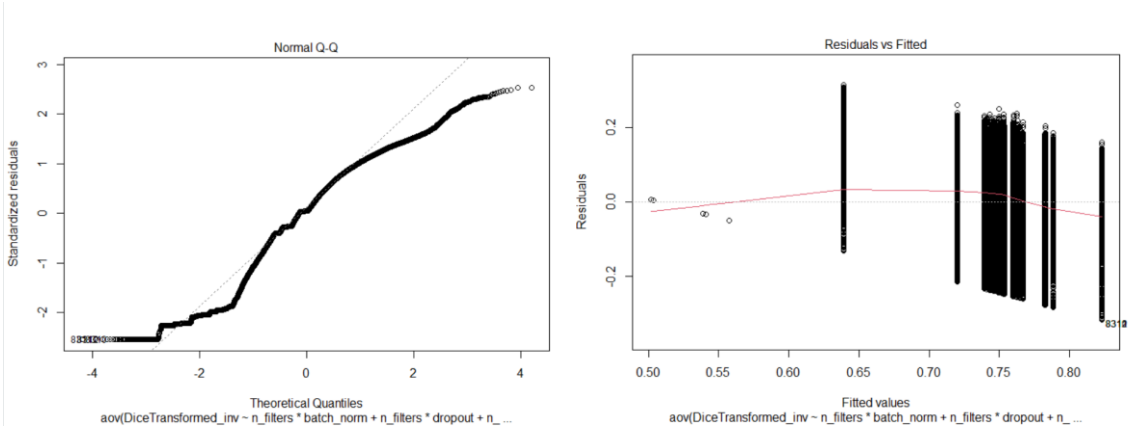
```
# Undersøker først kravet om normalfordelte residualer

# Laster inn data
data <- dice_per_slice_parametere

# Lager uavhengige faktorvariabler med nye navn for noen
data$n_filters <- factor(data$n_filters,
                         levels = c(48,64,80),
                         labels = c("F_48", "F_64", "F_80"))

data$batch_norm <- factor(data$batch_norm,
                          levels = c("yes","no"),
                          labels = c("BN_yes", "BN_no"))

data$dropout <- factor(data$dropout,
                       levels = c("0","0,5"),
                       labels = c("D_yes", "D_no"))

data$learning_rate <- factor(data$learning_rate,
                             levels = c("0,001", "0,0001", "0,00001"),
                             labels = c("LR_3", "LR_4", "LR_5"))

# ANOVA med interaksjoner
AOV.1 <- aov(f1_score ~n_filters*batch_norm + n_filters*dropout + n_filters*learning_rate + batch_norm*dropout +
               batch_norm*learning_rate + dropout*learning_rate, data = data)

summary(AOV.1)

# Diagnoseplot (for å undersøke antagelsen om normalfordeling)

# 1) Residuals vs. fitted
plot(AOV.1, 1)
# Gjennomsnittet bør være en rett linje nær null

# 2) QQ-plot
plot(AOV.1, 2)
# QQ-plottet bør følge den stipplede linjen

# Anderson-Darling normalitetsstest (for å undersøke antagelsen om normalfordeling)
library(nortest)
ad.test(AOV.1$residuals)

# Ettersom residualene ikke er normalfordelte, foretas det ulike transformasjoner av dataen

#Logaritmisk transformasjon
data$DiceTransformed = log((max(data$f1_score)+1) - data$f1_score)

AOV.log <- aov(DiceTransformed ~n_filters*batch_norm + n_filters*dropout + n_filters*learning_rate + batch_norm*dropout +
               batch_norm*learning_rate + dropout*learning_rate, data = data)

# Kvadratrot transformasjon:
data$DiceTransformed_sqr = sqrt(1.2 - data$f1_score)

```

```r
AOV.sqrt <- aov(DiceTransformed_sqr ~n_filters*batch_norm + n_filters*dropout + n_filters*learning_rate + batch_norm*dropout +
                  batch_norm*learning_rate + dropout*learning_rate, data = data)

# Invers transformering
data$DiceTransformed_inv = 1/((max(data$f1_score)+1) - data$f1_score)

AOV.inv <- aov(DiceTransformed_inv ~n_filters*batch_norm + n_filters*dropout + n_filters*learning_rate + batch_norm*dropout +
                batch_norm*learning_rate + dropout*learning_rate, data = data)

# Plotter diagnoseplot for de ulike transformerte dataen ved å bytte ut modellen som plottes

# 1) Residuals vs. fitted
plot(AOV.inv, 1)

# 2) QQ-plot
plot(AOV.inv, 2)

#Beregner også en Anderson-Darling normalitetstest for de ulike transformasjonene
# Anderson-Darling
ad.test(AOV.inv$residuals)

# Foretar også en Box-Cox transformasjon
library(MASS)

# Plusser på en liten variabel for å få positive verdier for responsvariabelen
y <- data$f1_score + 0.5

#Definerer en modell
data_model <- aov(y ~ n_filters*batch_norm + n_filters*dropout + n_filters*learning_rate + batch_norm*dropout +
                    batch_norm*learning_rate + dropout*learning_rate, data = data)

# Finner lambda ved undersøke plottet
boxcox(data_model, lambda = seq(1, 1.5, by = 0.05), plotit = TRUE)

# Utfører BoxCox-transformasjonen ved å sette inn lambda

data_model_bc = aov((((y^1.4)-1) / 1.4) ~  n_filters*batch_norm + n_filters*dropout + n_filters*learning_rate + batch_norm*dropout
                    + batch_norm*learning_rate + dropout*learning_rate, data = data)

# Plotter diagnoseplot
# Residuals vs. fitted
plot(data_model_bc, 1)

#QQ-plot:
plot(data_model_bc, 2)

#Anderson-Darling
ad.test(data_model_bc$residuals)

```

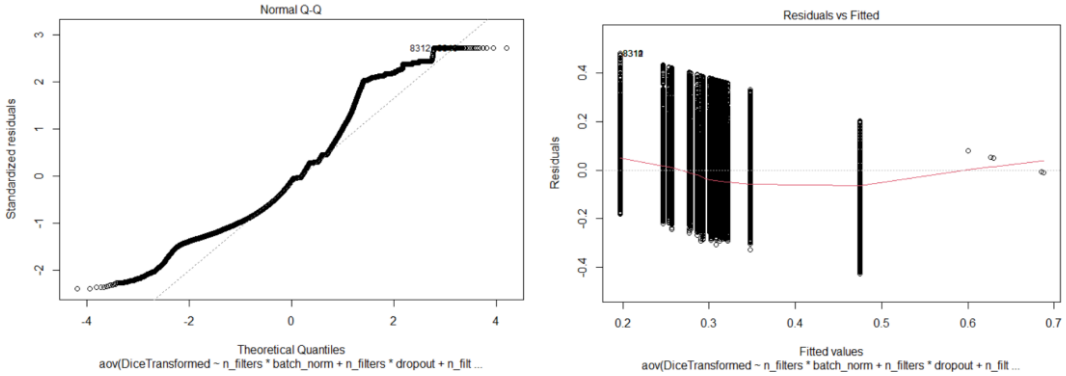
Diagnoseplott:



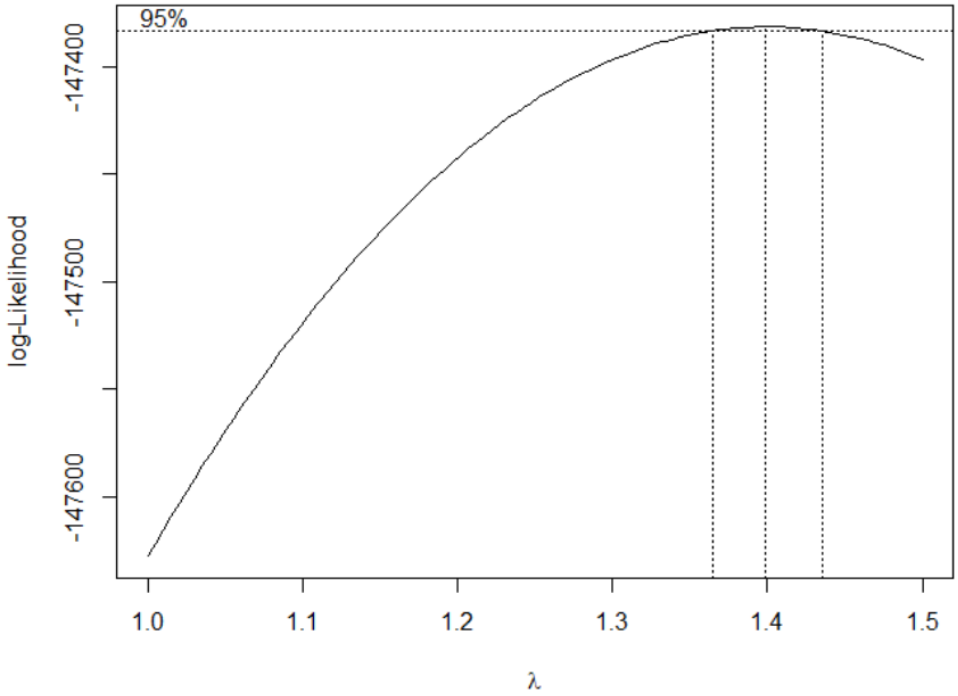
***Figur C.1**: Diagnoseplott for den kvadratrot-transformerte dataen. Figuren til venstre viser QQ-plottet og figuren til høyre viser plottet for residualer mot tilpasset modell.*



***Figur C.2**: Diagnoseplott for den invers-transformerte dataen. Figuren til venstre viser QQ-plottet og figuren til høyre viser plottet for residualer mot tilpasset modell.*



***Figur C.3:** Diagnoseplott for den log-transformerte dataen. Figuren til venstre viser QQ-plottet og figuren til høyre viser plottet for residualer mot tilpasset modell.*



Figur C.4: *Log-likelihood funksjonen som brukes til bestemmelse av lambda. Lambda-verdien som brukes i Box-Cox-transformasjonen bestemmes fra toppunktet i grafen.*

Statistiske tester: Friedmantest og Wilcoxon signed rank-test

```r
#Friedmantest og Wilcoxon signed rank test

#Laster inn dataen som er organinert i et unreplicated complete block design
data_nfilters <- mean_nfilters
data_lr<- mean_lr
data_batchnorm <- mean_batchnorm
data_dropout <- mean_dropout

#Laster inn nødvendige biblioteker
library(PMCMRplus)
library(ggpubr)
library(rstatix)


# Antall filtre

y_nfilter <- data_nfilters$f1_score

block_nfilter <- 1:1033

blocks_nfilter <- rep(block_nfilter,times=3)

groups_nfilter <- factor(c(rep("48", times=1033), rep("64", times=1033), rep("80", times=1033)))

# Friedmantest
friedman.t.chidist <- friedmanTest(y_nfilter, groups_nfilter, blocks_nfilter)
friedman.t.chidist[["p.value"]]

# Post-hoc med Nemenyi
nemenyi.all.comp <- frdAllPairsNemenyiTest(y_nfilter, groups_nfilter, blocks_nfilter, alternative = "two.sided")
nemenyi.all.comp

# Box-polt
ggboxplot(data_nfilters, x = "n_filters", y = "f1_score", add = "jitter")

# Beregner effektstørrelse
data_nfilters %>% friedman_effsize(f1_score ~ n_filters |Teller)


# Læringsrate
y_learningrate <- data_lr$f1_score

block_learningrate <- 1:1033

blocks_learningrate <- rep(block_learningrate,times=3)

groups_learningrate <- factor(c(rep("1,00E-03", times=1033), rep("1,00E-04", times=1033), rep("1,00E-05", times=1033)))

# Friedman
friedman.t.chidist <- friedmanTest(y_learningrate, groups_learningrate, blocks_learningrate)
friedman.t.chidist[["p.value"]]

# Post-hoc med Nemenyi
nemenyi.all.comp <- frdAllPairsNemenyiTest(y_learningrate, groups_learningrate, blocks_learningrate, alternative = "two.sided")
nemenyi.all.comp

# Box-plot
ggboxplot(data_lr, x = "learning_rate", y = "f1_score", add = "jitter")

# Beregner effektstørrelse
data_lr %>% friedman_effsize(f1_score ~ learning_rate |Teller)


# Batchnomalisering

#summary statistics
data_batchnorm %>%
  group_by(batch_norm) %>%
  get_summary_stats(f1_score, type = "median_iqr")

# Visualisering med box-plot
bxp <- ggpaired(data_batchnorm, x = "batch_norm", y = "f1_score",
                order = c("yes", "no"),
                ylab = "f1_score", xlab = "batch_norm")
bxp

# Wilcoxon signed rank test
stat.test <- data_batchnorm  %>%
  wilcox_test(f1_score ~ batch_norm, paired = TRUE) %>%
  add_significance()
stat.test

#Beregner effektstørrelse
data_batchnorm  %>%
  wilcox_effsize(f1_score ~ batch_norm, paired = TRUE)


# Dropout

#summary statistics
data_dropout %>%
  group_by(dropout) %>%
  get_summary_stats(f1_score, type = "median_iqr")

#Visualisering med box-plot
bxp <- ggpaired(data_dropout, x = "dropout", y = "f1_score",
                order = c("0", "0,5"),
                ylab = "f1_score", xlab = "dropout")
bxp

```

```r

#Wilcoxon signed rank test
stat.test <- data_dropout  %>%
  wilcox_test(f1_score ~ dropout, paired = TRUE) %>%
  add_significance()
stat.test

# Beregner effektstørrelse
data_dropout  %>%
  wilcox_effsize(f1_score ~ dropout, paired = TRUE)

```

Vedlegg D Resultater for testsett og Maastro-testsett

Vedlegg D inneholder en oversikt over gjennomsnittlig ytelsesmål per pasient for den beste modellen uten og med augmentering, og vurdert på testsettet og det eksterne Maastro-testsettet. Fiolinplott for Dice-score per tverrsnitt for hver pasient i Maastro-testsettet vises også her.

Testsett

Tabell D.1: *Ytelsesmål per pasient for de beste modellene med og uten bildeaugmentering vurdert på testsettet.*

	Modell uten augmentering			Modell med augmentering		
Pasient	Dice-score	HD_{95} [mm]	MSD [mm]	Dice-score	HD_{95} [mm]	MSD [mm]
5	0,730	30,0	2,24	0,747	31,4	2,00
8	0,736	24,4	1,00	0,758	22,2	1,00
13	0,759	22,1	1,00	0,752	21,2	1,00
16	0,442	19,1	4,00	0,491	22,0	3,00
18	0,797	12,6	0,000	0,819	2,83	0,000
21	0,694	35,1	1,00	0,623	39,1	2,00
36	0,645	15,1	2,00	0,739	17,9	1,00
44	0,792	3,32	0,000	0,805	4,00	0,000
52	0,826	4,24	1,00	0,769	4,00	1,00
55	0,784	4,12	0,000	0,767	3,74	0,000
60	0,778	9,85	1,00	0,771	11,0	1,00
61	0,783	3,46	0,000	0,716	15,0	1,00
67	0,647	47,8	2,00	0,740	38,7	1,00
73	0,697	4,00	1,00	0,716	3,00	1,00
74	0,753	14,4	1,00	0,796	10,5	1,00
77	0,831	4,00	1,00	0,851	3,61	1,00
82	0,541	27,0	4,12	0,539	13,0	4,00
91	0,806	24,9	1,00	0,817	15,1	1,00
93	0,275	27,6	11,2	0,324	24,2	3,00
99	0,885	3,00	0,000	0,864	6,40	0,000
110	0,003	41,3	27,3	0,003	41,0	27,8
116	0,643	21,7	2,00	0,625	42,0	2,24
120	0,812	9,11	1,00	0,842	10,7	0,000
130	0,684	12,0	0,000	0,728	4,24	0,000
140	0,697	18,5	1,41	0,785	19,5	1,00
148	0,699	35,7	1,00	0,755	32,1	1,00
153	0,713	4,58	0,000	0,693	2,45	0,000
154	0,744	10,3	0,000	0,781	3,61	0,000
162	0,804	30,0	1,00	0,776	39,0	1,00
164	0,764	14,2	1,00	0,787	38,6	1,00
169	0,742	52,0	2,00	0,820	42,5	1,00
184	0,898	18,7	0,000	0,862	7,91	1,00
191	0,809	5,66	0,000	0,788	25,7	1,00
194	0,645	23,7	1,41	0,730	10,6	1,00
209	0,804	6,71	1,00	0,830	63,0	1,00

217	0,761	6,40	1,00	0,857	4,00	0,000
223	0,702	32,2	0,000	0,693	29,8	1,00
233	0,839	14,3	1,00	0,877	14,0	0,000
242	0,799	37,2	1,00	0,784	38,5	1,00
249	0,780	30,8	1,00	0,814	31,3	1,00

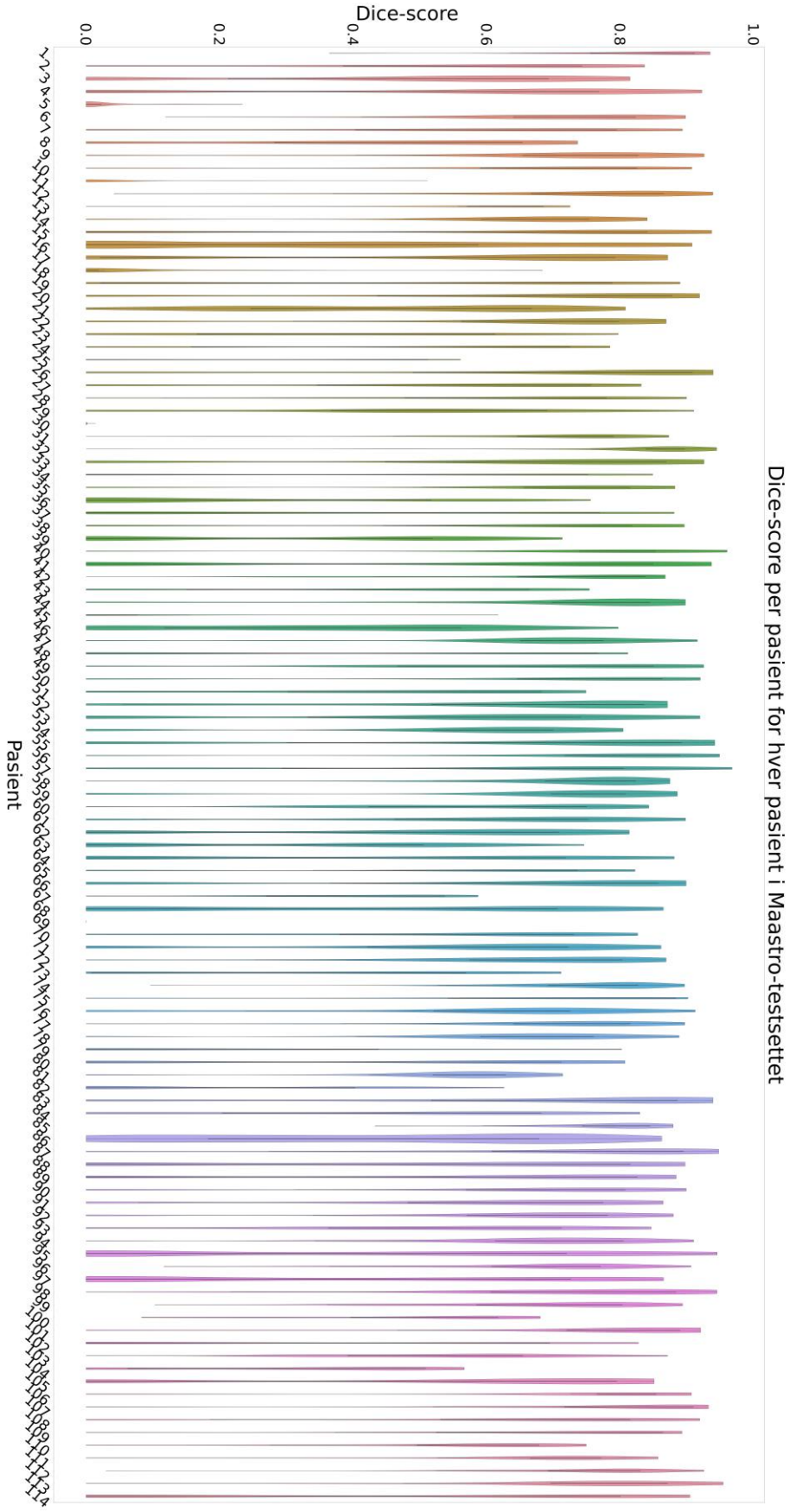
Maastro-testsett

Tabell D.2: *Ytelsesmål per pasient for de beste modellene med og uten bildeaugmentering vurdert på Maastro-testsettet.*

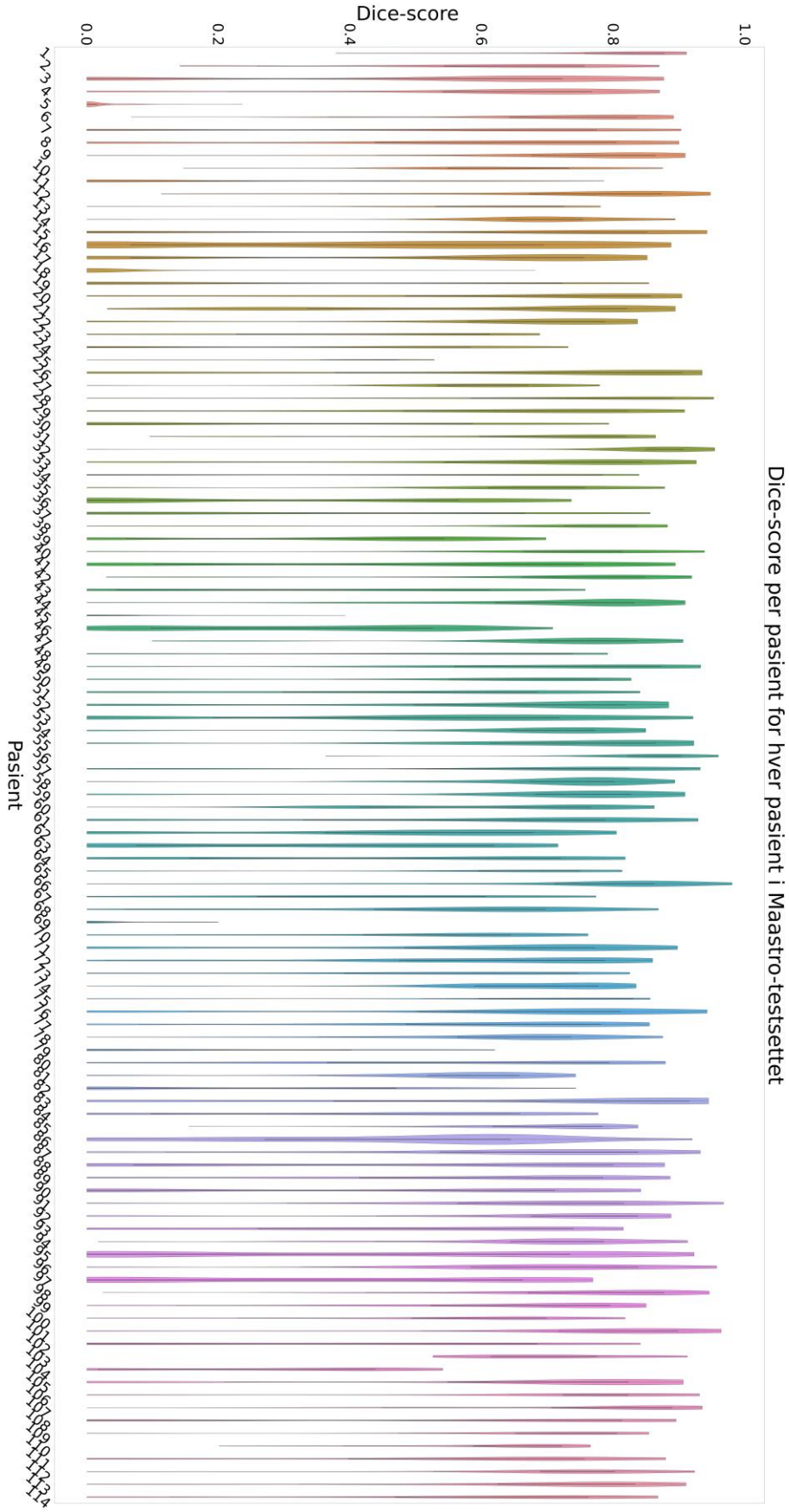
	Modell uten augmentering			Modell med augmentering		
Pasient	Dice-score	HD_{95} [mm]	MSD [mm]	Dice-score	HD_{95} [mm]	MSD [mm]
1	0,863	4,00	0,000	0,829	11,6	1,00
2	0,750	20,9	1,00	0,672	20,2	1,00
3	0,516	55,4	5,66	0,607	29,4	1,73
4	0,818	5,10	1,00	0,822	28,6	1,00
5	0,348	36,9	13,9	0,442	19,9	1,41
6	0,588	34,3	2,00	0,698	30,6	1,00
7	0,408	8,77	4,24	0,366	21,8	4,47
8	0,732	29,8	1,00	0,803	13,2	1,00
9	0,811	6,08	0,000	0,781	2,45	0,000
10	0,882	8,06	0,000	0,855	22,8	0,000
11	0,725	20,5	1,00	0,733	51,9	0,000
12	0,799	12,7	1,00	0,751	6,00	1,00
13	0,035	18,4	7,07	0,210	69,7	30,4
14	0,628	7,35	2,00	0,683	4,69	1,00
15	0,705	4,12	1,00	0,655	5,74	1,00
16	0,767	22,1	1,00	0,734	15,9	1,00
17	0,820	7,00	1,00	0,745	27,1	1,00
18	0,645	37,2	1,00	0,685	6,78	0,000
19	0,777	20,0	1,00	0,795	19,1	1,00
20	0,633	6,93	1,00	0,654	2,45	1,00
21	0,708	8,19	1,00	0,723	9,00	1,00
22	0,796	2,24	0,000	0,796	14,3	0,000
23	0,343	21,0	5,48	0,450	26,3	5,74
24	0,702	42,3	1,00	0,692	9,70	1,00
25	0,067	16,3	3,00	0,050	55,4	2,00
26	0,679	4,49	0,000	0,521	61,4	1,41
27	0,543	45,6	2,83	0,598	49,6	2,00
28	0,748	31,8	2,24	0,748	27,9	2,00
29	0,561	9,00	2,24	0,715	7,07	1,00
30	0,721	44,1	1,00	0,704	45,4	1,00
31	0,448	48,5	2,00	0,474	23,3	1,41
32	0,654	4,69	1,00	0,500	33,2	2,00
33	0,379	1,49	0,000	0,401	25,1	1,00
34	0,850	5,48	1,00	0,836	4,90	1,00

35	0,612	49,0	1,41	0,613	8,41	1,41
36	0,673	33,0	5,00	0,797	25,0	1,00
37	0,571	20,0	2,83	0,726	15,6	1,00
38	0,696	46,9	1,00	0,700	43,2	1,00
39	0,000	39,4	24,5	0,361	27,8	2,83
40	0,734	20,9	1,00	0,716	16,6	1,00
41	0,874	5,83	0,000	0,882	3,16	0,000
42	0,789	31,6	1,00	0,786	27,8	1,00
43	0,671	3,16	1,00	0,685	2,83	0,000
44	0,769	3,16	1,00	0,708	4,00	1,00
45	0,316	46,7	3,00	0,427	43,9	3,00
46	0,496	40,5	1,00	0,411	8,94	2,00
47	0,765	10,3	1,00	0,775	18,7	1,00
48	0,396	15,0	3,61	0,435	22,0	3,61
49	0,650	17,4	1,00	0,685	13,7	0,000
50	0,818	2,83	0,000	0,772	4,47	0,000
51	0,583	22,7	1,00	0,534	41,6	1,00
52	0,741	12,8	1,00	0,733	8,60	1,00
53	0,529	24,0	1,00	0,399	38,1	2,83
54	0,774	21,8	1,00	0,763	19,5	1,00
55	0,181	18,4	9,70	0,087	28,5	17,0
56	0,412	11,6	5,00	0,370	62,6	5,74
57	0,726	9,27	1,41	0,754	5,92	1,00
58	0,600	2,24	0,000	0,633	28,3	0,000
59	0,765	25,5	1,00	0,794	64,5	1,00
60	0,021	56,2	9,27	0,019	56,8	14,3
61	0,762	46,4	1,00	0,667	41,6	2,00
62	0,531	47,5	3,00	0,585	51,2	2,00
63	0,788	7,35	1,00	0,783	6,00	1,00
64	0,662	41,9	1,00	0,647	40,6	1,41
65	0,646	17,5	1,41	0,715	17,7	1,00
66	0,843	23,5	0,000	0,829	12,4	1,00
67	0,824	27,7	0,000	0,863	7,87	0,000
68	0,718	7,00	0,000	0,739	17,5	0,000
69	0,752	5,10	1,00	0,742	4,47	1,00
70	0,784	4,58	1,00	0,791	4,00	0,000
71	0,734	40,5	1,41	0,750	41,2	1,00
72	0,695	10,8	1,00	0,708	10,8	1,00
73	0,689	34,5	1,00	0,706	36,7	1,00
74	0,565	11,9	1,00	0,535	12,0	1,00
75	0,360	36,2	8,94	0,456	40,4	5,10
76	0,464	20,7	5,00	0,498	22,5	2,24
77	0,643	40,0	1,00	0,678	2,83	0,000
78	0,818	22,6	1,00	0,811	21,6	1,00
79	0,418	10,4	2,24	0,464	38,3	2,00
80	0,558	51,0	2,00	0,608	44,1	1,73
81	0,000	64,4	30,3	0,034	61,0	27,1
82	0,686	32,2	1,00	0,663	57,2	1,00

83	0,594	19,3	2,45	0,533	49,5	3,61
84	0,700	5,48	1,00	0,746	5,00	0,000
85	0,740	8,83	1,00	0,686	39,5	1,41
86	0,451	4,12	1,00	0,604	4,12	1,00
87	0,765	23,1	1,00	0,702	11,4	1,00
88	0,779	2,24	0,000	0,704	19,4	0,000
89	0,673	40,3	1,41	0,735	27,5	1,00
90	0,755	16,0	1,00	0,688	18,0	1,41
91	0,703	4,58	1,00	0,672	4,47	1,00
92	0,351	12,1	1,00	0,267	16,8	1,73
93	0,516	43,8	4,24	0,648	43,6	2,00
94	0,645	39,7	2,24	0,670	51,0	3,39
95	0,570	35,0	2,00	0,586	32,2	2,00
96	0,272	34,9	1,41	0,303	42,6	1,00
97	0,820	46,4	1,00	0,838	7,35	1,00
98	0,582	30,0	1,00	0,602	14,2	1,00
99	0,795	5,39	1,00	0,725	9,95	1,41
100	0,538	15,7	2,24	0,521	16,6	2,24
101	0,841	48,8	1,00	0,684	84,3	10,3
102	0,680	49,1	1,00	0,715	32,1	0,000
103	0,742	16,5	1,00	0,720	15,0	1,00
104	0,778	5,10	1,00	0,796	10,1	1,00
105	0,741	46,1	1,00	0,602	70,8	2,00
106	0,679	29,7	1,41	0,723	17,0	1,00
107	0,735	8,94	1,00	0,767	8,19	0,000
108	0,645	28,6	1,41	0,648	28,9	1,41
109	0,698	19,2	1,00	0,690	40,3	1,00
110	0,517	12,4	1,00	0,522	13,9	1,00
111	0,711	50,8	1,41	0,726	59,8	1,73
112	0,599	15,6	1,00	0,515	16,0	1,00
113	0,802	8,00	1,00	0,806	9,90	0,000
114	0,733	10,2	1,00	0,719	6,32	1,00



Figur D.1: *Fiolinplott som illustrerer fordelingen av Dice-score per tverrsnitt for hver pasient i Maastro-testsettet for den beste modellen uten augmentering. Lengden på fiolinene representerer variasjonen i gjennomsnittlig Dice-score for hver enkelt pasient, mens tykkelsen representerer hvor mange bildetverrsnitt som får denne Dice-scoren.*



Figur D.2: *Fiolinplott som illustrerer fordelingen av Dice-score per tverrsnitt for hver pasient i Maastro-testsettet for den beste modellen med augmentering. Lengden på fiolinene representerer variasjonen i gjennomsnittlig Dice-score for hver enkelt pasient, mens tykkelsen representerer hvor mange bildetverrsnitt som får denne Dice-scoren.*

Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway