Høgskulen på Vestlandet

BACHELOROPPGAVE

Nevralt Nettverk for å Identifisere CO2 i Karbonlagringseksperimenter

Neural Network-Enhanced CO2 Detection in Carbon Storage Experiments

Øyvind Aasmul Onarheim

Andre Kristopher Ripman

Tor Magne Solheimsnes

Fakultetet for Teknologi, Miljø-, og Samfunnsvitenskap

Institutt for Datateknologi, Elektroteknologi og Realfag

Veileder: Erlend Raa Vågset

Innleveringsdato: 13.05.2024

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: Nevralt Nettverk for å Identifisere CO2 i Karbonlagringseksperimenter	*Dato:* 13.05.2024
Forfatter(e): Øyvind Aasmul Onarheim Tor Magne Solheimsnes Andre Kristopher Ripman	
Studieretning: Informasjonsteknologi	*Antall sider u/vedlegg: 72*
Kontaktperson ved studieretning: Per Christian Engedal	*Antall sider vedlegg: 42*
Merknader: Ingen	*Antall disketter/CD-er:* 0
	Gradering: Ingen

Oppdragsgiver: Erlend Storvik	*Oppdragsgivers referanse:* Ingen
Oppdragsgivers kontaktperson: Erlend Storvik	*Telefon:* +47 93644984

Sammendrag:

Dette bachelorprosjektet har som mål å lage en maskinlæringsmodell som kan identifisere forekomsten av CO2 i bilder tatt av FluidFlower-modellen til Universitet i Bergen, og returnere en binær representasjon av bildet. I dette prosjektet er det utforsket ulike algoritmer og dataaugmenteringer for å best mulig trene et nevralt nettverk til å nå dette målet. FluidFlower-modellen brukes i en pågående forskning på karbonlagring i simulerte geologiske omgivelser, der det blir observert hvordan CO2 oppfører seg i disse omgivelsene. Formålet med forskningen er å bedre forståelsen for karbonlagring, og hvordan dette kan gjøres på en forsvarlig måte.

Stikkord:

Maskinlæring og nevrale nettverk	Bildesegmentering	Karbonlagring

Høgskulen på Vestlandet, Fakultet for teknologi, miljø- og samfunnsvitenskap

Postadresse: Postboks 7030, 5020 BERGEN Besøksadresse: Inndalsveien 28, Bergen

Tlf. 55 58 75 00 Fax 55 58 77 90 E-post: post@hvl.no Hjemmeside: http://www.hvl.no

FORORD

Denne rapporten dokumenterer arbeidet gjort på bachelorprosjektet *«Nevralt Nettverk for å Identifisere CO2 i Karbonlagringseksperimenter»* Med dette prosjektet konkluderes vår bachelorgrad i informasjonsteknologi med spesialisering innen maskinlæring og kunstig intelligens, ved Høgskulen på Vestlandet.

Vi ønsker å takke alle som har bidratt til gjennomføring av dette prosjektet. En spesiell takk går ut til Erlend Raa Vågset og Erlend Storvik for sine roller som veileder og oppdragsgiver, og for sine gode råd og konstruktive tilbakemeldinger gjennom prosjektets levetid. Uten deres kunnskap innen maskinlæring og karbonlagring, ville dette prosjektet vært langt mer krevende.

Avslutningsvis ønsker vi å takke hverandre for et vell gjennomført prosjekt, der gruppedynamikken vår har vært avgjørende for prosjektets suksess, og vi er takknemlig for lærdommen vi har opparbeidet oss gjennom dette prosjektet.

INNHOLDSFORTEGNELSE

Innholdsfortegnelse Figur

Innholdsfortegnelse Tabell

Ordliste

Begrep	Forklaring
Darcy skala	Et porøst medium sett på avstand.
Dataaugmentering	Er en teknikk for å øke mengden treningsdata tilgjengelig for en AI-modell
Fluid Flower	Modell hos UiB for å etterligne sandlag og geologiske formasjoner, for å simulere karbonlagring i liten skala.
Morfologisk dilasjon	Gjør objekter mer synlig, og fyller igjen små hull i objekter
Morfologisk erosjon	Fjerner flytende piksler og tynne linjer.
Multikromatisk	Noe som involverer mer enn en farge
Generalisering	En maskinlæringsmodells evne til å tilpasse seg til ny og usett data
Geomekaniske egenskaper	Referere til de elastiske egenskapene i materialet
Høynivå	Fjerner mye av den komplekse og underliggende logikken. Mindre detaljert kontroll over modell og arkitektur
Kaggle	En online plattform for datavitenskap og maskinlæring, kjent for å arrangere konkurranser, tilby datasett og notebooks
Karbondioksid	En gass som dannes ved forbrenningsprosesser.
Lag i nevrale nettverk	Lagene i et nevralt nettverk referere til en samling noder som jobber sammen på en gitt dybde i nettverket.
Lavnivå	Beholder mye av den komplekse og underliggende logikken. Mer detaljert kontroll over modell og arkitektur.
Maske	Et referansebilde, brukes som et fasitsvar for modellen under trening
Morfologisk operasjon	En klasse av algoritmer som behandler bilder basert på formene i dem. Disse operasjonene behandler bilder ved å anvende et strukturelt

Begrep	Forklaring
	element eller kjerne til bildepunkter (pixels), som gir nye verdier basert på de geometriske strukturene i nærheten.
Node	Node er et punkt i en elektrisk krets der to eller flere kretselementer kobles sammen
Permeabilitet	Et uttrykk for cellemembranens gjennomtrengelighet for ulike stoffer
Plumer	En lang sky med røyk eller damp som ligner på en fjær når den sprer seg.
Porøsitet	Porøsiteten gir et mål på hvor mye væske eller gass en bergart kan inneholde
ResNet	ResNet er en Deep Learning modell som er bygget på CNN arkitektur og som ofte blir brukt til bildebehandling
Superkritisk væske	Væske med temperatur og trykk over henholdsvis kritisk temperatur og trykk, der trykket ikke er stort nok til å komprimere det til et fast stoff.
TDS	Den totale konsentrasjonen av oppløste stoffer i en væske
Torchvision	Verktøy for augmentering av bilder fra PyTorch.

Akronymliste

Begrep	Forklaring
AI	Artificial Intelligence
atm	Atmosfærisk trykk
BGR	Blå-Grønn-Rød
BRG	Blå-Rød-Grønn
CNN	Convolutional Neural Network - en klasse av dype nevrale nettverk som er mest brukt til å analysere bilder.
CNN	Konvolusjonell Nevralt Nettverk
CO2	Karbondioksid, en gass som dannes ved forbrenningsprosesser.
CRISP-DM	Cross-industry standard process for data mining
DNN	Dypt Nevralt Nettverk
GBR	Grønn-Blå-Rød
GPU	Grafikkort
GRB	Grønn-Rød-Blå
HVL	Høgskulen på Vestlandet
KI	Kunstig intelligens
ML	Maskinlæring
psi	Pounds per square inch, pund per kvadrattomme
RBG	Rød-Blå-Grønn
RGB	Rød-Grønn-Blå
UiB	Universitet i Bergen

1. Innledning

1.1 Kontekst

Klimaendringer bidrar direkte til humanitære kriser som hetebølger, skogbrann, flom, tropiske stormer og orkaner, og de øker i skala, frekvens og intensitet (WHO, 2023). Det er presentert flere forslag på hvordan det skal reduseres CO2 utslipp for å minske klimaendringene som skjer rundt oss. En mulig løsningene er å benytte seg av det som kalles geologisk karbonlagring, som i dette tilfellet vil si å lagre CO2 i gamle oljereservoarer (USGS, u.å.).

1.2 Motivasjon

For å forstå og forbedre effektiviteten av CO2-lagring, er det essensielt å kunne overvåke og kvantifisere distribusjonen av CO2 som blir plassert i reservoarer. På Universitetet i Bergen (UiB) gjennomføres det eksperimenter for å visualisere CO2-distribusjon i sandlag ved hjelp av en tank som er laget for å simulere hvordan lagring av CO2 vil foregå på en større skala. Disse eksperimentene produserer bilder som inneholder informasjon om CO2-tilstedeværelse, konsentrasjon og dybde.

Oppdragsgiver er en del av en forskningsgruppe som består av forskere fra HVL og UIB, som forsker på karbonlagring og hvordan dette kan gjøres på en forsvarlig måte uten konsekvenser for økosystemet rundt. Formålet med forskningen er å bedre forståelsen av geologisk karbonlagring. Dette gjøres spesifikt gjennom lab-eksperimenter der de første ut-dataene er bilder av CO2 som på forskjellige måter er fanget i sandlag. For å videre analysere og forstå hva som har skjedd i eksperimentene analyseres disse bildene, og sammenlignes opp mot matematiske og fysiske modeller som er satt opp for å predikere (kunne forutsi noe om) hvordan CO2 beveger seg i sandlagene. Disse matematiske modellene baserer seg nesten utelukkende på det som kan klassifiseres som homogeniserte eller oppskalerte modeller, som kalles "Darcy-skala" på fagspråket. Disse modellene fokuserer da ikke på væskeflyt mellom enkelte sandkorn, men heller på den overordnede strukturen til flyten (You, *et al*, 2021).

Det er et pågående forskningsprosjekt om hvordan CO2 kan lagres i underjordiske geologiske formasjoner. Under disse eksperimentene blir det produsert store mengder data, og det blir tatt tusenvis av bilder som skal analyseres. Det er et stort behov for å effektivisere og automatisere analysen av disse bildene, og i den forbindelse skal vi lage en løsning som gjør nettopp dette.

1.3 Prosjekteier

Erlend Storvik stiller som prosjekteier og oppdragsgiver, samtidig stiller Erlend Raa Vågset som veileder. Erlend Storvik deltar i prosjektet som ekspert innenfor matematikk og karbonlagring, mens Erlend Raa Vågset deltar som veileder, og ekspert innen maskinlæring.

1.4 Problembeskrivelse og mål

Formålet med dette prosjektet er å finne en løsning for å identifisere hvor CO2 befinner seg i bilder tatt av FluidFlower-tanken til UiB. Dette er en fysisk tank som simulerer karbonlagring i geologiske formasjoner (UiB, u.å). Bildene sammenlignes så opp mot matematiske og fysiske modeller som er satt opp for å predikere hvordan CO2 beveger seg i de geologiske formasjonene. Et av de første stegene i bildeanalysen er å identifisere hvor det er CO2, sett fra øynene til en oppskalert modell. Dette prosjektet har som mål å lage et nevralt nettverk som mottar et bilde fra et slikt eksperiment og returnerer informasjon om hvor det er CO2. Dersom tiden tillater det, skal løsningen også kunne identifisere de forskjellige sandlagene. Oppdragsgiver har som mål at forskningsgruppen kan gjøre videre kalkulasjoner basert på resultatene produsert i dette prosjektet.

1.5 Oppbygging av rapporten

Kapittel 1 - Innledning: Introduserer prosjektet og forteller om motivasjon, hvem prosjektet er interessant for og formålet med gjennomføringen.

Kapittel 2 - Prosjektbeskrivelse: Går inn på bakgrunnen for prosjektet og eventuelt tidligere arbeid, forskning, teorier og kunnskap prosjektet skal bygge videre på.

Kapittel 3 - Teoretisk Bakgrunn: Går igjennom nødvendig teoretisk bakgrunn for å kunne forstå konteksten til prosjektet.

Kapittel 4 - Prosjektdesign: Dette kapittelet beskriver teknologier og metoder som er brukt for å gjennomføre det ingeniørfaglige arbeidet.

Kapittel 5 - Detaljert Løsning: Dette kapittelet går igjennom hvordan prosjektresultatene har blitt oppnådd ved å beskrive valgt løsning.

Kapittel 6 - Resultater: Går igjennom resultatene som har blitt oppnådd i løpet av prosjektet, og hvilke evalueringsmetoder som er brukt for å verifisere og validere prosjektet.

Kapittel 7 - Diskusjon: Her diskuteres de ulike resultatene, samt det blir gitt en sluttevaluering av prosjektet.

Kapittel 8 - Konklusjon: Dette kapittelet presenterer konklusjonen av arbeidet, og anbefalinger for videre arbeid.

2. Prosjektbeskrivelse

Dette kapittelet går igjennom den praktiske bakgrunnen til prosjektet, samt hvilke begrensinger som har blitt satt for prosjektet og hvilke ressurser som blir tildelt gruppen. Avslutningsvis beskrives relevant litteratur for prosjektet.

2.1 Praktisk bakgrunn

Her presenteres tidligere arbeid som relateres til dette prosjektet, samt initialt krav og løsningsforslag som ble presentert av oppdragsgiver i forkant av prosjektet.

2.1.1 Tidligere arbeid

En tidligere løsning på problembeskrivelsen som er presentert i 1.4, var å benytte teknikker som matematisk regularisering, glatting og segmentering for å analysere bilder. Det ble imidlertid fastslått av oppdragsgiver at disse metodene ikke var skalerbare, da hvert bilde måtte bli behandlet manuelt. Dette gjorde prosessen uegnet for standardisering og automatisering i større skala.

2.1.2 Initielle krav

Kravene som legges til grunn for denne oppgaven, er å bruke en maskinlæringsmodell, eller et nevralt nettverk, som kan analysere bilder tatt av FluidFlower-tanken til UiB. Grunnleggende krav til en vellykket løsning er at den skal detektere hvor det er CO2-forekomst, samt fjerne støy i form av sandkorn for å beholde den globale strukturen.

2.1.3 Initielle løsnings-idé

Forslag fra oppdragsgiver er å bruke maskinlæringsalgoritmer, nærmere bestemt et nevralt nettverk. Løsning skal kunne motta et bilde fra eksperimentet og deretter returnere en sann/usann representasjon av tilstedeværelsen til CO2. Oppdragsgiver foreslår videre å bruke Python som programmeringsspråk og tilhørende rammeverk/biblioteker.

2.2 Begrensninger

Som en del av prosjektarbeidet settes det begrensninger til utførelse av arbeid. Trening av maskinlæringsmodellene kan kreve kraftig maskinvare som gruppen ikke har tilgjengelig. Dette kan sette begrensninger på trening av modellene, da det er nødvendig å finne alternative måter å trene den på.

Utvikling av nevrale nettverk og maskinlæringsmodeller krever spesifikk kompetanse. Dette kan påvirke fremgangen av prosjektet, da prosjektgruppen må opparbeide seg nødvendig kunnskap. Det vurderes at det er mer hensiktsmessig å bruke eksisterende nevrale nettverk som er spesialisert på bildesegmentering, fremfor en egenutviklet løsning. Manglende forkunnskap kan føre til at det ikke blir funnet best mulig løsning for prosjektet.

2.3 Ressurser

For å kunne jobbe med, og fullføre prosjektet, er det nødvendig med tilgang til både fysiske, tekniske og faglige ressurser. Dette kapittelet vil gå igjennom ressursene som brukes for utviklingen av en ferdig løsning.

Fysiske ressurser

Det kan kreves betydelig med datamaskinkraft for å trene en dyplæringsmodell. Dette var noe gruppen ikke hadde tilgang til fysisk, derav ble det brukt en nettside som heter Kaggle for å lage Kaggle-notebooks. Kaggle er en interaktiv notatbok som både kan kjøre, og vise resultatet av Python kode. Dette er en gratistjeneste, der en kan bruke Kaggle sin eksterne Grapichal Processing Unit (GPU) for å få tilstrekkelig ytelse til å trene modellen (Staff, 2023).

Datasett

Datasettet er skapt i forbindelse med et pågående forskningsprosjekt, som undersøker hvordan CO2 oppfører seg i geologiske formasjoner (Fernø, *et al*, 2024). For å bygge en dyplæringsmodell er det essensielt med store datasett, slik at modellen får trent tilstrekkelig for å kunne generalisere til usett data (InfoSysBPM, u.å). Det ble tildelt datasett med totalt 685 forskjellige bilder, med muligheter for flere dersom det skulle være behov for det. Innholdet i datasettet er beskrevet nærmere i punkt 3.4.

Veiledning og kommunikasjon

Kommunikasjon i gruppen gjøres ved fysisk oppmøte på Høgskulen på Vestlandet i Førde. Ellers brukes Discord eller Teams dersom det ikke er mulig å møte fysisk.

Både oppdragsgiver og veileder bistår med prosjektskriving. Oppdragsgiver bistår primært med forståelse for datasett og den bak omliggende forskningen på karbonlagring, mens veileder bistår hovedsakelig til strukturering og skriving av rapport.

2.4 Litteratur om problemstillingen

Under forarbeidet til prosjektet, ble det funnet forskningsartikler relatert til maskinlæring og karbonlagring. Da karbonlagring og maskinlæring er hovedfokuset i denne oppgaven, er det valgt å presentere en forskning om maskinlæring for å karakterisere CO2 i karbonlagring, og en forskning fra Equinor som omhandler forskning på karbonlagring.

Hanlin Sheng, Xinming Wu, Xiaoming Sun, og Long Wu presenterte en studie i 2023, " Deep learning for characterizing CO2 migration in time-lapse seismic images" (Sheng, *et al*, 2023).

Denne forskningsartikkelen deler elementer med bachelorprosjektet, spesielt bruken av dyplæringsmodeller for observasjon og overvåking av CO2-lagring. Til tross for likhetene, skiller den seg fra prosjektoppgaven på to hovedområder: datatypene og hvordan dataen brukes. I denne studien blir det brukt seismisk data som er hentet fra faktiske geologiske formasjoner, mens i bachelorprosjektet blir det brukt data i form av bilder som er hentet ut fra en simulert geologisk formasjon.

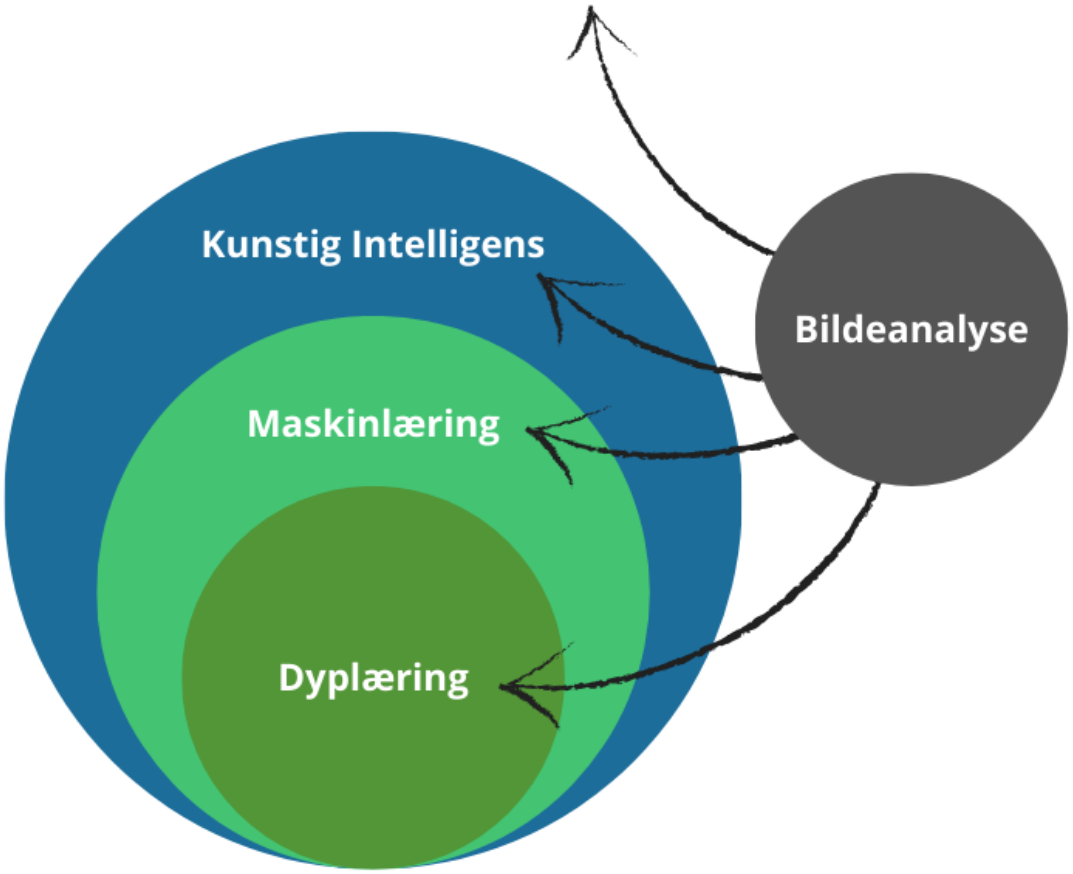
Prosjektet som er presentert i denne rapporten baserer seg på data som er hentet i forbindelse med et pågående forskningsprosjekt fra UiB, Room-Scale CO2 Injections in a Physical Reservoir Model with Faults (Fernø, 2024), som oppdragsgiver er en del av. I dette forskningsprosjektet blir det brukt 2D-bilder for å identifisere CO2 og sandlag, mens forskningen " Deep learning for characterizing CO2 migration in time-lapse seismic images" anvender 3D (eller 4D om man inkluderer tid som en dimensjon) seismiske data for å detektere CO2-plumer, noe som kan tillate en mer omfattende og detaljert analyse av CO2-migrasjon.

3. Teoretisk Bakgrunn

Dette kapittelet gir en innføring i den teoretiske bakgrunnen som er nødvendig for å bedre forstå hva som har blitt jobbet med gjennom prosjektet, og prosjektets nytteverdi. Først snakkes det generelt om AI, maskinlæring og dyplæring, før vi går inn på maskinlæring sin rolle innen forskning, og mer spesifikt innenfor CO2-lagring.

Videre beskrives det hvordan maskinlæring har forbedret bildeanalyse innenfor forskning, og viktigheten av nøyaktig bildeanalyse i kontekst av CO2-lagring. Deretter snakkes det grunnleggende om hva CO2-lagring er og hvordan det blir brukt. Avslutningsvis blir det snakket om datasettet som har blitt utlevert i forbindelse med oppgaven, og hvordan dette datasettet har blitt til.

3.1 Maskinlæring og bildeanalyse



Figur 3-1 - Figuren viser forholdet mellom kunstig intelligens, maskinlæring og dyplæring, og hvordan bildeanalyse benytter teknikker fra alle de tre områdene.

Her blir det gitt en innføring i kunstig intelligens, maskinlæring og dyplæring, samt hvordan dette blir brukt innen forskning. Det blir deretter vist til hvordan maskinlæring har forbedret bilde analyse, og hvorfor det er viktig med nøyaktig bildeanalyse i konteksten av prosjektet. Avslutningsvis snakkes det om bildeanalyse i ustrukturert data.

3.1.1 Kunstig intelligens

Kunstig intelligens (KI) er en samlebetegnelse på teknologier som lar maskiner simulere menneskelig intelligens og problemløsningsevner. På egen hånd, eller kombinert med andre teknologier som for eksempel sensorer eller roboter, kan KI utføre handlinger som normalt trenger menneskelig intelligens eller påvirkning. Bruken av KI blir mer og mer utbredt i samfunnet, og et par eksempler på dette er digitale assistenter, autonome biler eller generative KI-modeller som eksempelvis "ChatGPT". Som illustrert i figur 3-1, er KI et vidt begrep og er en samlebetegnelse for mange former av algoritmer, der blant annet maskinlæring og dyplæring er en underkategori. Dyplæringsalgoritmene er laget på en slik måte at de skal simulere den menneskelige hjernen, og etterligne hvordan den tar valg og avgjørelser som gjør at den kan ha en tilsynelatende intelligent oppførsel. På denne måten benytter KI seg av store datasett og lærer fra denne dataen over tid. Dermed kan KI-algoritmer komme frem til mer nøyaktige klassifikasjoner og prediksjoner enn tradisjonelle algoritmer, innenfor blant annet bildeanalyse og tekstanalyse (IBM, u.å. a).

3.1.2 Maskinlæring

Maskinlæring er en gren av Kunstig intelligens (KI) der målet er å utvikle algoritmer og modeller som gjør at maskinen kan etterligne hvordan mennesker lærer, og gradvis forbedrer nøyaktigheten til modellen. Som følge av dette skal modellen kunne gjøre valg eller analysere data uten å spesifikt programmere den for å gjøre de spesifikke valgene eller analysene. Dette kan gjøres på flere forskjellige måter, men den grunnleggende ideen er at datamaskinen skal analysere mønstre i data, forhold og avhengigheter mellom forskjellige datapunkter og trekke ut informasjon og konklusjoner basert på dette. Innenfor maskinlæring er det flere forskjellige metoder for å lære opp modellene, som blant annet veiledet læring, semi-veiledet læring og u-veiledet læring (IBM, u.å. b).

3.1.3 Dyplæring

Dyplæring (Deep Learning) er en undergruppe av maskinlæring som bruker flerlags nevrale nettverk. Dette kalles dype nevrale nettverk, som brukes for å simulere den komplekse beslutningsevnen til den menneskelige hjernen.

Et dypt nevralt nettverk (DNN) er et nevralt nettverk med tre eller flere lag, men i praksis har de ofte mange flere lag. Et DNN er trent på store mengder data for å identifisere og klassifisere fenomener, gjenkjenne mønstre og sammenhenger, vurdere muligheter og lage prediksjoner samt ta beslutninger. At dype nevrale nettverk har flere lag, hjelper med å optimalisere resultatene for større nøyaktighet. Et lag i et nevralt nettverk referere til en samling noder på en gitt dybde i det nevrale nettverket. Lagene består av tre hoveddeler, som er inputlag, skjult lag og outputlag. Inputlaget er det laget der data blir matet inn som skal trenes eller predikeres på, der hver node representerer en egenskap i dataen (IBM, u.å. c).

Et nevralt nettverk kan ha ett eller flere skjulte lag. Det skjulte laget er hvor «magien» i nevrale nettverk skjer, her prøver hvert lag av modellen å lære om forskjellige aspekter med dataen den skal prosessere ved å gjøre komplekse utregninger på inputdataen. Output laget produserer det endelige resultatet, eller prediksjonen, som er gjort av modellen (Sarita, 2023).

Konvolusjonelle nevrale nettverk (CNN) er en type DNN ofte brukt for bildeanalyse, og er spesielt effektiv på bildeklassifisering og objekt gjenkjenning (Pragya, 2024).

Dyplæring eliminerer deler av databehandlingen som normalt er involvert i ML, da algoritmene som brukes her kan være mer robuste. Dermed kan dyplæringsmodellene ta inn ustrukturert data som bilder, og automatisere utvinningen av viktig data. Dette fjerner avhengigheten av menneskelige input for å komme frem til egenskaper som bestemmer forskjellen mellom for eksempel “katt” eller “hund” på et bilde, men heller lar dyplæringsalgoritmer bestemme egenskapene som er viktigst for å skille hvert dyr fra et annet. Dyplæring skiller seg fra maskinlæring på hvilken type data den jobber med og med hvilke metoder den lærer. Maskinlæring bruker ofte strukturert og markert data for å gjøre prediksjoner, mens dyplæring eliminerer mye av behovet for å pre-prosessere data og kan gjøre prediksjoner på ustrukturert data som eksempelvis tekst og bilder. Dette gjør at dyplæring kan kreve enorme mengder datakraft, og kan være veldig energikrevende (IBM, u.å. d).

3.1.4 Maskinlæring innen forskning

Maskinlæring har blitt et mer og mer verdifullt verktøy, spesielt innen forskning. Maskinlæring har åpnet for helt nye muligheter innen dataanalyse og hvordan en tolker komplekse og store datasett (Salomão, 2024). Det blir brukt maskinlæring innen forskning, blant annet for å analysere proteinstrukturer (Audagnotto, *et al,* 2022). Det blir også brukt maskinlæring til å klassifisere astronomiske data der modellen har blitt trent opp på datasett bestående av millioner av objekter (Sabat'es, 2022). Maskinlæring blir også brukt for å predikere klimamønstre (Gibson, *et al,* 2021). Forskere kan trene maskinlæringsmodeller til å finne skjulte mønstre, gjøre prediksjoner og for å få en dypere forståelse for komplekse fenomener ved å bruke store datasett (Salomão, 2024).

3.1.5 Hvordan har maskinlæring forbedret bildeanalyse

Maskinlæring (ML), har blitt tatt i bruk av bedrifter og vitenskapelige eksperimenter for å revolusjonere måten vi analyserer visuell data. Maskinlæring har blant annet gjort det mulig å automatisk trekke ut informasjon fra bilder, identifisere objekter, se trender og forbedre kvaliteten på bilder.

Kunstig syn (computer vision) er en gren av kunstig intelligens som gjør det mulig for datamaskiner å tolke og forstå den visuelle verden. Maskiner bruker teknikker som maskinlæring og nevrale nettverk til å behandle bilder og videoer, noe som gjør dem i stand til å identifisere objekter, ta beslutninger og forutsi utfall basert på visuelle inndata (IBM, u.å. e). Det brukes kunstig syn for å analysere data fra digitale bilder for å se etter mønstre i dataen, blant annet for å kunne klassifisere bilder. Som andre maskinlæringssystemer, trenger kunstig syn store mengder data for å trenes.

En kombinasjon av disse teknologien har gjort det lettere å forstå visuelle data og muliggjør en langt mer effektiv analyse og forståelse for dataen. Dette har direkte påvirket forbedringen av arbeidsmetoder og økt produktiviteten innenfor visuell analyse (IITK, 2023).

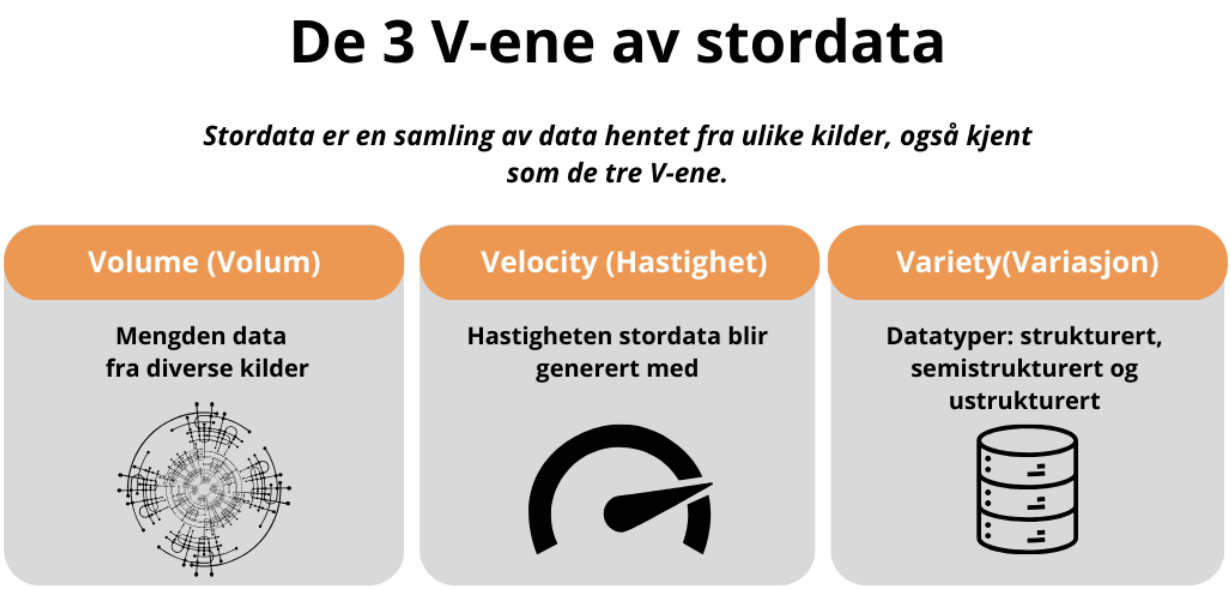
3.1.6 Hvorfor er det viktig med nøyaktig bildeanalyse i konteksten av CO2-lagring

Nøyaktig bildeanalyse kan hjelpe med å evaluere integriteten til de geologiske barrierene som forhindrer CO2 fra å lekke ut fra lagringsformasjonene. Dette kan være essensielt for å sikre langtids lagring av CO2 uten komplikasjoner, og forhindre skade på nærliggende økosystemer (Zappone, *et al,* 2021). Moderne teknikker for bildeanalyse som benytter seg av nevrale nettverk, bidrar til en dypere forståelse av hvordan CO2 oppfører seg i geologiske formasjoner og hvordan

den beveger seg over tid (Alqahtani, *et al*, 2023). Dette er avgjørende for å optimalisere injeksjonen av CO2 i reservoarer, samt bruken av de gamle oljereservoarene. Dette hjelper også med å utvikle overvåkningsstrategier som er nødvendig for å håndtere og dempe risikoen gjennom hele prosjektets levetid (Zappone, *et al*, 2021). Dermed kan nøyaktig bildeanalyse være avgjørende for å sikre at lagringsprosessen er så effektiv som mulig, og at det lagrede karbondioksidet forblir i de planlagte geologiske formasjonene da det kan være skadelig for nærliggende økosystemer dersom den lekker ut av reservoarene.

3.1.7 Utfordringene ved bildeanalyse av ustrukturert data

Ustrukturert data passer typisk ikke inn i tradisjonelle pre-definerte datamodeller, og dermed blir KI og ML brukt for å tilby solide og skalerbare løsninger for å håndtere og trekke ut viktig informasjon fra denne typen data (CIOpages, 2023).



Figur 3-2 - Illustrasjon av de tre v'ene i store data (Lutkevich, u.å.)

Tre av de største utfordringene med ustrukturert data er: volum, variasjon, hastighet - også kjent som "The 3 V's of big data". Figur 3-2 illustrerer de 3 v'ene i store data.

1. **Volum:** Store mengder ustrukturert data skaper utfordringer innen lagring, prosessering og analyse.

2. **Variasjon:** Ustrukturert data kommer i mange variasjoner, dette kan eksempelvis være bilder, video eller tekst. Hver av disse trenger sine egne metoder for utvinning, lagring og analysering som skaper et ekstra lag av kompleksitet.
3. **Hastighet:** IBM estimerer at 90% av all data tilgjengelig i dag, har blitt generert de to siste årene. Hastigheten av datagenerering skaper utfordringer for sanntidsanalyser og innsikter, som kan være utdaterte før de kan bli brukt (CIOpages, 2023).

Med utgangspunkt i bilder, kan det være store variasjoner på kvalitet ut ifra hvordan bildet er tatt, hvilken oppløsning, gjenskinn, belysning, skyggelegging og andre forstyrelser. Dette kan skape utfordringer i utvinningen av dataegenskaper, og i prosessen med å klassifisere bildene. Innenfor dyplæring læres sammenhenger direkte fra dataen, dermed er dyplæringsmodeller som klassifiserer bilder direkte påvirket av bildekvaliteten (Nazare, 2018).

En annen utfordring i håndtering av ustrukturert data er å kunne forstå hva dataen betyr, analysere den og produsere brukbare innsikter basert på dataen (CIOpages, 2023).

3.2 Grunnleggende om CO2-lagring

CO2-lagring omhandler det å lagre CO2 fra for eksempel industriutslipp eller kraftproduksjon, slik at det ikke slipper ut i atmosfæren (Sokkeldirektoratet, u.å.).

CO2 kan bli lagret under bakken som en superkritisk væske, som vil si et stoff med temperatur og trykk over henholdsvis kritisk temperatur og trykk, der trykket ikke er stort nok til å komprimere det til fast stoff (Grøn, 2022). Karbonlagring utnytter egenskapene til CO2, ved å lagre den under bakken der forholdene tillater karbondioksidet å bli superkritisk. CO2 blir superkritisk dersom den får en temperatur høyere enn 31.1 celsius og et trykk som er høyere enn 72.9 atmosfærisk trykk (atm), som tilsvarer rundt 1057 trykk i pund per kvadrattomme (psi). Hovedfordelen med å lagre CO2 i en superkritisk tilstand, er at nødvendig lagringsområdet er vesentlig mindre enn om den hadde vært lagret under "vanlige" forhold (National Energy Technology Laboratory, u.å). I forskningsprosjektet er det ingen superkritiske forhold som påvirker CO2-lagring.

Temperaturen og trykket i jordens kjerne øker naturlig desto dypere man går, og på dybder fra 800 meter og nedover er trykket og temperaturen som regel over de kritiske grensene til CO2, som gjør at den vil holde seg som en superkritisk væske.

FNs klimapanel (IPCC) og Det internasjonale energibyrået (IEA) anslår at andelen av utslippsreduksjoner som må gjøres ved hjelp av karbonlagring ligger på mellom 12 til 20 prosent

(NHO, u.å.). Karbonlagring er dermed et viktig hjelpemiddel for å bekjempe klimaendringene, samt nå FNs klimamål i henhold til Paris avtalen (FN, u.å.).

3.3 Maskinlæringens rolle i CO2-lagring

Det har blitt gjennomført flere forskninger der det er brukt maskinlæring i kombinasjon med karbonlagring, eksempelvis forskningen av Hanlin Sheng, Xinming Wu, Xiaoming Sun, og Long Wuch(2023), "Deep Learning for å karakterisere CO2 migrasjon i time-lapse seismiske bilder" og Harnessing the power of machine learning for carbon capture, utilisation, and storage (CCUS) – a state-of-the-art review (Yan, *et al,* 2021).

I forskningsprosjektet «Harnessing the power of machine learning for carbon capture, utilisation, and storage (CCUS) – a state-of-the-art review», blir det forsket på muligheten til å bruke maskinlæring (ML) for å analysere, og velge, best mulige geologiske formasjoner for CO2-lagring. Dette inkluderer evaluering av porøsitet, permeabilitet og geomekaniske egenskaper av geologiske formasjoner for å estimere evnen den har til å lagre CO2 på en effektiv og sikker måte. Porøsitet sier noe om hvor store hullene er, i forhold til det faste materialet. Permeabilitet er et mål på hvor lett væske flyter igjennom det porøse materialet, og geomekaniske egenskaper referere til de elastiske egenskapene i materialet. ML kan også bli brukt i etterkant av lagring for å hjelpe med å predikere mulige risikoer, som lekkasje eller ustabile formasjoner.

En viktig årsak til at ML er brukt i denne prosessen, er å konstruere input-output relasjoner når informasjon mangler eller der den grunnleggende teorien er uklar. Typisk data som har blitt brukt i tidligere forskninger er seismisk, brønn målinger med hjelp av trykk eller den totale konsentrasjonen av oppløste stoffer i en væske (TDS), porøsitet, permeabilitetskart, og injeksjons-/produksjonshastighet.

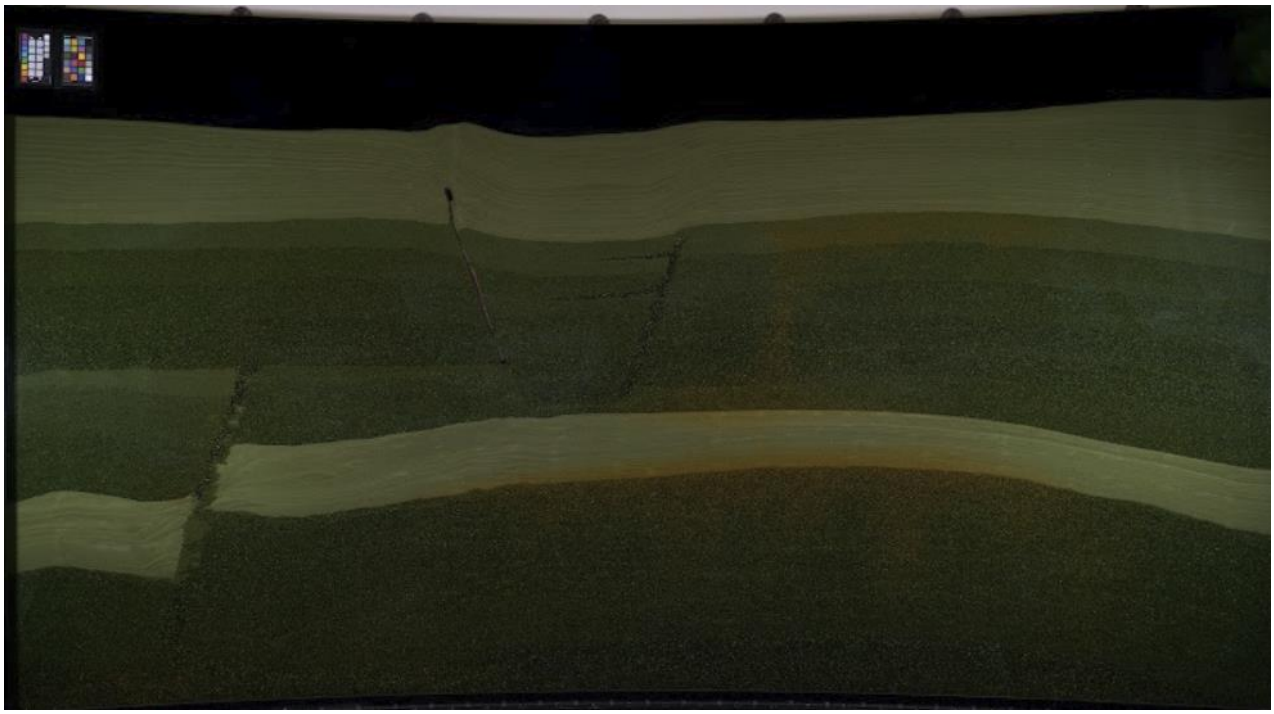
Det er fortsatt en forventning om en bedre og mer universell modell som kan håndtere og effektivisere hele prosessen i et karbonlagringsprosjekt (Yan, *et al.,* 2021).

3.4 Datasett

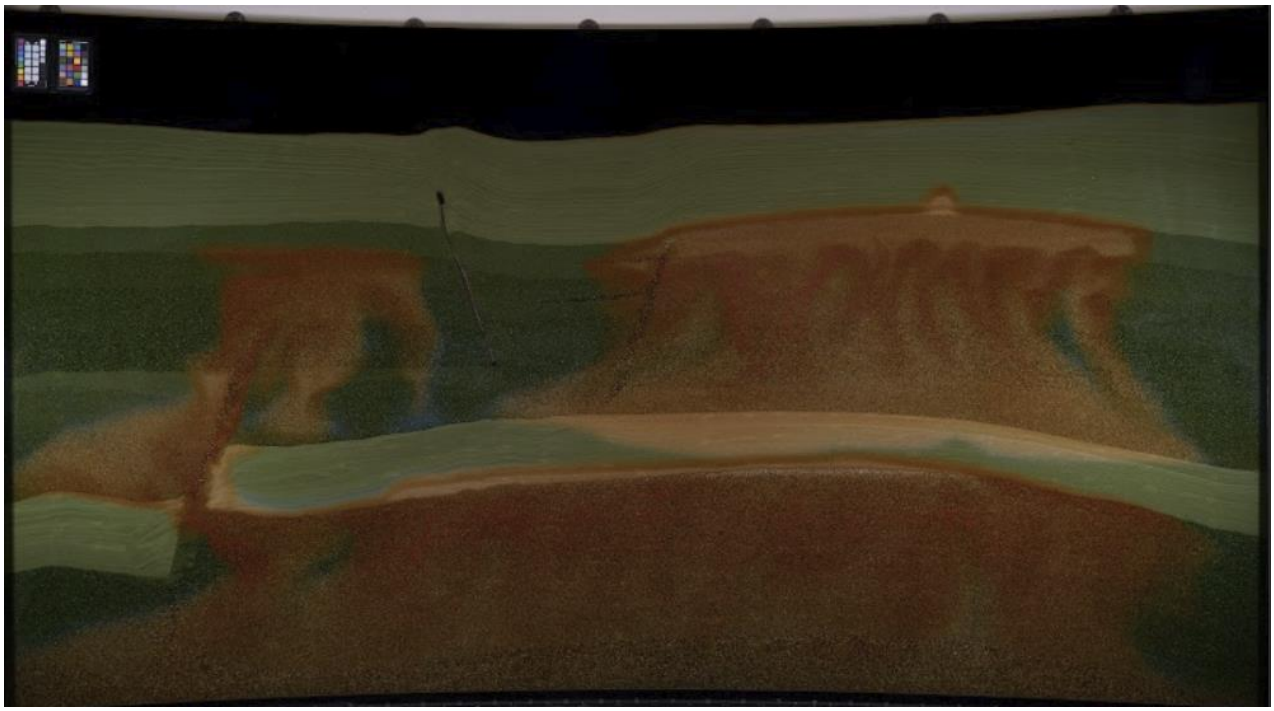
Som en del av oppgaven ble det utdelt et datasett bestående av bilder tatt av FluidFlower-tanken til UiB, med en oppløsning på 7952 x 4472. FluidFlower er et visuelt hjelpemiddel for å illustrere hvordan CO2 oppfører seg i omgivelser som simulerer geologiske formasjoner (UiB, u.å.).

Det blir gjort opptak av CO2-injeksjoner med hjelp av timelapse bildeserie. I forkant av prosjektet er det blitt gjort fem repetisjoner av eksperimenter som er identisk utført. For hvert av de fem repetisjonene, som kalles C1, C2, C3, C4 og C5, er det skapt et datasett. Hvert datasett inneholder 137 høyoppløste bilder med følgende intervaller: 10 bilder før CO2 injeksjon, med 20 sekunder intervaller; bilder hvert 5 minutt de første 360 minuttene (6 timer) av eksperimentet (73 bilder); bilder hver time til det er gått 48 timer (42 bilder); bilder hver sjette time til eksperimentet er ferdig (12 bilder) (Fernø, *et al.*, 2024).

For å visualisere hvordan CO2 oppfører seg i FluidFlower-tanken er det nødvendig å representere CO2 ved hjelp av farger. Det blir brukt multikromatiske sporstoffer for å oppnå dette, ved at sporstoffene endrer farge basert på pH-verdien i vannet, og CO2 gjør at vannet blir surere. Hvilke farger som blir valgt er en pågående forskning, så fargene kan endre seg flere ganger i fremtiden. I Figur 3-3 og Figur 3-4 nedenfor ser vi eksempler fra tildelt datasett.



Figur 3-3 - Første bilde fra c4 mappen, start av syklus. Rød/grønn bilde



Figur 3-4 - Siste bilde fra c4 mappen, slutt av syklus. Rød/grønn bilde

4. Prosjektdesign

I dette kapittelet blir det gått igjennom forskjellige mulige løsninger på prosjektet, hvilke tilnærminger som er valgt og hvorfor de er valgt. Videre vil det bli snakket om hvilke ressurser som brukes til valgt tilnærming, samt hvordan prosjektet er organisert og planlagt. Til slutt blir det gått igjennom risikoer og hvordan prosjektet som en helhet skal evalueres.

4.1 Forslag til løsning

Her presenteres de alternative løsningene som vurderes brukt i prosjektet. De ulike alternativene diskuteres samt det presenteres dataaugmenteringene som brukes på datasettet.

4.1.1 Alternativ løsning 1: Ingen maskinlæring

I alternativ løsning 1 lages det en algoritme for å detektere CO2 uten bruk av maskinlæring. Da oppdragsgiver ikke stiller krav til at det nødvendigvis må brukes maskinlæring, men at ønsket resultat skal kunne returnere et bilde der CO2 er markert og representert som en binær verdi. Dermed vil denne tilnærmingen kunne oppfylle krav stilt til oppgaven. Fordelen med en slik løsning er at den kan kreve vesentlig mindre datakraft enn en maskinlæringsmodell, samt at gruppen har mer erfaring innen algoritmeimplementering, enn maskinlæring. Algoritmen vil basere seg på farger, der det settes grenseverdier basert på Rød-Grønn-Blå (RGB) verdiene i bildene. Alle verdier som treffer mellom de to grenseverdiene blir konvertert til hvit, og svart om ikke, som vil produsere en binær representasjon av bildet.

Da en fungerende løsning må ta hensyn til at fargeverdiene kan endre seg, må det også implementeres en løsning for å enkelt kunne endre grenseverdiene basert på fargevalgene i bildene. I bildene er det flere steder der det er hull, og dersom hullene er mindre enn en gitt størrelse må de fylles igjen. Her vil det bli brukt morfologiske operasjoner for å løse dette, som innebærer å bruke dilasjon og erosjon for å fylle hullene (Chhikara, 2022).

4.1.2 Alternativ løsning 2: PyTorch

I alternativ løsning 2 blir det sett på muligheten for å bruke PyTorch, som er et rammeverk for å bygge dyplæringsmodeller. Dyplæringsmodeller brukes blant annet i applikasjoner for bildegjenkjenning og tekstprosessering. PyTorch egner seg til å trene dyplæringsmodeller med måten den utnytter datakraft og parallellisering, med at en kan splitte trening av modellen til flere

GPU- og CPU kjerner. Det var også naturlig å vurdere PyTorch da det er et rammeverk som er bygget med Python (Nvidia, u.å).

PyTorch tilbyr også store modifiseringsmuligheter for modellen, som gjør at alle parametere kan tilpasses etter behov. Dersom en har inngående kunnskap innen dyplæringsmodeller og rammeverk, vil PyTorch for mange være det beste valget grunnet mulighetene til å finjustere modellen etter behov, samt tilgang på store og omfattende biblioteker.

4.1.2 Alternativ løsning 3: Fast.ai

I alternativ løsning 3 brukes et dyplæringsbibliotek som heter Fast.ai for å trene modellene som skal gjenkjenne CO2. Fast.ai har som mål å gjøre dyplæring mer tilgjengelig for allmennheten, med å abstraktere mye av kompleksiteten til et tradisjonelt dyplæringsrammeverk som eksempelvis PyTorch. Den fjerner også nødvendigheten for å ha en inngående kunnskap innenfor matte og koding for å kunne bruke dyplæringsmodeller. Fast.ai tilbyr et høynivå brukergrensesnitt som bygger på dyplæringsrammeverket PyTorch (Howard, 2017).

Fast.ai støtter flere forskjellige typer CNN modeller, blant annet: ResNet, VGGNet16_bn, AlexNet, DenseNet og SqueezeNet (Fast.ai, u.å. a). I dette prosjektet er det mulig å bruke samtlige av de fem nevnte modellene, da de alle kan brukes med fast.ai og vil være passende for bilde segmentering. Alle modellene vil bli trent på samme datasett, slik at de kan sammenlignes direkte opp mot hverandre.

Fordelen med denne løsningen er at den krever mindre forkunnskap om dyplæring, som kan gjøre denne løsningen mer tidsbesparende, samt det kan være mulig å få bedre resultater enn med et annet rammeverk som krever mer manuelt oppsett.

4.1.3 Augmentering av datasett i valgt løsning

I alternativ løsning 2 og 3 er det nødvendig å vurdere hva slags teknikker en bruker for å kunne trene opp en mer treffsikker modell. Dataaugmentering er en mye brukt teknikk innen maskinlæring og dyplæring (Awan, 2022). Dette gjøres ved å utføre ulike transformasjoner på treningssettet slik at modellen trenes opp på et potensielt bedre, og mer variert datasett, samtidig som at dette øker størrelsen på datasettet. Maskene som skal brukes til datasettet blir tegnet for hånd, og da dette er tidskrevende arbeid kan augmenteringer av datasettet være viktig for å få mest mulig treningsdata.

Det finnes flere forskjellige måter å augmentere data på, men her er det valgt ut to typer augmenteringer som blir ansett som passende til dette prosjektet.

Geometriske transformasjoner:
Geometriske transformasjoner vil for eksempel være rotering, beskjæring eller forskyving av bildene.

Det er ikke alle geometriske transformasjoner som er like relevante for dette prosjektet, da det er ønsket å bevare mest mulig av detaljene i bildene. Eksempelvis så vil det ikke være hensiktsmessig å rotere, forvrenge eller forskyve bildet da bildene i datasettet kommer fra et kontrollert laboratoriums miljø og vil alltid ha samme kvalitet. Da ønsket er at modellen skal gjenkjenne den overordnet strukturen i bildet, vil det være imot sin hensikt å rotere bildet vertikalt, da CO2 i tanken alltid vil renne nedover.
Av de geometriske transformasjonene er det kun å vende bildene som er ansett som hensiktsmessig, da dette ikke endrer den overordnete strukturen og vil utvide datasettet med reelle transformasjoner. Transformasjonen vil bli gjort på noen tilfeldige bilder fra datasettet, som deretter blir en del av treningsdataen til modellen.

Videre må bildene konverteres til et mindre format slik at de kan brukes av en maskinlæringsmodell, da de originale bildene er for store til å kunne brukes til en maskinlæringsmodell direkte. Her må en endre størrelsene på bildene, enten ved å krympe bildene eller å splitte disse opp i mindre segmenter. Å krympe bildene vil ikke være hensiktsmessig da det er ønsket å beholde mest mulig detaljer av det ordinære bildet. Derfor blir det heller valgt å splitte opp bildene i mindre segmenter.

Fargetransformasjoner:

Innen bildeklassifisering og segmentering handler fargetransformasjoner om å justere eller endre på fargene i bildene.

I utdelt datasett er det kun én type fargekombinasjon, der CO2 er farget rød og resten i forskjellige grønn/brun nyanser. Dermed er det liten fargevariasjon i datasettet, som kan gjøre at modellen lærer seg å kjenne igjen fargene i bildene, og ikke den overordnete strukturen. En konsekvens av dette kan være at modellen ikke vil lære seg å generalisere godt nok til nye farger i bildene. Da det i fremtiden vil komme nye farger i FluidFlower-tanken, er det viktig at modellen kan generalisere

seg til å håndtere nye farger. Dermed blir fargeaugmentering ansett som en av de viktigere augmentasjonene for dette datasettet.

En mulig løsning for å skape variasjon i datasettet, er ved å bytte fargekanaler på bildene. Det vil si at en eksempelvis bytter et RGB-bilde til å være Blå-Grønn-Rød (BGR)-bilde. Resultatet er at en får et bilde med helt andre farger, uten å endre den grunnleggende strukturen i bildet. Disse bildene vil bli lagt til i det eksisterende datasettet, slik at en får et større og mer variert datasett.

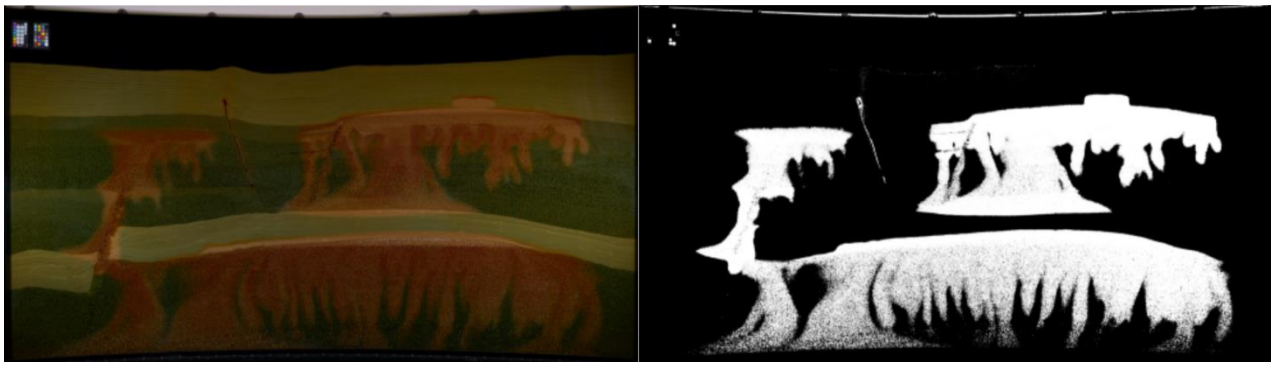
Videre utføres det endringer på lysstyrke, metning, kontrast og fargetonene, i tilfeldig grad, på alle bildene.

4.1.3 Diskusjon av alternativene

Det er presentert 3 alternative løsninger, som alle har sine styrker og svakheter. De vil variere i kompleksitets- og tidsnivå det tar å gjennomføre de. Her diskuteres de 3 alternativene, og hvilke fordeler og ulemper de ulike løsningene har.

Det ble tidlig bestemt å teste ut mulighetene for å ikke bruke dyplæring, men heller bruke en algoritme som ikke er basert på maskinlæring. I alternativ løsning 1 lages en algoritme for å gjenkjenne RGB-verdier basert på en grenseverdi.

I form av tidsbruk, er dette vurdert til å være den mest effektive løsningen for dette prosjektet, da det ikke er behov for å trene en maskinlæringsmodell. I første iterasjon av denne løsningen, ble markeringen av CO2 relativt god, men uten å fylle igjen hull i bildet. For å rette på dette ble det laget en morfologisk operasjon, som i praksis fylte alle hull som var under en forhåndsdefinert størrelse. Det ble tidlig vurdert at en denne løsningen ikke ville være tilstrekkelig for å kunne tilfredsstille kravene stilt til ønsket resultat. Denne vurderingen ble gjort grunnet for mange forskjellige variabler som kunne oppstå i et bilde, og det vil være for komplekst for gruppen å ta høyde for alle variabler som kan endre seg i bildene manuelt, med en slik algoritme. En annen grunn er at fargene i FluidFlower-tanken endrer seg jevnlig, og det ble opplevd som for komplekst å kunne ta høyde for dette. Det positive resultatet fra denne løsningen var at bildene den genererte, med noen modifiseringer, kunne brukes som “masker” til å videre trene maskinlæringsmodeller. Figur 4-1 viser resultat produsert under testing av alternativ løsning 1.



Figur 4-1 - Original bilde i rød/grønn og resultat produsert under testing av alternativ løsning 1

Alternativ løsning 2 og 3 har en ganske lik tilnærming, da begge bruker PyTorch i en eller annen form. Forskjellen er i hovedsak hvilken grad kompleksiteten til modellen er abstrahert. I alternativ løsning 3 blir det brukt fast.ai sammen med flere forskjellige CNN modeller. Her testes alle modellene på samme datasett, med de samme augmenteringene. I dyplæringsfag som gjøres ved siden av bachelor, er mye av pensum basert på fast.ai som gjør at det kan være et naturlig valg for oppgaven. Samtidig er fast.ai en mindre kompleks løsning, som gjør at kravene til forkunnskap ikke er like store som med PyTorch. Det kan samtidig være lettere å teste forskjellige modeller på kortere tid. Til gjengjeld vil PyTorch tilby flere muligheter for finjustering av modellen, samtidig som at nye modeller vil bli tidligere tilgjengelig for PyTorch enn for fast.ai (Nvidia, u.å). Det kan være naturlig å begynne med prototyping og testing av modell i fast.ai, før man eventuelt kan gå videre til PyTorch om det trengs mer kontroll og finjustering av modellen (Howard, 2017).

4.2 Valgt løsning

Det ble vurdert at det er for komplekst for gruppen å sette seg inn i PyTorch for å få et resultat som er bedre enn det som er mulig med fast.ai, da også grunnet tidsbegrensninger. Samtidig er det en kjent løsning å lage prototyper med fast.ai, før en eventuelt går videre til PyTorch. Det blir derimot ikke brukt PyTorch i dette prosjektet, men det kan være aktuelt for videre arbeid som eventuelt gjøres i etterkant av prosjektet. Løsningen med å lage algoritmer ble opplevd som for mangelfull til å oppfylle kravene for prosjektet. Fast.ai er dermed den løsningen som blir valgt.

4.3 Valg av verktøy

Python

Python er et objektorientert og imperativt høy-level programmeringsspråk som ofte blir brukt til dataanalyse, maskinlæring og automatisering (Dvergsdal, u.å). C++ og Python er begge språk som kan brukes til maskinlæring, men da Python er det språket gruppen har mest erfaring med samt det er det veileder anbefalte, valgte vi Python, versjon 3.12 til dette prosjektet.

Kaggle

Kaggle er en nettside der man kan konkurrere med andre innen maskinlæring, og løse oppgaver gitt av både privatpersoner og bedrifter. Kaggle kan også brukes for læring, samarbeid, og sånn som prosjektgruppen har brukt det, for interaktive notatbøker som kan kjøre kode (Staff, 2023).

U-Net

U-Net er et nevralt nettverk som ble laget i 2015 for segmentering av medisinske bilder, men som i dag hovedsakelig blir brukt til semantisk segmentering av alle typer bilder, dermed var det dette nettverket prosjektgruppen landet på i utgangspunktet (Datascientest, 2023).

Labelstudio

Labelstudio er et open-source verktøy som brukes til å markere bildedata, finjustere og validere maskinlæring modeller. Dette verktøyet brukes til å lage masker som modellen bruker som fasit under trening (LabelStudio, u.å.).

GoodDay

GoodDay er en planleggingsplattform som har en samling med verktøy for å både planlegge og håndtere prosjekter. Under prosjektet brukes den til timeregistrering, Gantt-diagram, og som et kanban verktøy (GoodDay, u.å.). Da de fleste i gruppen har gode erfaringer med dette verktøyet, ble det valgt som planleggings plattform for bachelorprosjektet

Notion

Notion er et alt i ett planleggingsverktøy som gruppen bruker til å lage status- og ukesrapporter. Notion ble valgt grunnet gruppen har erfaring med dette verktøyet fra tidligere prosjekt (Notion, u.å.).

Fast.ai

Fast.ai er et maskinlæringsbibliotek som tilbyr brukeren høyt nivå komponenter som kan raskt og enkelt gi gode resultater i standard dyplæringsdomener, dermed ble fast.ai valgt for å enkelt få tilgang på lavnivå komponenter som kan kombineres for å lage nye tilnærminger, redusere kompleksiteten og redusere tiden for prototyping (fast.ai, u.å. b).

Torchvision

Torchvision er et bibliotek i Python som tilbyr en rekke verktøy for bildetransformasjoner. Disse vil være nødvendige for å kunne utføre de nødvendige augmenteringene av bildene. Torchvision hører i tillegg til PyTorch, som er rammeverket Fast.ai er bygget på. Dermed vil det være et naturlig valg for dette prosjektet (Torchvision, u.å).

Matplotlib:

Et bibliotek som brukes til å vise grafer. Det er ideelt for å visualisere data på en forståelig måte (Matplotlib, u.å).

Numpy:

Står for Numerical Python, og er et bibliotek som gir støtte for store, flerdimensjonale tabeller og matriser, sammen med et stort bibliotek av høynivå matematiske funksjoner (W3Schools, u.å. a)

Pandas:

Et datahåndteringsbibliotek optimalisert for å arbeide med datasett. Brukes for å analysere, utforske og manipulere data (W3Schools, u.å. b).

PIL (Pillow):

Python Imaging Library (PIL), nå kjent som Pillow. Det er et bibliotek for å åpne, manipulere og lagre forskjellige bildeformater i Python (MargCampusSoft, 2023).

OpenCV:

Står for Open Source Computer Vision Library, et bibliotek for computervisjon, maskinlæringsprogramvare og bildeprosessering. (GeeksForGeeks, 2024)

Valgte CNN modeller:

Resnet50:

ResNet50 er en modell med 50 lag, som ofte brukes til bildegjenkjenning og bildesegmentering. Modellen lærer ved å stable såkalte residualblokker (residual blocks). Disse blir brukt som "snarveier" for å hoppe over lag, noe som gjør det mulig å legge til flere konvolusjonslag uten tap av hastighet og ytelse. Dette er den største og mest komplekse av modellene valgt i dette prosjektet. Det er også en av de dypere modellene som blir brukt (Datagen, u.å).

SqueezeNet1_1:
En mer kompakt modell designet for å ha færre parametre uten å være mindre nøyaktig. Laget for bruk på mindre enheter med begrenset minne og prosesseringskraft.

Bruker et unikt oppsett kalt "fire modules", en spesiell type av konvolusjonslag som reduserer antall parameter (Malhotra, *et al*, 2023).

VGG16_bn:
En 16 lags modell primært brukt til bildebehandlingsoppgaver på grunn av dens enkelhet og effektivitet. Batch-normalisering hjelper med forbedre treningen av modellen(Simonyan, Zisserman, 2015).

Alexnet
En av de tidligste dype konvolusjonsnettverken med 8 lag, inkludert 5 konvolusjonslag og 3 fullt tilkoblede lag.

(Mahotra, *et al*, 2023)

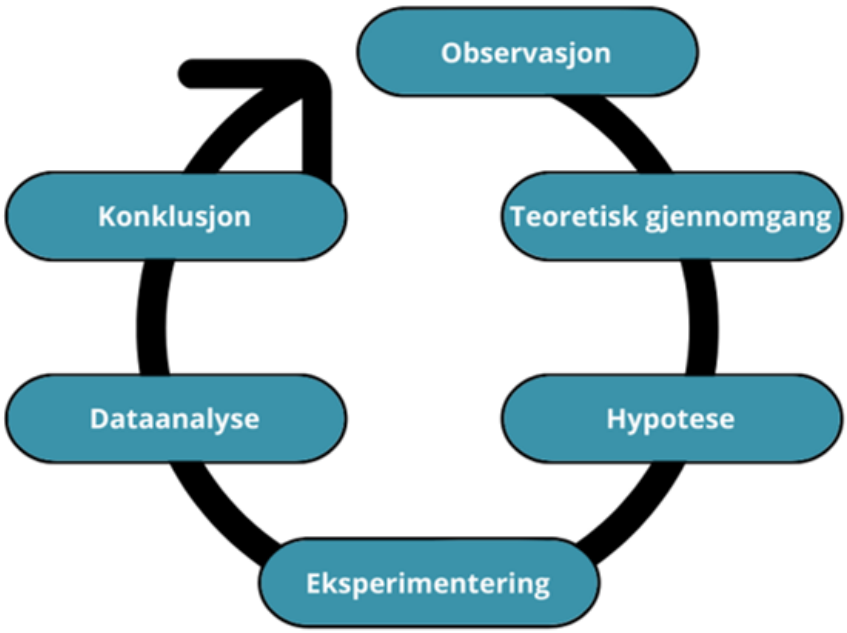
4.4 Prosjektmetodikk

Dette delkapittelet går igjennom utviklingsmetodikk som er brukt i prosjektet, samt prosjektplan og risikovurdering.

4.4.1 Utviklingsmetodikk

For å sikre effektiv prosjektutvikling, var det nødvending å anvende metoder som har god tilpasningsevne, struktur og nøyaktig evaluering. Videre vil bruken av vitenskapelig metode, Cross-industry standard process for data mining (CRISP-DM) og smidig utvikling bli diskutert nærmere, for å gi en oversikt over utviklingsprosessen i prosjektet.

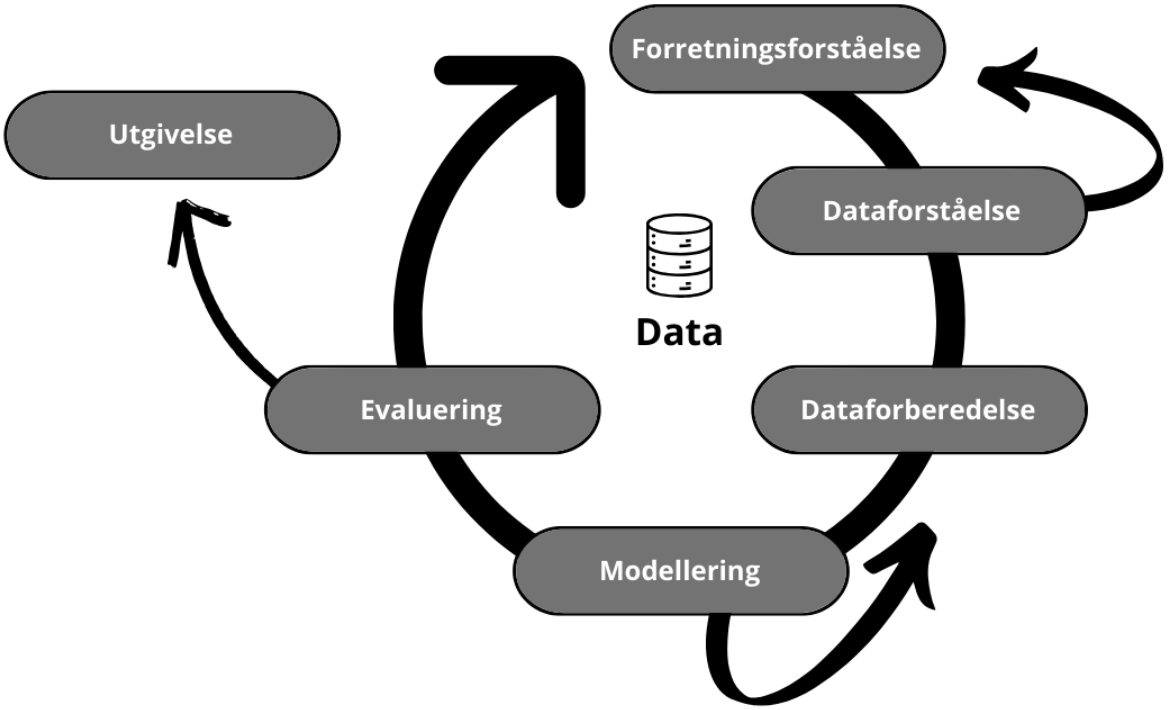
Vitenskapelig metode er et sett med retningslinjer for hvordan man skal gå frem for å undersøke en problemstilling eller et spørsmål på en vitenskapelig måte. Metodikken legger vekt på systematiske metoder for å frembringe ny kunnskap som er så objektiv som mulig (Rørvik, 2013). Den vitenskapelige metoden inneholder observasjon, der spørsmål og nøkkelproblemer identifiseres. Det utvikles en testbar hypotese som tilbyr en mulig forklaring på problemet. Eksperimentering og testing av hypotesen skaper deretter grunnlag for en konklusjon om hypotesen er støttet eller motbevist.



Figur 4-2 - Figuren viser stegene i vitenskapelig metode

Prosjektet følger stegene som er involvert i vitenskapelig metode, se Figur 4.2 for en illustrasjon av de forskjellige stegene i vitenskapelig metode. Fra starten har oppgavebeskrivelsen blitt studert for å få en god forståelse av problemstillingen, deretter er det undersøkt forskninger og litteratur relatert til prosjektet. Oppgavebeskrivelsen og tidligere forskninger har skapt grunnlaget for å formulere problemet, samt hvilke metoder som skal brukes. Hypotesene står for fremgangsmåtene for å løse prosjektets problemstilling, og utførelsen av hypotesen står for eksperimenteringen. Resultatene fra maskinlæringsmodellen blir analysert for hver iterasjon, før versjonen som treffer best i henhold til målene blir nådd. Ved å anvende en standardisert tilnærming som vitenskapelig metode, sikrer vi at forskningsresultatene er objektive, reproduserbare og grundig undersøkt.

CRISP-DM (Cross-Industry Standard Process for Data Mining) er en prosessmodell som gir en systematisk tilnærming til datavitenskapelige prosjekter. Modellen inkluderer seks faser: Forretningsforståelse, dataforståelse, datatilberedelse, modellering, evaluering og distribusjon (Hotz, 2024). Siden den strukturerte rammen til CRISP-DM passer med de systematiske aspektene til vitenskapelig metode, og inneholder viktige steg for prosjektet. Er det naturlig å bruke CRISP-DM, som en utvidelse av den eksperimentelle delen av vitenskapelig metode for en mer omfattende utvikling. Se Figur 4-3 for en illustrasjon av CRISP-DM.



Figur 4-3 - Figuren viser stegene i CRISP-DM

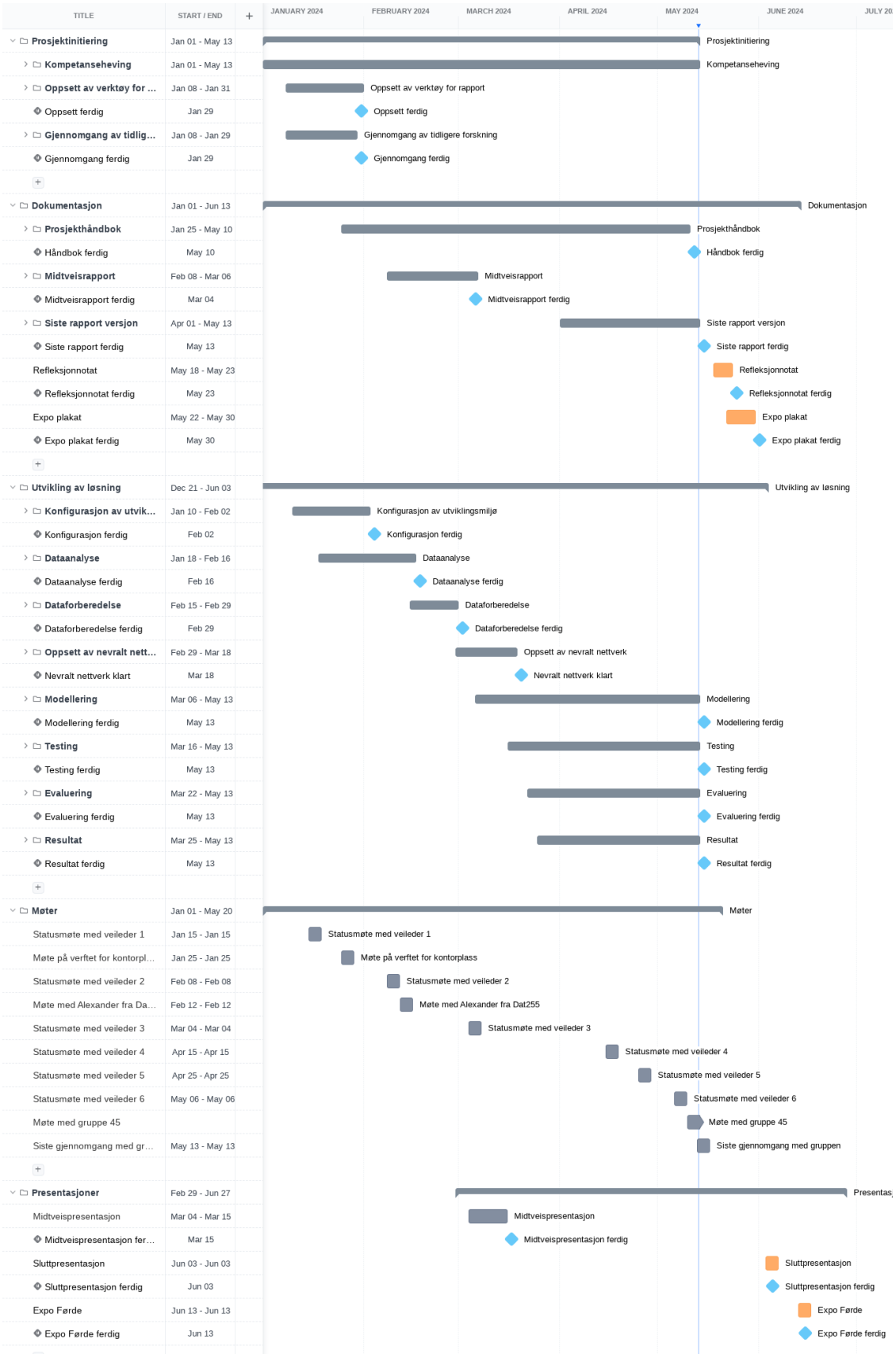
Smidig utvikling muliggjør rask testing og validering av hypoteser gjennom gjentatte iterasjoner av utviklingsprosessen (Szalvay, 2004). Dette stemmer med den vitenskapelige metoden, som legger stor vekt på systematiske og empiriske undersøkelser for å fremme kontinuerlig forbedring. Ved å anvende vitenskapelig metode og CRISP-DM for å sikre smidig iterativ utvikling, har vi kunnet evaluere og gjøre endringer til maskinlæringsmodellen kontinuerlig basert på innsikt og oppdagelser gjort underveis.

Denne systematiske og iterative fremgangsmåten har ikke bare ført til læring og tilpasning gjennom prosjektet, men har også forbedret evnen til å håndtere usikkerhet og gjøre nødvendige justeringer raskt.

4.4.2 Prosjektplan

Fremdriftsplanen for prosjektet presenteres gjennom et detaljert GANTT-diagram. Dette diagrammet skisserer den planlagte tidslinjen for prosjektet og gir en inngående gjennomgang av de forskjellige arbeidsoppgavene som skal fullføres, inkludert varigheten for hver oppgave. Arbeidsoppgavene i prosjektet er strukturert i fem sentrale deler: prosjektinitialisering, dokumentasjon, utvikling av løsning, møter og presentasjoner. Status på oppgavene blir representer med fargekoder; grå for ferdig, og oransje for ikke ferdig.

Prosjektet har 18 milepæler, som er representert med diamanter i Figur 4-4, og markerer viktige hendelser i prosjektet. Milepælene inkluderer konfigurasjon av utviklingsmiljø, klargjøring av data, oppsett av nevralt nettverk, modellering og evaluering av resultater. Oppgavene under «Dokumentasjon» punktet i Figur 4-4 inneholder iterasjoner i form av oppgaver. Se Vedlegg 1 for visualisering av iterasjonene. Figur 4-4 viser GANTT-diagrammet som hører til prosjektet, uten å vise de spesifikke underpunktene. Modellering, testing og evaluering repeteres iterativt i utviklingsfasen.



Figur 4-4 - Figuren viser et GANTT-diagram som viser til fremdrift i prosjektet, "Diamantene" representerer milepæler, grå viser ferdige oppgaver og oransje viser ikke ferdige oppgaver.

4.4.3 Risikovurdering

For å identifisere mulige risikoer ble det gjennomført en risikoanalyse for prosjektet. For hver risiko eller hendelse blir det gitt en vekting for sannsynlighet og konsekvens, som sammen gir et risikoprodukt. Tabell 4.1 viser hvordan Risikoproduktet blir kalkulert ut ifra vektingen og sannsynligheten som er gitt. Tabell 4.2 er en oversikt over de fire hendelsene med høyest risikoprodukt. Se Vedlegg 1 for fullstendig risikoanalyse.

Sannsynlighet	**Svært Høy (5)**	**5**	**10**	**15**	**20**	**25**
	Høy (4)	**4**	**8**	**12**	**16**	**20**
	Middels (3)	**3**	**6**	**9**	**12**	**15**
	Lav (2)	**2**	**4**	**6**	**8**	**10**
	Svært Lav (1)	**1**	**2**	**3**	**4**	**5**
		Svært Lav (1)	**Lav (2)**	**Middels (3)**	**Høy (4)**	**Svært Høy (5)**
	Konsekvens					

Tabell 4-1 - Oversikt over hvordan utregning av risikoprodukt blir gjort

	Hendelse /Risiko	Årsak	Sannsynlighet	Konsekvens	Risiko-produkt	Tiltak
1	Feil i algoritmens gjenkjennelse av CO2	Utilstrekkelig trening av datasett eller feil i maskinlæringsalgoritmen	Middels(3)	Svært høy(5)	15	Øke mengden og variasjonen i treningsdata, og forbedre valideringsprosedyrer.
2	Maskinlæringsmodellen generaliserer ikke godt nok til nye data.	Over-tilpassing på treningsdata.	Middels(3)	Høy(4)	12	Implementere kryssvalidering og øke datasettets diversitet.

3	Dårlig datakvalitet som påvirker modellens nøyaktighet.	Feil under innsamling eller overføring av data.	Middels(3)	Høy(4)	12	Etablere kvalitetsk ontroll for datainnsamling.
4	Manglende ekspertise i teamet	Spesialisert kunnskap om maskinløring eller domenespesifikke krav er ikke tilstrekkelig.	Lav(2)	Høy(4)	8	Opplæring av teamet.

Tabell 4-2 - Utsnitt av risikoanalyse

4.5 Evalueringsplan

Målet med evalueringsplanen er å sikre at prosjektets hovedmål oppnås med et tilstrekkelig resultat. Gjennom ukentlige møter med oppdragsgiver vil en kunne kontinuerlig måle, analysere og forbedre modellens ytelse. Disse møtene gir en mulighet til å motta direkte tilbakemeldinger på framgangen, noe som er avgjørende for å sikre en best mulig løsning.

For å vurdere modellens kvantitative resultater på CO2 prediksjon, blir det brukt trenings- og validerings tap, dice loss og nøyaktighet. Se punkt 6.1 for mer informasjon om evalueringsmetodene. Den endelige vurderingen av modellen vil basere seg på tilbakemeldinger fra oppdragsgiver. Denne prosessen er viktig for å identifisere og korrigere eventuelle feilkilder i modellens tolkning av bildet.

5. Detaljert Løsning

Dette kapittelet har som mål å presentere en detaljert løsning for prosjektet, fra start til slutt. Hvert trinn i prosessen, inkludert problembeskrivelse, innsamling av data, klassifisering av bilder, valg og trening av modell, samt testing og evaluering av modell, vil bli gått gjennom i detalj. Kapittelet skal med andre ord gi en helhetlig beskrivelse av hvordan prosjektet har gått fra problembeskrivelse, til en ferdig løsning.

5.1 Overordnet problem

I forbindelse med det pågående forskningsprosjektet på simulert karbonlagring i FluidFlower modellen, blir det generert tusenvis av bilder. Oppdragsgiver ønsker dermed en måte å markere CO2 forekomsten i bildene effektivt og automatisk, slik at det kan gjøres videre matematiske beregninger på bildene.

5.2 Datasett

Kvaliteten på datasettet vil direkte påvirke evnen til å trene en nøyaktig modell for å identifisere CO2. Derfor er valg av datasett og dataaugmenteringer et viktig steg i prosessen for å sikre høy treffsikkerhet.

5.2.1 Dataforberedelse

Datasettet som ble brukt under prosjektet, stammer fra FluidFlower-modellen til Universitet i Bergen (se detaljer i punkt 3.4). Bildene er produsert i et laboratorium under kontrollerte forhold, noe som sikrer at hvert bilde alltid har høy kvalitet, uten støy eller irrelevant informasjon som kunne hatt negativ innvirkning på læringsprosessen til modellen. Før trening av modellen kan begynne, blir det gjennomført noen viktige steg for å forberede treningssettet.

5.2.2 Valg av bilder

I starten av prosjektet fikk vi utdelt et datasett bestående av bildeserier fra eksperimenter gjennomført med FluidFlower-modellen. Datasettet inneholdt rundt 140 bilder hver. Det ble da gått gjennom hver av disse og valgt ut flere bilder med ulike variasjoner i CO2-distribusjon, noe som kan påvirke modellens evne til å kunne gjenkjenne CO2 under forskjellige forhold.

5.2.3 Tegning og forberedning av masker

Datasettet som ble utdelt i forbindelse med prosjektet besto av umarkert data, som vil si at det ikke var laget noe masker, eller fasit, som viser hva som er korrekt markering av CO2. For å bruke datasettet til å trene en dyplæringsmodell er det nødvendig at modellen har et referansegrunnlag for hva som er riktig, og hva som er feil, i segmenteringsprossesen. For å løse dette ble det tegnet masker for hånd ved hjelp av et verktøy som heter LabelStudio. Disse maskene fungerer som en fasit, som modellen kan sjekke prediksjonen sin opp mot under trening. Dermed er det essensielt for å trene opp en treffsikker modell at disse er korrekte.

Ved å bruke LabelStudio, ble det tegnet 18 masker av utvalgte bilder, for å markere hvor det er CO2. Videre ble maskene ytterligere behandlet av et Python-script *"Noise_Cleanup"* for å fjerne støy i maskene. Dette blir gjort ved å utføre morfologiske operasjoner for å fjerne små og uønskede elementer (Chhikara, 2022). Måten dette fungerer på er at det blir utført en erosjon, for å så utføre en dilatasjon. Erosjon fjerner piksler på objektenes kanter, og dilatasjon gjenoppretter objektets størrelse uten at de minste partiklene kommer med.

Avslutningsvis blir maskene lastet opp som et datasett i Kaggle. Deretter blir de behandlet videre for å sikre at de er binære, altså at det er kun svart og hvitt i maskene. Dette må gjøres for å sikre at disse er i et format som kan bli akseptert av modellen. Figur 5-1 nedenfor viser et bilde fra datasettet, der venstre halvdel er ubehandlet og høyre halvdel viser hvordan masken er tegnet opp.

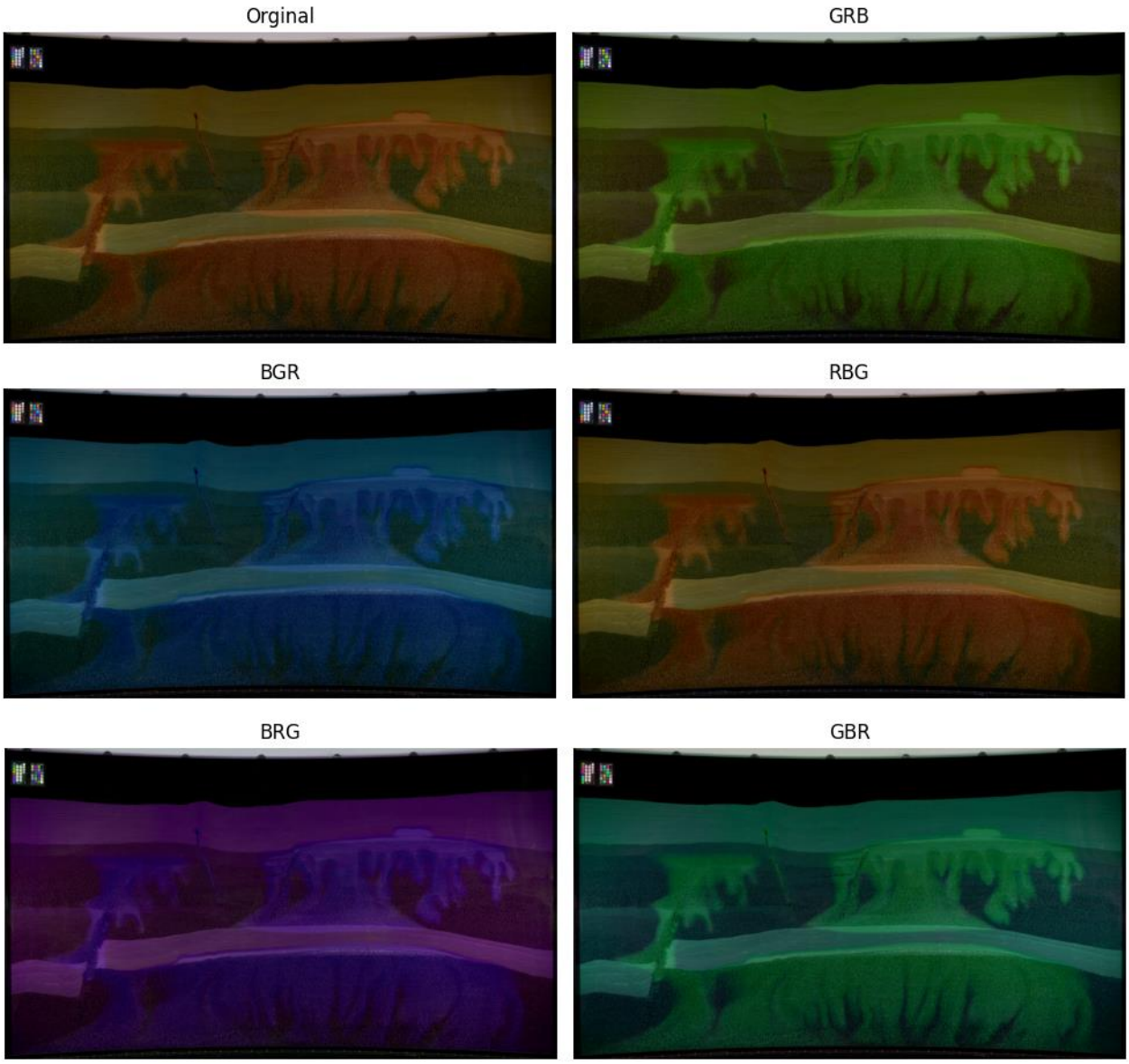
Figur 5-1 - Bilde som viser et halvt ubehandlet rød/grønn bilde fra datasettet og andre halvdel tegnet opp som en maske

5.2.4 Forberedning av bilder

For å redusere mest mulig bakgrunnsstøy på bildene ble først de øverste 700 pikslene av bildet fjernet. Dette er fordi at denne delen av bildet inneholder hverken CO2 eller andre elementer som vi er interessert i, samt en farge-graf som er brukt for kalibrering av bilder som kan påvirke treningen av modellen. Dermed kan modellen fokusere mer på de områdene CO2 faktisk kan være til stede.

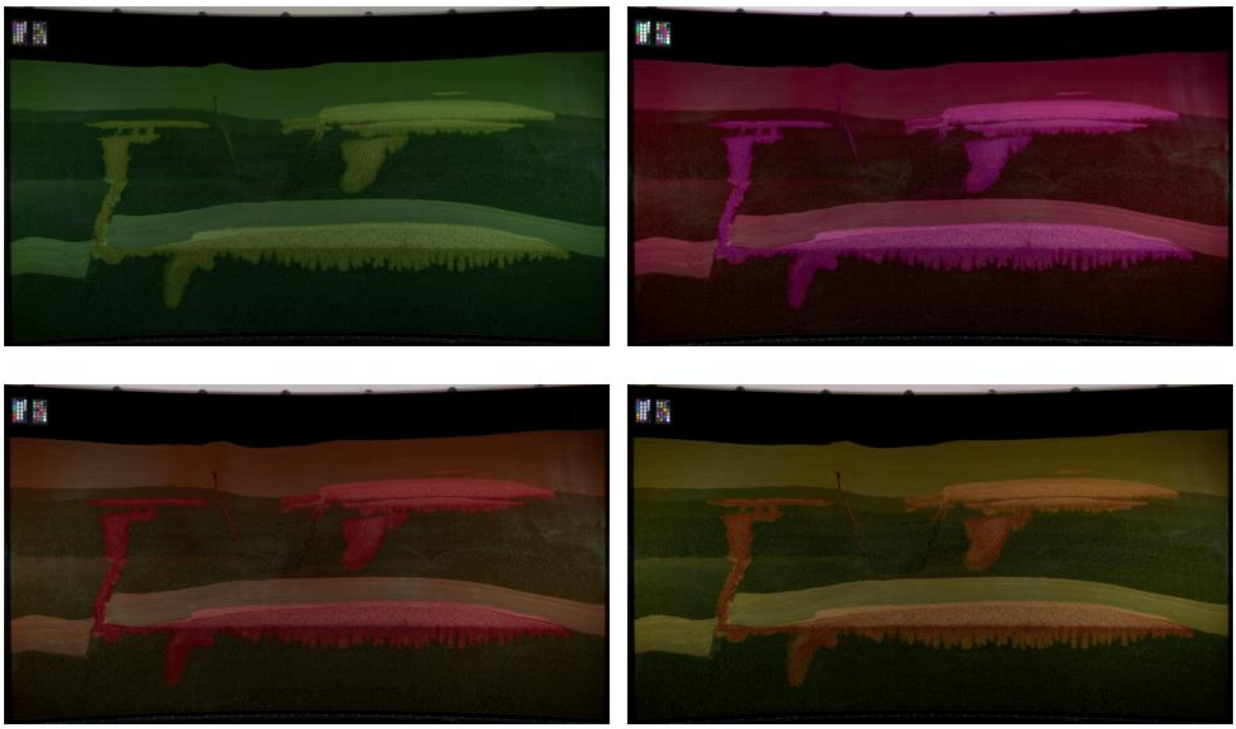
5.2.5 Dataaugmentering

I denne seksjonen blir det gått gjennom teknikkene brukt for å gjøre augmenteringer på datasettet og hvordan disse ble gjennomført. Dette inkluderer teknikker som bytting av fargekanaler og geometriske transformasjoner, samt justeringer av lysstyrke og kontrast. Alle bildene i datasettet ble først behandlet ved å manipulere fargekanalene i bildet. Det ble totalt gjort 5 forskjellige bytter av fargekanalene. RGB er standarden på fargekanalen, og bildene ble stokket om til følgende verdier GBR, BRG, GRB, RBG og BGR. Et eksempel på dette er at RGB-verdien (255, 100, 200), blir til GBR-verdien (100, 200, 255). På denne måten blir datasettet 6 ganger større enn det var i utgangspunktet. Figur 5-2 viser eksempler på bilder etter fargekanalene er byttet.



Figur 5-2 - Eksempler på bilder fra datasett etter å ha byttet fargekanaler

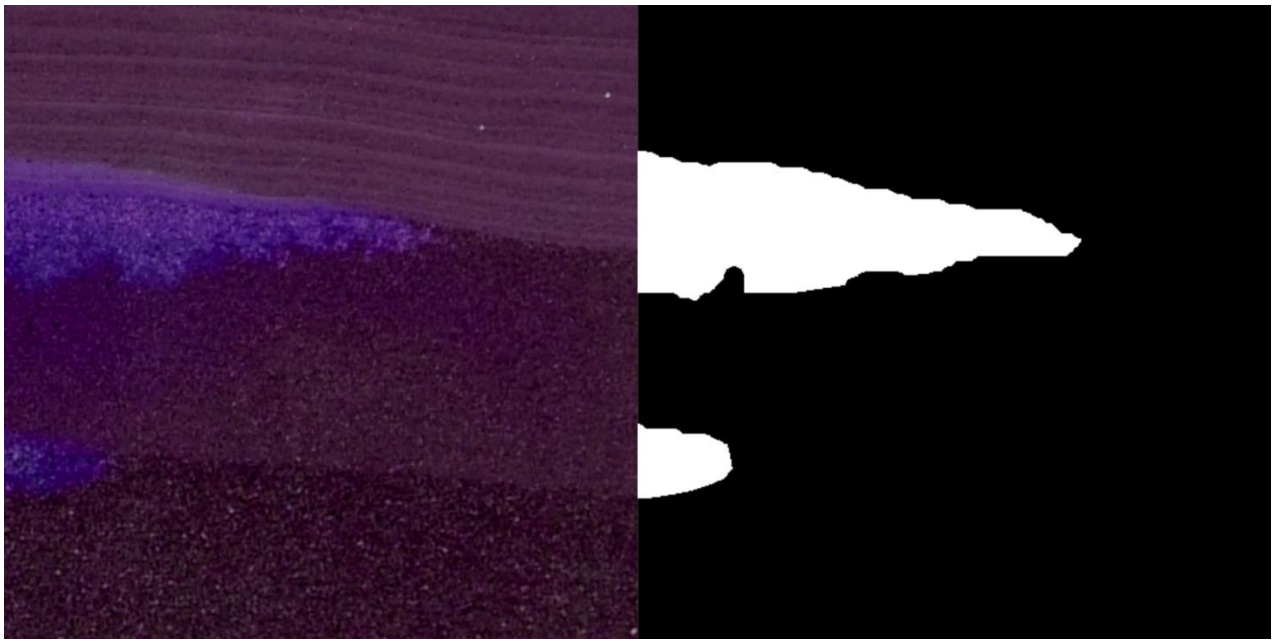
For å skape ytterlige variasjon i bildene blir det her tatt i bruk PyTorch sitt TorchVision bibliotek, som inneholder en funksjon som heter Color Jitter. Denne funksjonen utfører tilfeldige justeringer på lysstyrke, metning, kontrast og fargetonen på bildet. Figur 5-3 viser bilder som er blitt augmentert med Color Jitter funksjonen. Til slutt blir bildene vendt horisontalt, og deretter lagt til igjen i datasettet (Torchvision, u.å.).



Figur 5-3 - Eksempel på bilder fra datasett etter at det har blitt utført Color Jitter

5.2.6 Segmentering av bilder

Etter augmenteringene på datasettet har blitt utført, må de tilpasses slik at de er på et format som passer til modellen. Det er grunnet at modellen ikke kan trenes opp på bilder som har en oppløsning på 7952 x 4472, da disse er for store til å kunne bli effektivt håndtert av modellene, spesielt når en trener flere bilder samtidig. Som en løsning på dette blir bildene delt opp i mindre segmenter med en størrelse på 224 x 224. Dette er en dimensjon som brukes i mange pre-trente modeller. Den originale bildekvaliteten blir dermed beholdt, som gjør at viktige detaljer og former på bildene beholdes. Bildene som deles opp blir indeksert etter deres opprinnelige posisjon, og deretter lagret i en mappe for hvert segment av bildet. Det samme blir også gjort for maskene slik at hvert segment har en tilhørende maske. Figur 5-4 viser et eksempel på et bildesegment, med tilhørende maske.



Figur 5-4 - Eksempel på et segment, med et fargeaugmentert bilde til venstre og tilhørende maske til høyre

5.4 Trening av modell

Denne delen omhandler selve treningen av modellene. Det trenes opp fire modeller; Resnet50, AlexNet, Squeezenet1_1 og VGG16_bn.

Før selve treningen kan starte, må en gå gjennom noen steg for å sikre at modellen er satt opp riktig. Først må læringsraten finnes og settes, deretter må det bli valgt partistørrelse og til slutt valgt antall epoker modellen skal trenes.

5.4.1 Læringsrate

Læringsrate styrer hvor store "steg" som blir tatt under trening av modellen, med andre ord hvor raskt eller sakte en modell lærer. For å finne læringsraten for hver modell blir det brukt en funksjon fra fast.ai, lr_find. Denne funksjonen gir en graf som viser hvordan forskjellige læringsrater påvirker treningsprosessen (Mavropalias, 2019).

5.4.2 Partistørrelse

Størrelsen på partiet sier noe om hvor mange bilder som prosesseres samtidig under en iterasjon. En større partistørrelse (batch size) kan redusere treningstiden, men det vil også kreve mer minne av grafikkortet. Hver modell blir derfor testet for å kunne finne størst mulig partistørrelse som GPU-minnet kan håndtere.

5.4.3 Valg av antall epoker

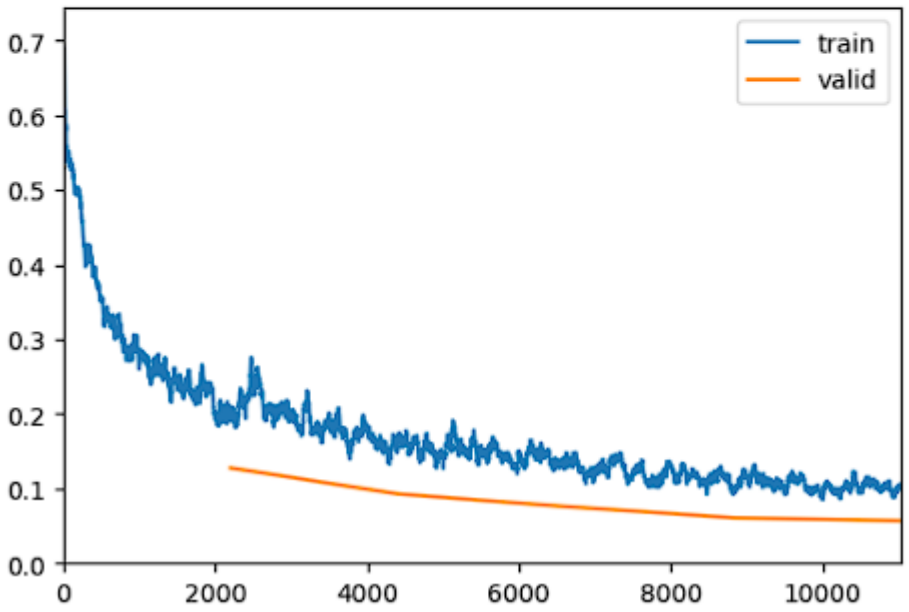
For å sikre et godt sammenligningsgrunnlag mellom modellene, er treningstiden på modellene tilnærmet lik. Hver modell trenes i omtrent 6 timer. For å kunne bestemme hvor mange epoker hver modell kan kjøre på gitt tid, testes treningstidene på hver modell. I neste steg av prosessen trenes modellen.

5.4.4 Trening

Treningen gjøres ved å bruke fast.ai sin fit_one_cycle-metode, som er en metode for å trene nevrale nettverk. Metoden implementerer en teknikk som kalles ”one cycle learning rate policy” som justerer opp læringsraten gradvis i første halvdel av treningen, for å så gradvis justere den ned igjen mot slutten av treningen. Dette tillater modellen å utforske et bredere område av løsningsrommet raskt, før den finjusterer seg mot mer stabile løsninger. I praksis betyr dette at ved å bruke denne metoden, kan en potensielt trene modellere raskere og oppnå bedre resultater med mindre manuelle innstillinger av læringsraten (Mavropalias, 2019).

Under treningsprossesen blir det målt og loggført for treningstap (train loss), validering tap (validation loss), dice score og nøyaktighet (Accuracy) for videre analyse etter treningen. Se punk 6.1.1 for mer detaljer om disse metodene. Treningsdataen og den trente modellen eksporteres for videre analyse. Figur 5-5 viser hvordan treningsdataen blir presentert under trening.

epoch	train_loss	valid_loss	dice	segmentation_accuracy	Recall	time
0	0.205321	0.127682	0.911334	0.961639	0.942921	1:10:27
1	0.162219	0.092786	0.929328	0.969151	0.957129	1:10:23
2	0.138129	0.075934	0.935783	0.971810	0.965354	1:10:19
3	0.094162	0.060962	0.944449	0.976864	0.950643	1:10:25
4	0.104249	0.057072	0.950220	0.978881	0.962294	1:10:31



Figur 5-5 - Figuren viser Treningsdata presentert som en graf. Viser train loss og valid loss for hvert parti.

5.6 Test og evaluering

Etter modellene er trent eksporteres de til en ny Kaggle notebook for å teste og sammenligne de forskjellige modellene på nye bilder. Resultatet av dette blir presentert i kapittel 6. Det velges ut varierte bilder vi ønsker å teste og sammenligne på. Disse segmenteres opp i mindre bilder, slik at de kan tilpasses modellen. Det predikeres på hvert segment, og segmentene settes sammen igjen til et bilde. Prediksjonene blir presentert sammen med det ordinære bildet for sammenligning. Resultatet fra treningen av de forskjellige modellene blir tegnet opp i grafer og tabeller for å videre kunne sammenligne.

6. Resultater

Dette kapittelet beskriver evalueringsmetoder og verktøy som brukes for å bedømme resultatet av prosjektet, som vil si modellenes evne til å klassifisere CO2. Deretter vises resultatene fra de forskjellige modellene, både visuelt i form av grafer og matriser, samt i form av tabeller. Alle resultatene er fra trening og testing av modellene, på rød/grønn bilder.

6.1 Evalueringsmetode

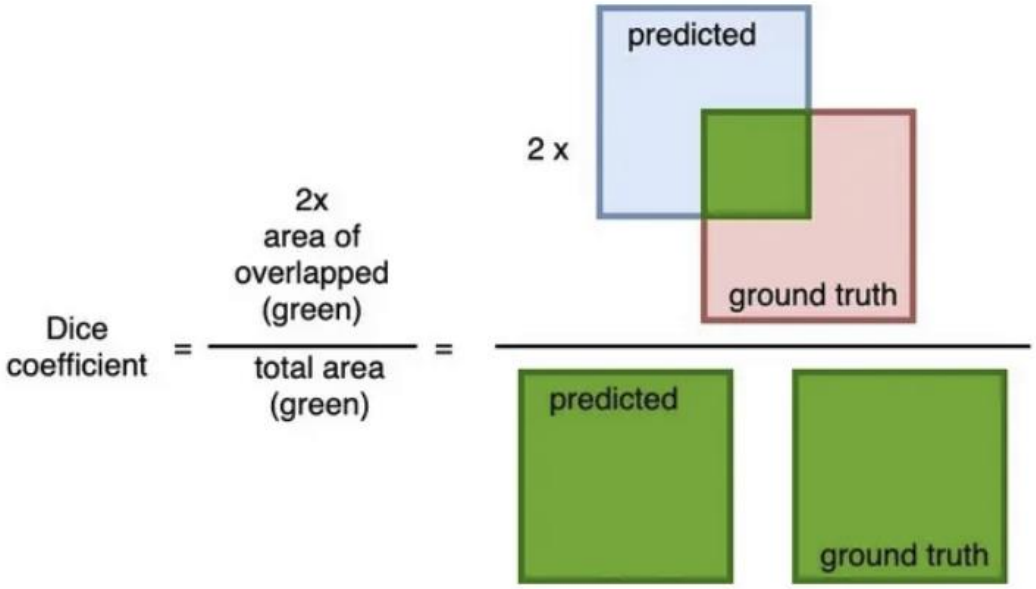
For å evaluere prosjektet er det brukt diverse evalueringsmetoder og verktøy. Her vil det bli gått igjennom de metodene som er brukt i dette prosjektet.

6.1.1 Oversikt over evalueringsmetoder og verktøy:

For å evaluere prosjektets kvalitet og relevans brukes det kvalitative og kvantitative metoder. Innenfor maskinlæring finnes det tap (loss) funksjoner som brukes for å bedømme hvor bra resultatet er, og taps funksjonene bedømmer dette på forskjellige måter. Resultatene viser til dice loss, trenings- og validerings tap (loss) og nøyaktighet (accuracy), for å evaluere de kvantitative resultatene.

Dice loss

Dice loss er en måleenhet i maskinlæring for å bedømme hvor god en modell er på å segmentere bilder og i hvor stor grad segmentene overlapper hverandre. En dice loss på 0 betyr at resultatet ikke overlapper fasit i det hele tatt, mens en dice loss på 1 betyr at segmentene overlapper hverandre perfekt. Figur 6-1 nedenfor viser hvordan koeffisienten blir regnet ut. Målet er altså å maksimere dice loss for å få mest mulig overlapp mellom to segmenteringer (Schumacher, u.å).



Figur 6-1 - Bilde av dice loss utregning (Neville,2023)

Trenings tap (Training loss)

Trenings tap er en måleenhet for å se hvor bra modellen passer til treningsdata i datasettet. Det betyr at den måler hvor mye feil modellen tar når den trener på datasettet. Med trenings tap er målet å minimere antall feil modellen gjør, som betyr at desto lavere resultatet er, desto bedre er modellen. Trenings tap er kalkulert med å ta summen av alle feilene, og dette blir målt etter hvert parti(batch) i treningssyklusen (Baeldung, 2024).

Validerings tap (Validation loss)

I motsetning til trenings tap, er validerings tap en måleenhet for å se hvor bra modellen gjør det på usett data, altså på valideringsdelen av datasettet. Dette er den delen av datasettet som modellen ikke trener på. Validerings tap er kalkulert på samme måte som trening tap. Dette betyr at validerings tap sier noe om hvor bra modellen generaliserer til ny data. Validerings tap er kalkulert etter hver epoke (epoch). Et lavt resultat vil si at modellen generaliserer bra (Baeldun, 2024).

Nøyaktighet (Accuracy)

Nøyaktighet er en måleenhet som måler hvor ofte en maskinlæringsmodell får riktig resultat på prediksjonene sine. Nøyaktighet måles ved å dele antall riktige prediksjoner, på totalt antall prediksjoner. Med andre ord, så sier nøyaktighet hvor ofte modellen har rett. Nøyaktighet blir målt

på en skala mellom 0 og 1, der et resultat på 0 sier at modellen ikke er nøyaktig i det hele tatt, og et resultat på 1 sier at modellen er helt nøyaktig (Rastogi, 2023).

6.1.2 Validering

Det gjennomføres jevnlige møter med oppdragsgiver, der det diskuteres hva som er blitt gjort fra forrige møte og hvilke utfordringer gruppen har møtt på og hvordan de har blitt løst. Som en del av møtene gir oppdragsgiver tilbakemelding på hvordan resultatene er, og hva som mangler for å få en bedre løsning i henhold til ønsket sluttresultat. Gjennom prosjektets levetid jobbes det iterativt sammen med oppdragsgiver for å sørge for at det blir utviklet riktig løsning.

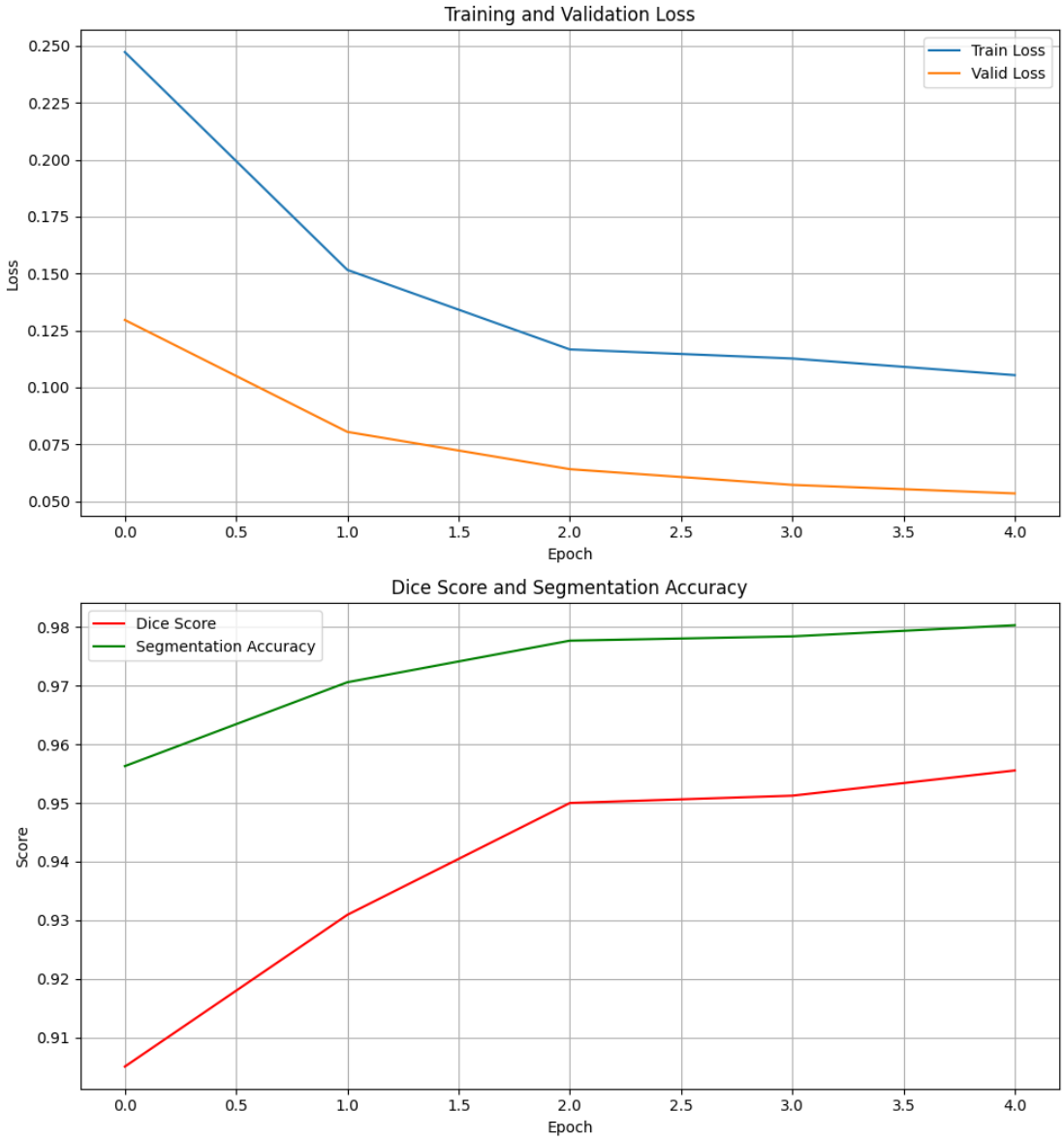
6.2 Resultater med data augmentering

Her presenteres grafer som en visualisering av resultatene til de ulike modellene som er testet i forbindelse med prosjektet. Videre presenteres tabeller som viser de forskjellige evalueringsresultatene sammen med standardavvik over alle kjørte epoker. Augmenteringene som gjøres på alle modellene er bytting av fargekanaler, color jitter (endre lysstyrken, metningen, fargetonen og kontrasten i bildene) og horisontal flipping av bildene. Alle modellene testes på samme bildene, som alle er av type rød/grønn bilder.

Tabell 6-1 til Tabell 6-4 nedenfor viser gjennomsnittet av epokene den gjeldende modellen har kjørt for å trene på datasett med dataaugmentering. Tabellene gir en evaluering av trenings tap, validerings tap, dice score og nøyaktighet, samt standardavvik.

6.2.1 ResNet50

Nedenfor viser Figur 6-2 resultatene fra trening av ResNet50 modellen, med dataaugmentering. Modellen kjøres i 5 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-1.



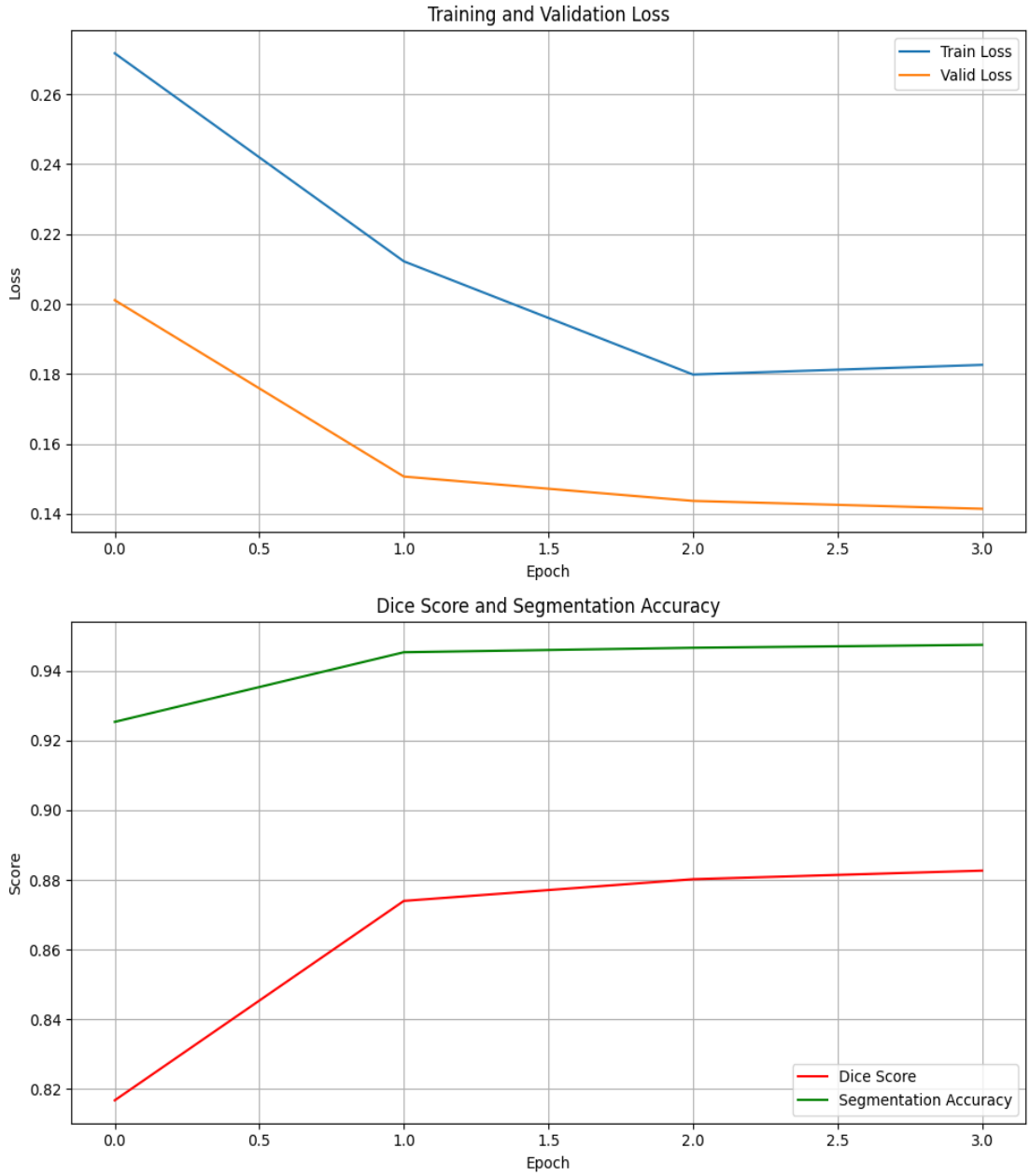
Figur 6-2 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for ResNet50. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
ResNet50 Augmented	**0.15 ± 0.06**	**0.08 ± 0.03**	**0.94 ± 0.02**	**0.97 ± 0.01**

Tabell 6-1- Tabell med train loss, valid loss, dice score og accuracy for ResNet50 trent på datasett med augmenteringer

6.2.2 VGG16_bn

Nedenfor viser Figur 6-3 resultatene fra treningen av VGG16_bn modellen, med dataaugmentering. Modellen kjøres i 4 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-2.



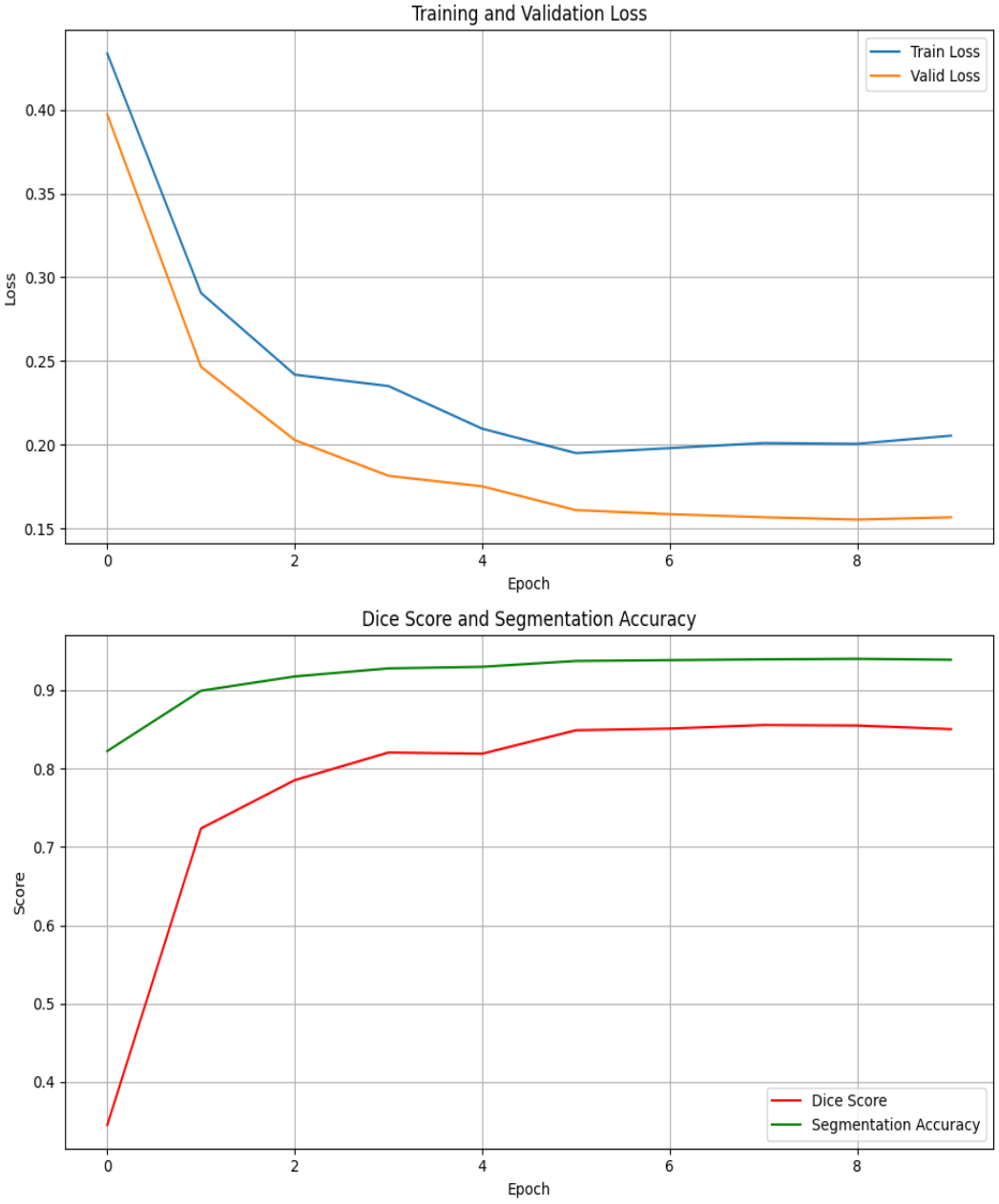
Figur 6-3 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for VGG16_bn. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
VGG16_bn Augmented	**0.21 ± 0.04**	**0.16 ± 0.03**	**0.86 ± 0.03**	**0.92 ± 0.04**

Tabell 6-2 - Tabell med train loss, valid loss, dice score og accuracy for VGG16_bn trent på datasett med augmenteringer

6.2.3 AlexNet

Nedenfor viser Figur 6-4 resultatene fra treningen av AlexNet modellen, med dataaugmentering. Modellen er kjørt i 10 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-3.



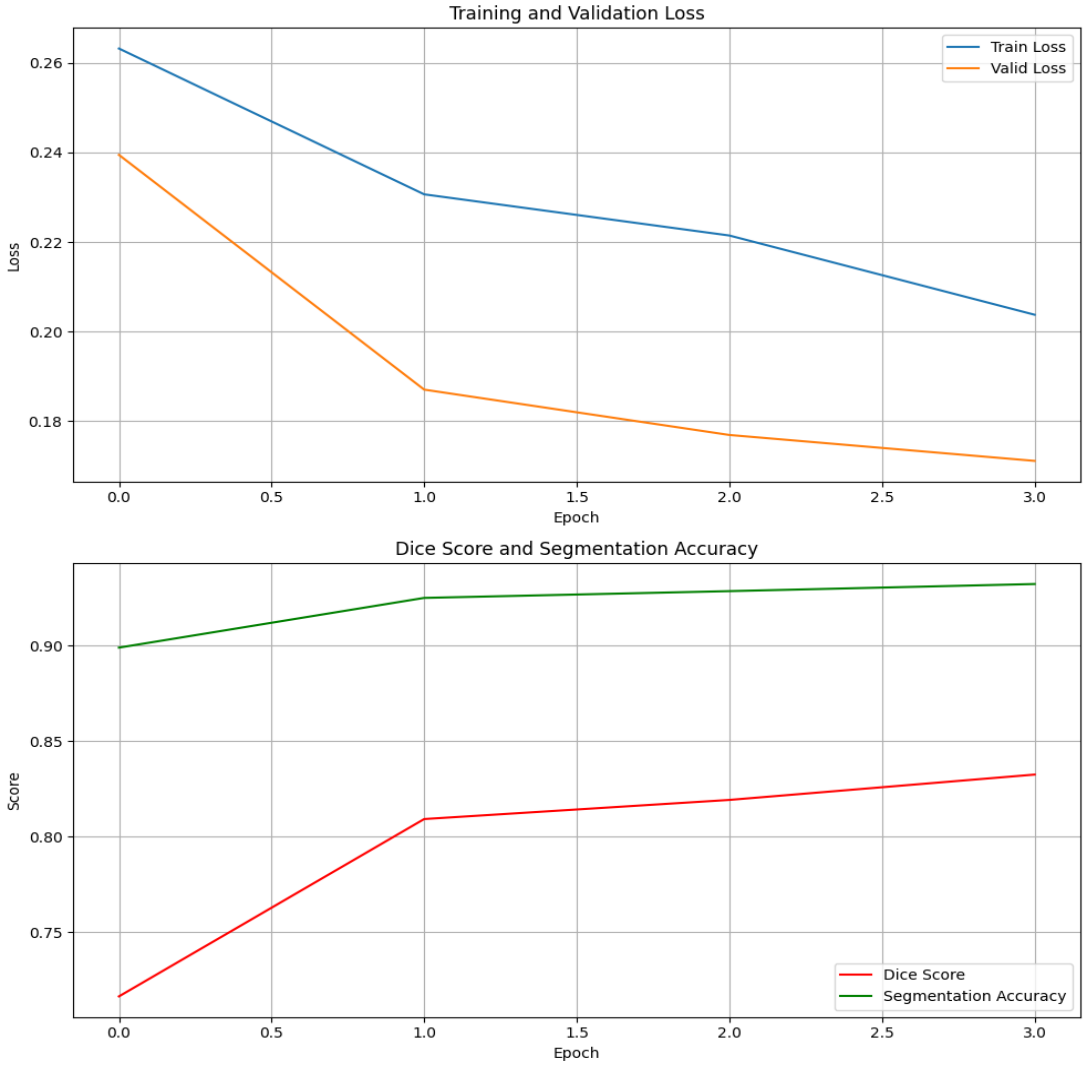
Figur 6-4 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for AlexNet. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
AlexNet Augmented	**0.24 ± 0.07**	**0.20 ± 0.08**	**0.78 ± 0.16**	**0.92 ± 0.04**

Tabell 6-3 - Tabell med train loss, valid loss, dice score og accuracy for AlexNet trent på datasett med augmenteringer

6.2.4 SqueezeNet

Nedenfor viser Figur 6-5 resultatene fra treningen av SqueezeNet modellen, med dataaugmentering. Modellen kjøres i 4 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-4.



Figur 6-5 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for SqueezeNet. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
SqueezeNet1_1 Augmented	**0.23 ± 0.02**	**0.19 ± 0.03**	**0.79 ± 0.05**	**0.92 ± 0.02**

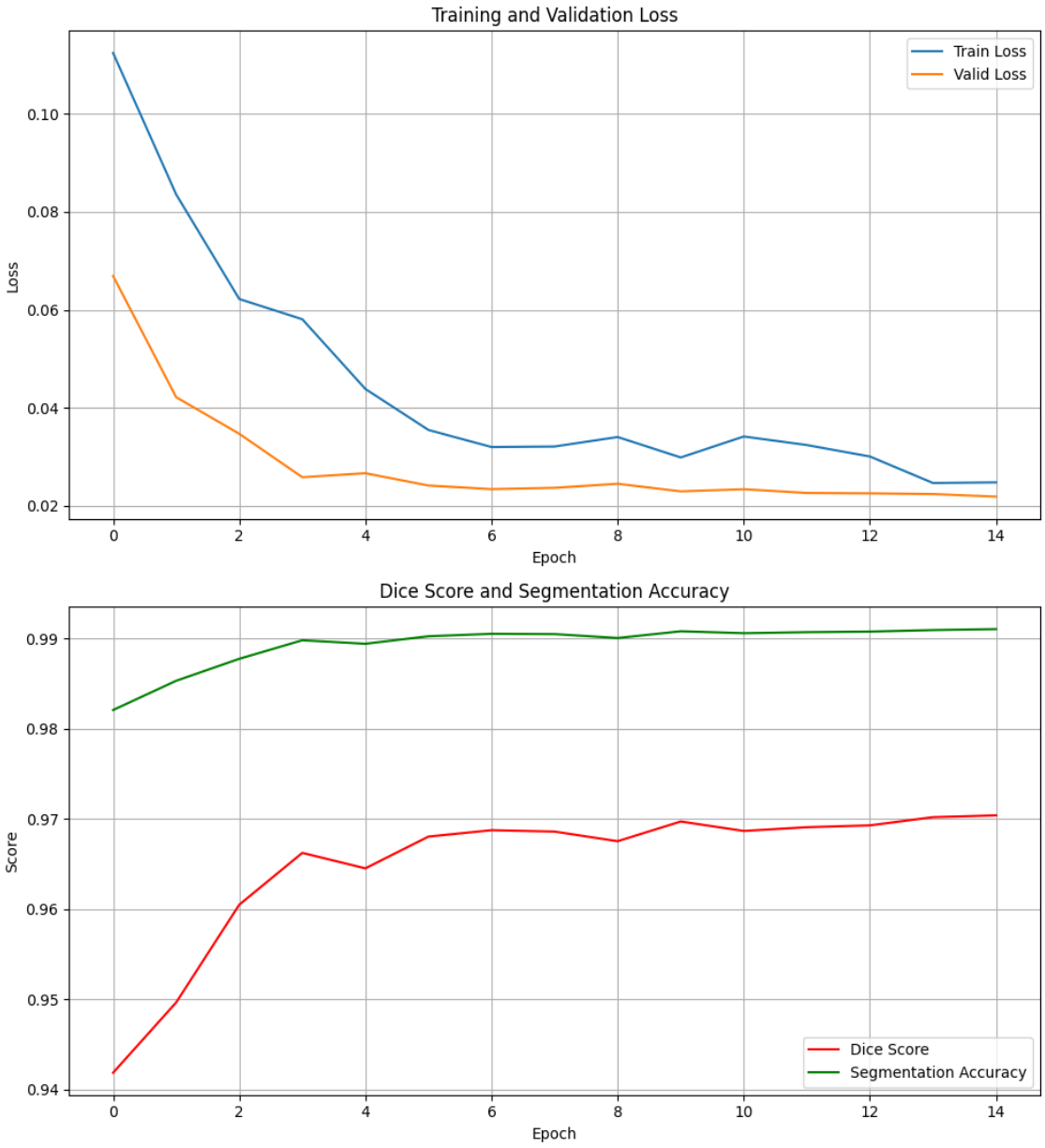
Tabell 6-4 - Tabell med train loss, valid loss, dice score og accuracy for SqueezeNet trent på datasett med augmenteringer

6.3 Resultater uten data augmentering

Her presenteres grafer for å visualisere resultatene av trening på datasett uten augmenteringer, gjennom alle epokene. Det presenteres også en tabell som viser de forskjellige evalueringsresultatene sammen med standardavvik over alle kjørte epoker. Alle modellene testes og trenes på samme rød/grønn bilder.

ResNet50:

Nedenfor viser Figur 6-6 resultatene fra treningen av ResNet modellen, uten dataaugmentering. Modellen kjøres i 15 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-5.



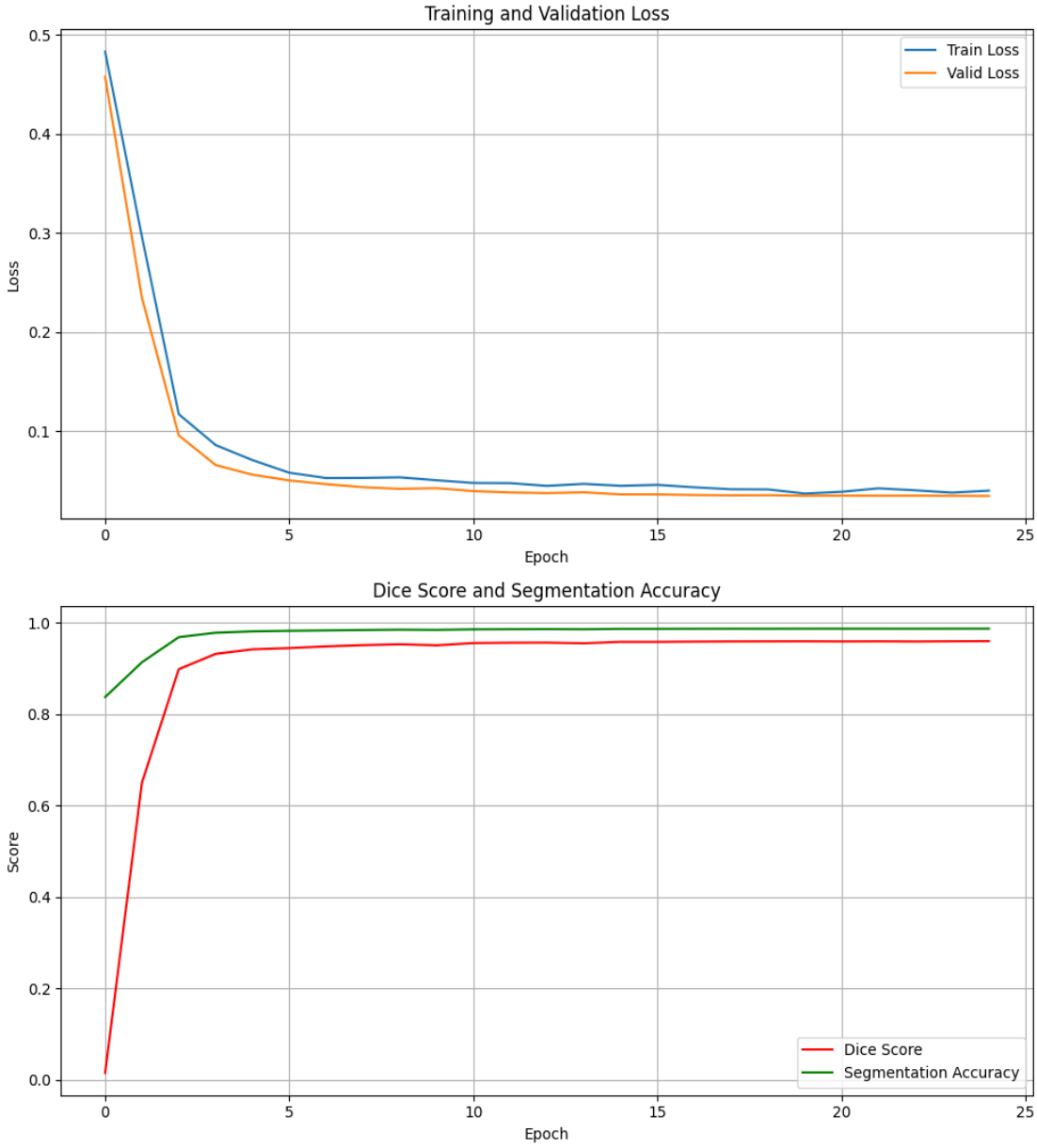
Figur 6-6 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for ResNet50. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
ResNet50	**0.04 ± 0.02**	**0.03 ± 0.01**	**0.96 ± 0.01**	**0.99 ± 0.00**

Tabell 6-5 - Tabell med train loss, valid loss, dice score og accuracy for ResNet50 trent på datasett uten augmenteringer

AlexNet:

Nedenfor viser Figur 6-7 resultatene fra trening av AlexNet modellen, uten dataaugmentering. Modellen er kjørt i 25 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-6.



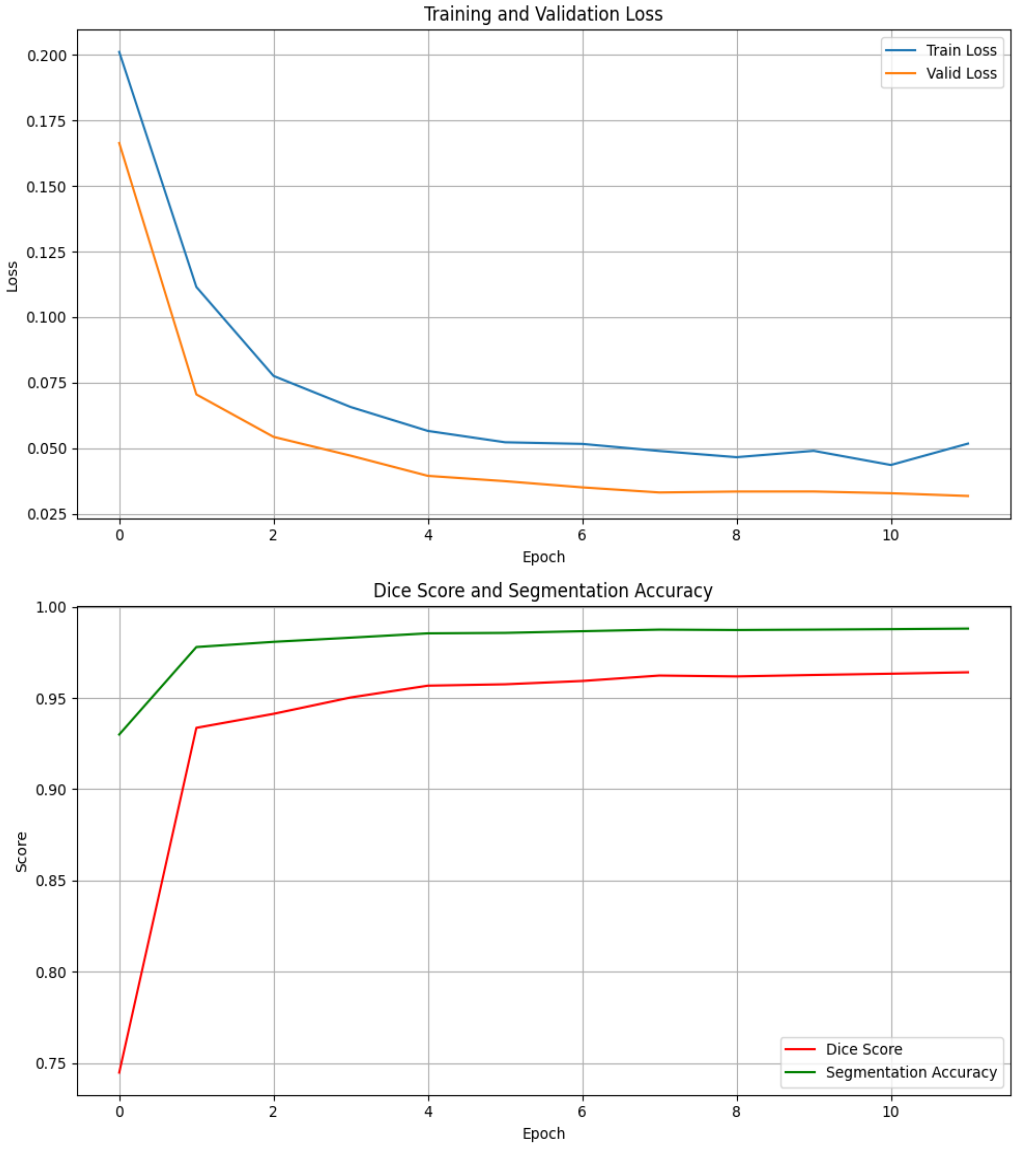
Figur 6-7- Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for AlexNet. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
AlexNet	**0.08 ± 0.10**	**0.07 ± 0.09**	**0.90 ± 0.19**	**0.98 ± 0.03**

Tabell 6-6 - Tabell med train loss, valid loss, dice score og accuracy for AlexNet trent på datasett uten augmenteringer

VGG16_bn:

Nedenfor viser Figur 6-8 resultatene fra treningen av VGG16_bn modellen, uten dataaugmentering. Modellen er kjørt i 12 epoker. Gjennomsnittet og standardavviket av treningen vises i Tabell 6-7.



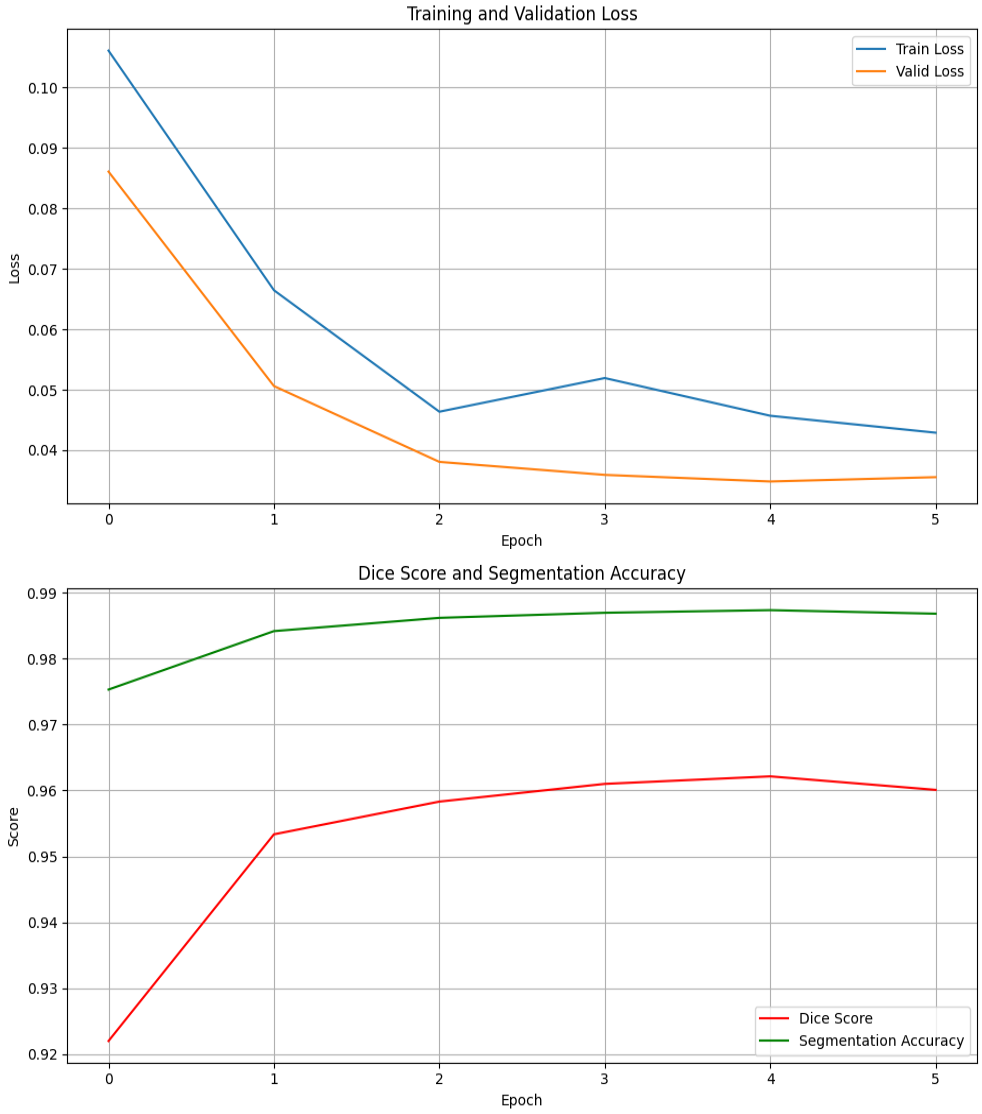
Figur 6-8 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for VGG16_bn. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
VGG16_bn	0.07 ± 0.04	0.05 ± 0.04	0.94 ± 0.06	0.98 ± 0.02

Tabell 6-7 - Tabell med train loss, valid loss, dice score og accuracy for VGG16_bn trent på datasett uten augmenteringer

SqueezeNet:

Nedenfor viser Figur 6-9 resultatene fra treningen av SqueezeNet modellen, uten dataaugmentering. Modellen er kjørt i 6 epoker. Gjennomsnittet og standardavviket av kjøringene vises i Tabell 6-8.



Figur 6-9 - Figuren viser en graf som representerer train-, valid-, og dice loss, samt accuracy for SqueezeNet. Blå graf viser train loss, oransje viser valid loss, grønn viser segmentation accuracy og rød viser dice score.

Modell	Train Loss	Valid Loss	Dice Score	Accuracy
SqueezeNet1_1	0.06 ± 0.02	0.05 ± 0.02	0.95 ± 0.02	0.98 ± 0.00

Tabell 6-8 - Tabell med train loss, valid loss, dice score og accuracy for SqueezeNet trent på datasett uten augmenteringer

7. Diskusjon

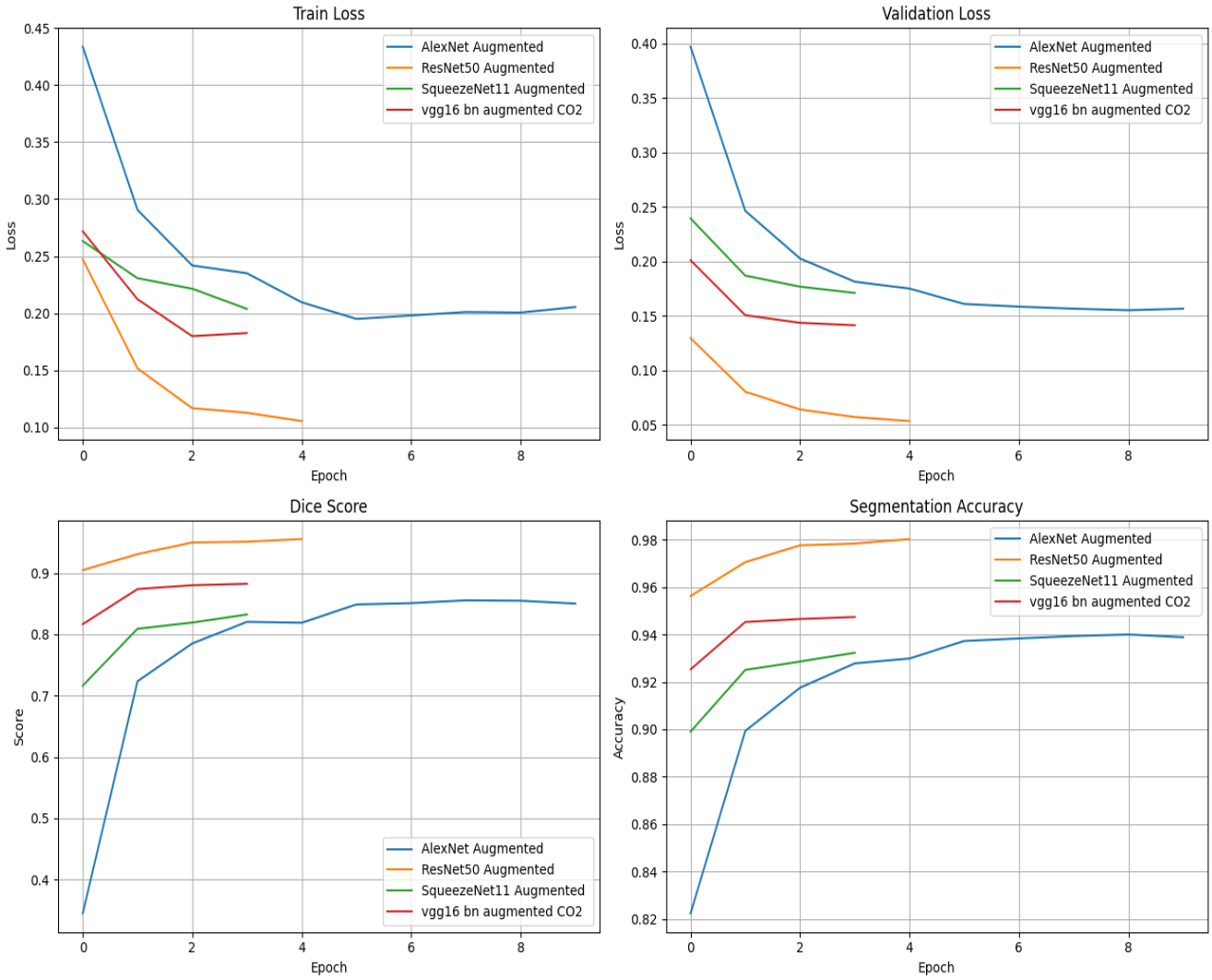
I diskusjonen sammenlignes resultatene fra de ulike modellene og hvordan de presterer, samt hva resultatene betyr. Det mest overraskende fra resultatene er at modellene som trenes på datasett uten dataaugmentering, oppnår bedre resultat enn de som er trent med dataaugmentering. I dette kapittelet legges det frem kvantitative og kvalitative data som blir samlet inn gjennom de ulike evalueringsmetodene. Deretter blir det gått igjennom begrensninger og utfordringer, før det gjøres en sluttevaluering og en gjennomgang av de helhetlige perspektivene for å vurdere prosjektet i en større kontekst. De kvantitative resultatene som diskutere i dette kapittelet er basert på trening og validering av rød/grønn bilder, mens resultatene av tester på gul/blå bilder presenteres i den kvalitative delen av diskusjonen. Alle modellene trenes i omtrent 6 timer, for å gjøre rettferdige sammenligninger. Dette vil si at mindre komplekse modeller vil kunne kjøre igjennom flere epoker, sammenlignet med de mer komplekse modellene. En interessant sammenligning her er om en mer kompleks modell som kjøres igjennom få epoker, er bedre eller dårligere enn en mindre kompleks modell som kjøres gjennom flere epoker.

7.1 Sammenligning av resultater med dataaugmentering

Her diskuteres de kvantitative og kvalitative resultater fra modellene som er testet på datasett med dataaugmentering. Den kvantitative delen av kapittelet snakker om resultatene som er hentet ut fra grafene og tabellene som er presentert i kapittel 6.2, og sammenligner de ulike modellene opp mot hverandre. Den kvalitative delen viser bildene de ulike modellene har produsert og sammenligner resultatene. Det testes fire forskjellige modeller, AlexNet, ResNet50, SqueezeNet og Vgg16_bn som alle gir forskjellige kvantitative og kvalitative resultater.

7.1.1 Analyse av kvantitative resultater

Her presenteres de kvantitative resultatene fra AlexNet, ResNet50, SqueezeNet og Vgg16_bn. Figur 7-1 viser sammenligning av treningsresultatene til alle modellene, med de forskjellige evalueringsmetodene.



Figur 7-1 - Figuren viser resultater for AlexNet, ResNet50, SqueezeNet1_1, VGG16_bn under trening på augmentert datasett

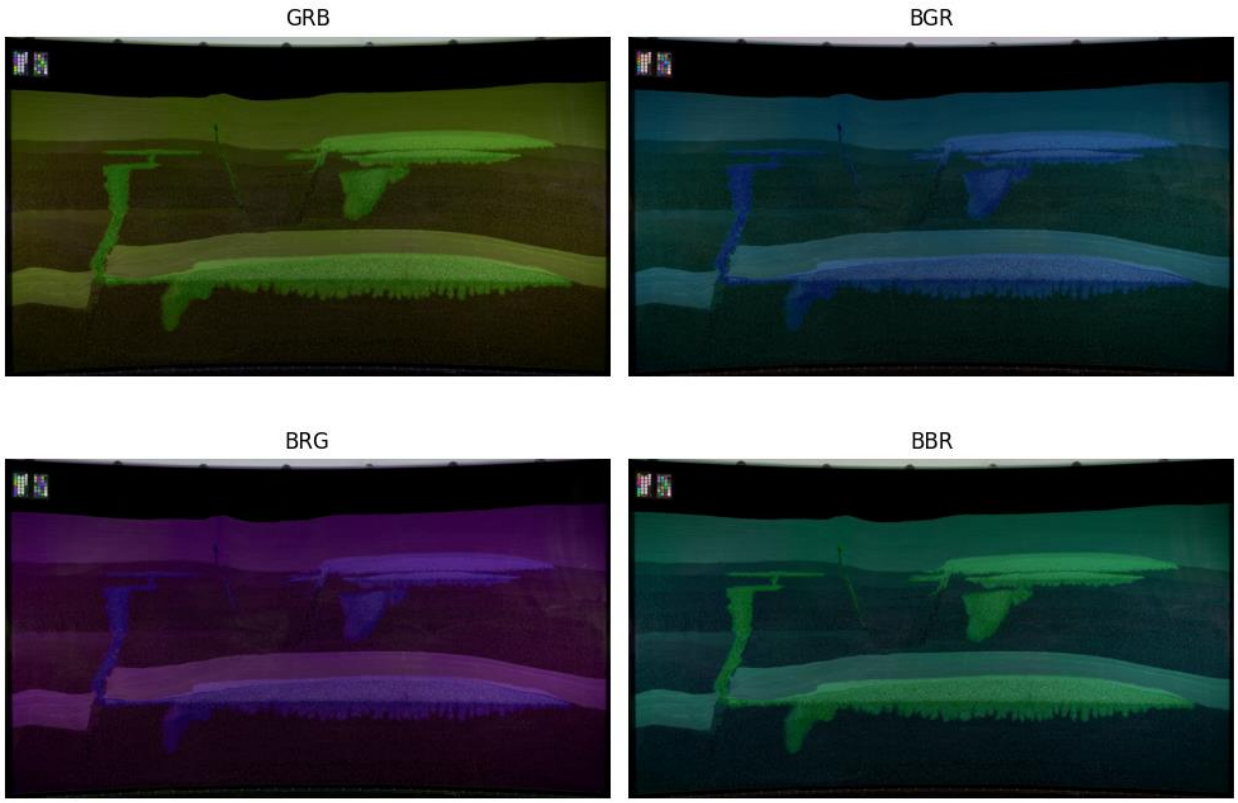
Hvorfor har det blitt brukt de spesifikke dataaugmenteringsmetodene?

Som en del av det å kunne få et mer variert og større datasett, har det blitt sett på mange mulige augmenteringer og modifikasjoner av bildene. Det skal i utgangspunktet ikke være nødvendig å gjøre flere geometriske augmenteringer, da det vil gjøre at modellen trener seg opp på å gjenkjenne strukturer og former i bildet som ikke vil være der på et bilde som ikke er augmentert. Augmenteringene som har blitt valgt, er grunnet at det er de som ble vurdert til å være de enkleste og mest tidseffektive for dette prosjektet. Samtidig burde det vært gjort tester på de forskjellige augmenteringene mellom hver treningsiterasjon av modellene, for å se den faktiske effekten av hver enkelt augmentering og deretter vurdert om effekten av augmenteringen er tilstrekkelig. Dersom den viser seg og ikke være effektiv, så burde den vært byttet ut med en annen augmentering som kan vise til bedre resultater og tilsvarende tidseffektivitet.

Bytte av fargekanaler:

Det ble gjort totalt seks forskjellige stokkinger av fargekanalene, som vil si å bytte ut hvilke verdier som er på hvilken fargekanal. Bytting av fargekanalene var ansett som den viktigste augmenteringen, da gruppen tidlig ble gjort oppmerksom på at fargene i bildene vil endre seg med tiden. Derfor er det viktig at modellen ikke baserer seg på hvilke farger som er i bildet, men heller på den underliggende strukturen. Optimalt burde denne augmenteringen blitt testet tidligere i prosjektet for å se om den faktisk førte til bedre generalisering. Da det ikke ble tildelt bilder med nye farger før i slutten av prosjektet, var ikke dette noe som var lett å gjennomføre. Det kunne derimot vært laget tilsvarende bilder som i datasettet, med andre farger, for å kunne teste generaliseringen.

Figur 7-2 viser eksempler på bilder der det er utført fargeaugmenteringer. GRB, BGR og BBR bildene viser et ønsket resultat av fargeaugmenteringen, der det er klare skiller mellom fargene på CO2-strukturen og sandlagene rundt. Derimot viser BRG bildet farger i CO2-strukteren og sandlagene som muligens er for like til at modellen differensierer fargene godt nok. Her skulle det nok vært en mer selektiv utvelgelse av hvilke fargeaugmenteringer som ble brukt til trening av modell.

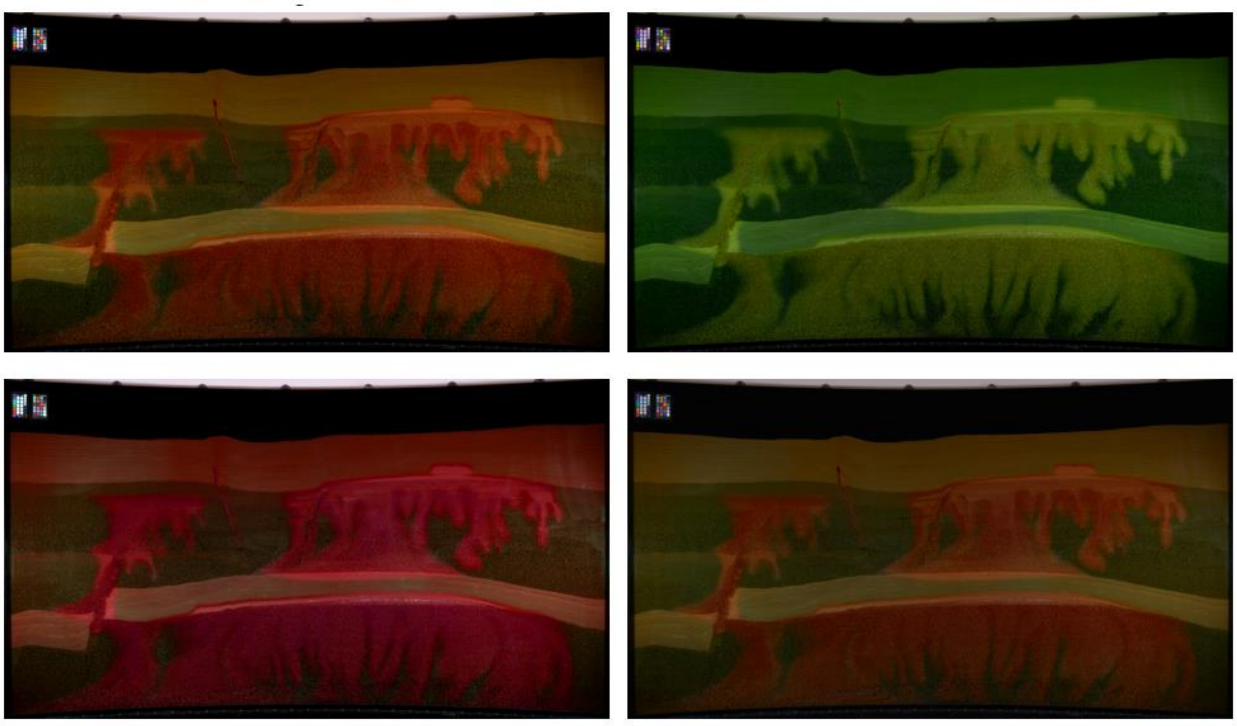


Figur 7-2 - Eksempler på bilder fra datasett etter å ha utført fargeaugmenteringer.

Color Jitter:

En annen augmentering som ble gjort, er noe som kalles color jitter. Color Jitter er en augmentering som endrer lysstyrken, metningen, fargetonen og kontrasten i bildene. Grunnen til at denne augmenteringen ble valgt, var i utgangspunktet for å ytterliggere sikre at modellen ikke skal basere seg på farger i bildet. Dette har trolig vært en mangelfull vurdering, da lysstyrke, metningen, fargetone og kontrast i bildet ikke nødvendigvis vil gjøre modellen bedre på generalisering av farger. Det kunne vært mer naturlig å bruke de augmenteringene dersom det var variasjon av bildekvalitet, lysforhold og lignende på bildene i datasettet. Dette vil ikke være tilfelle, da alle bildene vil bli tatt under samme forhold.

Som en kan se på bildene i Figur 7-3, har denne augmenteringene muligens tatt vekk for mange detaljer i bildene, da fargene blir for like. Som eksempelvis det grønne bildet, er det kun forskjellige nyanser av grønt, som igjen kan gjøre at modellen ikke vet hva den skal se etter i bildet.



Figur 7-3 - Eksempler på bilder fra datasett etter å ha utført Color Jitter

Horisontal vending av bildene:

En horisontal vending av bildet vil kunne bidra til å øke datasettet uten at det skal påvirke den underliggende strukturen i bildet og hvordan de grunnleggende CO2-strukturene ser ut. En vertikal vending vil til gjengjeld endre hele CO2-strukturen, og strukturen vil dermed bli feil grunnet at CO2 kun renner nedover, og aldri oppover. Derfor ble det besluttet å ikke ha vertikal vending av bildet, da modellen vil kunne begynne å se etter strukturer som aldri kan eksistere i et reelt bilde.

Resultat av de forskjellige modellene

ResNet50:

ResNet50 modellen har gått igjennom 5 epoker, og begynte med et trenings tap på 0.247 og endte på 0.105 som tilsier at treningen av modellen er veldig effektiv. Validerings tap begynner på 0.130 og reduseres kraftig gjennom epokene før den ender på 0.053, som tilsier at modellen generaliserer ekstremt bra. Dice score er veldig bra fra første iterasjon, og har en kraftig økning fra 0.905 til 0.956 som sier at modellen er høyst effektiv på bildesegmentering. Segmenteringsnøyaktighet begynner på 95.63% og øker til 98.03%, som vil si at modellen har en utmerket nøyaktighet.

Vgg16_bn:

VggNet16 modellen har kjørt i 4 epoker og den startet med et trenings tap på 0.272 i første epoke og gikk gradvis ned, før den endte på 0.183 etter siste epoke. Validerings tap begynte på 0.201 og endte på 0.141, som tilsier en god generaliseringsevne. Dice score begynner på 0.817 og øker til 0.883 etter siste kjørte epoke, samt en segmenterings nøyaktighet på 92.53% til 94.74%. Dette tilsier at modellen er effektiv på bildesegmentering.

AlexNet:

AlexNet modellen trente i 10 epoker, som vil si at treningsalgoritmen har gått igjennom datasettet i 10 iterasjoner for å trene modellen. Trenings tap starter på 0.434 og har en jevn nedgang gjennom hver iterasjon, der den til slutt ender på 0.210 som indikerer forbedring i modelltreningen over tid. Validerings tap startet på 0.397 og går jevnt nedover i hver iterasjon før den til slutt ender

på 0.175, som tilsier at modellen generaliserer bra uten å over-tilpasse på treningsdataen. Dice score begynner på 0.345 og øker til 0.819 som viser til en kraftig økning i segmenterings evne fra første til siste epoke. Dette betyr at modellen blir bedre og bedre til å segmentere bildene riktig. Segmenteringsnøyaktighet begynner på 82.23% og øker til 92.99% ved siste epoke, som sier at modellen blir mer nøyaktig over tid.

SqueezeNet:

SqueezeNet modellen har kjørt igjennom 4 epoker og begynte med et trenings tap på 0.263 og gikk ned til 0.204, som viser en moderat nedgang over de 4 epokene. Validerings tap begynner på 0.239 og går ned til 0.171 som er en bra nedgang. Dice score begynner på 0.716 og øker til 0.833, som er en god økning på 4 epoker. Segmenterings nøyaktighet begynner på 89.90% og øker til 93.23%, som viser til en relativt god nøyaktighet.

7.1.2 Sammenligning av kvantitative resultater

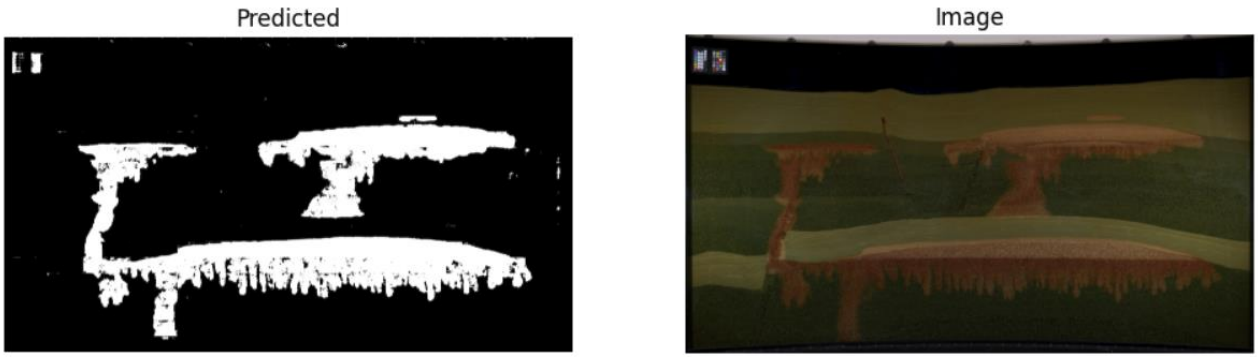
AlexNet er en av de minst komplekse modellene som er testet i dette prosjektet, som blir gjenspeilet av at den har kjørt 10 epoker på samme tid som de andre modellene har kjørt 4 eller 5 epoker. Samtidig har den generelt lavere resultater, med unntak av SqueezeNet. En kan, ut ifra de presenterte resultatene, diskutere om AlexNet er en modell som ikke egner seg like godt som de andre modellene som ble testet, til dette spesifikke datasettet. ResNet50 er den som presterer best på samtlige måleenheter. Den har lavere trening- og validering tap, høyere dice score og bedre segmentering nøyaktighet. Dette indikerer at denne modellen er den beste av de 4 modellene som er testet, med de spesifikke dataaugmenteringene som er gjort, og til denne type bilde segmentering. Samtidig, da bildene i datasettene er ganske like selv med dataaugmentering og fra forskjellige datasett, er det ikke sikkert at modellen faktisk er så god på generalisering som tallene skulle tilsi. Derfor blir det til slutt testet med nye bilder, der fargene i tanken og strukturen i sanden er helt annerledes. Dette er bilder prosjektgruppen fikk tilgang på helt i slutten av prosjektet, så her har vi ikke de samme kvantitative resultatene. Vi går derfor mer inn på dette punktet i den kvalitative analysen, hvor vi kan vise til faktiske bilder som resultater.

7.1.3 Kvalitative resultater

Her blir det gått gjennom de kvalitative resultatene, og sammenligner de forskjellige modellene sine resultater i form av bilder og om de gjenspeiler de kvantitative resultatene. Det er forskjell på hvor gode modellene er på å gi riktige prediksjoner på segmenteringene, men alle modellene gir relativt gode prediksjoner på rød/grønn bildene. Alle bildene har støy helt i toppen av bildet, der det har blitt feilaktig predikert CO2, men dette vil ikke påvirke det faktiske resultatet da denne delen av bildet kan fjernes fra alle prediksjonene. Grunnen til dette er at det aldri vil være CO2-forekomst der. En fellesnevner på alle prediksjonene som er gjort på gul/blå bildene, er at modellene er langt mindre nøyaktig på de bildene enn samme modell er på rød/grønn bildene. Dette kan være naturlig å forvente, da modellen kun har trent på rød/grønn bilder. Samtidig er en del av augmenteringene i datasettet endring av fargekanalene i bildet, så andre farger i bildet er ikke ukjent for modellen og forventningen var at de skulle prestere bedre på nye farger i testbildene. Det som er spennende med gul/blå bildene er at selv om den markerer veldig mye CO2 der det ikke er CO2, så treffer den nærmest perfekt der det faktisk er CO2.

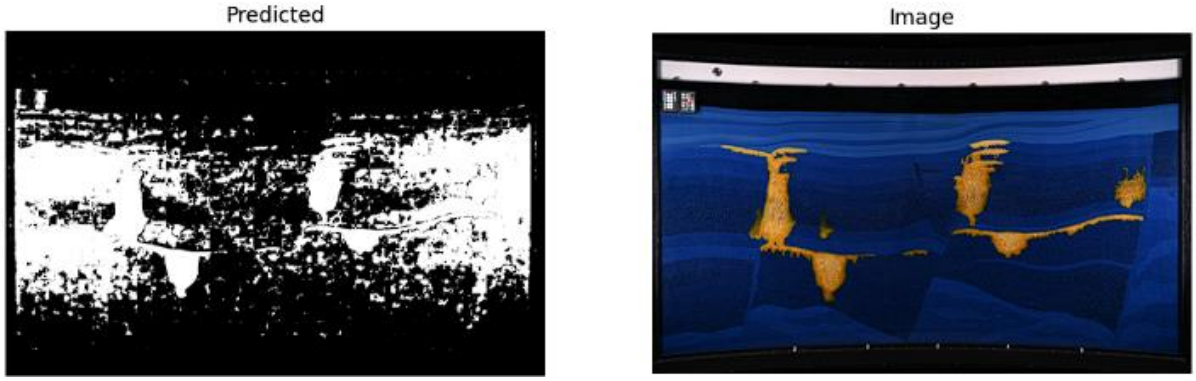
ResNet50

I Figur 7-4 nedenfor ser vi prediksjonene til ResNet50 modellen. Dette var den modellen som presterte best basert på samtlige av de kvantitative resultatene, og det gjenspeiles i prediksjonen. Figuren viser en prediksjon som treffer veldig bra på den overordnede strukturen av bildet, men med noe støy på høyre siden av bildet, samtidig som den ikke treffer hundre prosent på detaljene i CO2 strukturen. ResNet50 modellen utmerker seg i forhold til de andre modellene, med at den har mindre hull i de store partiene med CO2. De kvantitative resultatene viste at ResNet50 var den modellen med best resultater på trening av modellen med de gjeldende data augmenteringene. Prediksjonene på testbildene sier det samme; ResNet50 er den modellen som produserer de beste resultatene på dette datasettet.



Figur 7-4 - Figuren viser prediksjon gjort av ResNet50 modellen, sammen med original rød/grønn bilde den er testet på.

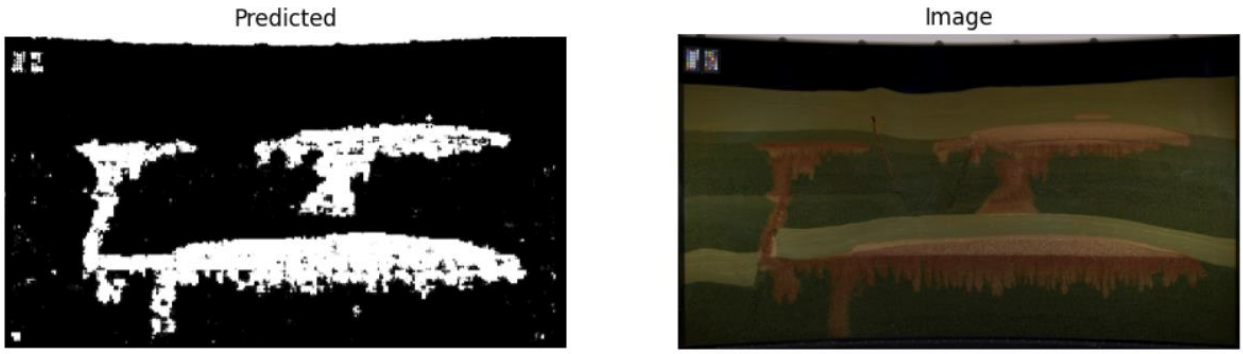
Figur 7-5 viser ResNet50 sin prediksjon på gul/blå bildet, som er vesentlig mindre nøyaktig enn prediksjonen på rød/grønn bildet. Resultatet her følger samme trenden som resultatet av AlexNet modellen, der modellen predikerer mer eller mindre helt riktig der det skal være CO2, samtidig som den predikerer CO2 veldig mange steder der det ikke skal være CO2. Det interessante her er at ResNet50, selv om det er den modellen med best resultat på rød/grønn bildene, er den som predikerer mest feil på gul/blå bildene, som kan tyde på at den over-tilpasser på treningsdataen.



Figur 7-5 - Figuren viser prediksjon gjort av ResNet50 modellen, sammen med original gul/blå bilde den er testet på.

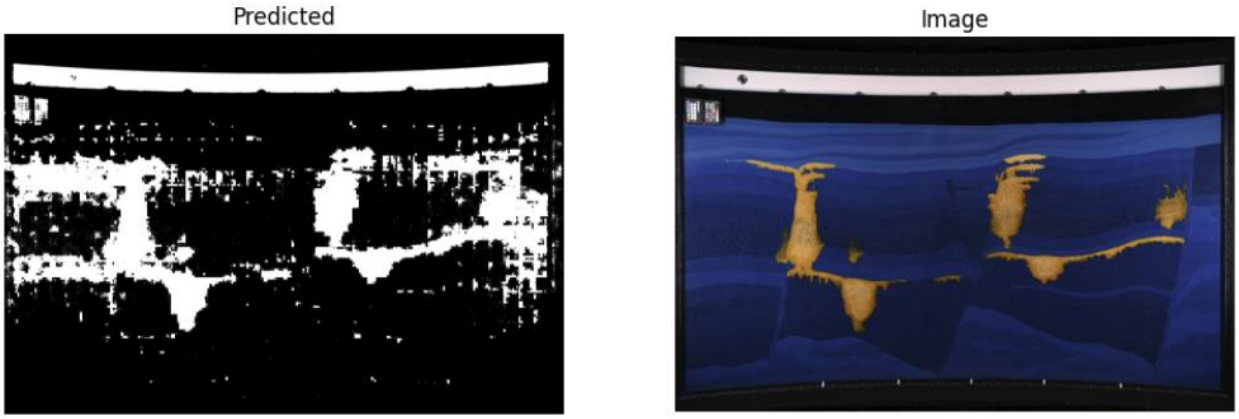
Vgg16_bn

Figur 7-6 viser prediksjonene gjort av VggNet16_bn modellen på rød/grønn bilde. Prediksjonene her passer til forventningene stilt av de kvantitative resultatene, og viser en modell som hverken er best, eller dårligst på dette spesifikke datasettet. Det er noe støy i bildet, der den viser CO2 der det ikke er CO2 i original bildet. Det er også en del detaljer i den overordnede CO2-strukturen som mangler.



Figur 7-6 - Figuren viser prediksjon gjort av VggNet16_bn modellen, sammen med original rød/grønn bilde den er testet på.

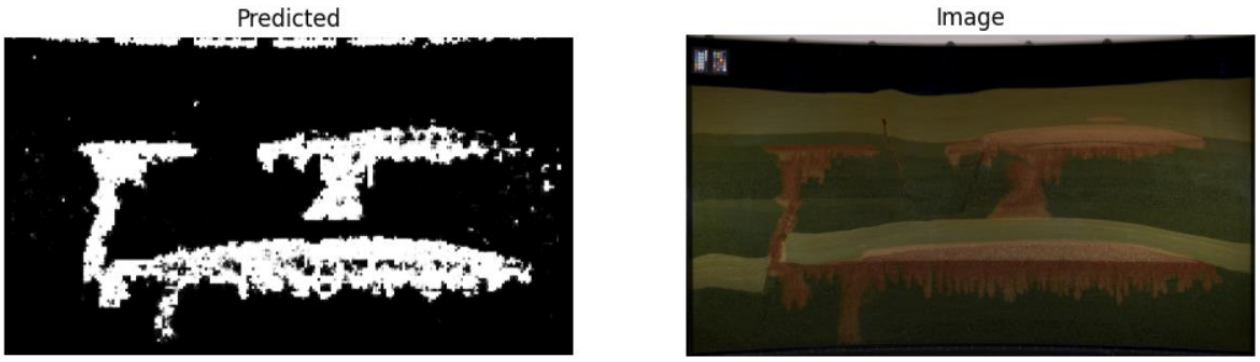
Figur 7-7 viser prediksjonen modellen har gjort på gul/blå bildet, og resultatet her har spennende implikasjoner. VggNet16_bn er en modell som gjør det dårligere enn ResNet50 på samtlige kvantitative resultater, men samtidig presterer bedre på dette bildet enn ResNet50. Dette kan være flere grunner til, men en mulig forklaring kan være at et godt resultat på rød/grønn bildene impliserer at modellen over-tilpasser på datasettet da det har for lite variasjon.



Figur 7-7 - Figuren viser prediksjon gjort av VggNet16_bn modellen, sammen med original gul/blå bilde den er testet på.

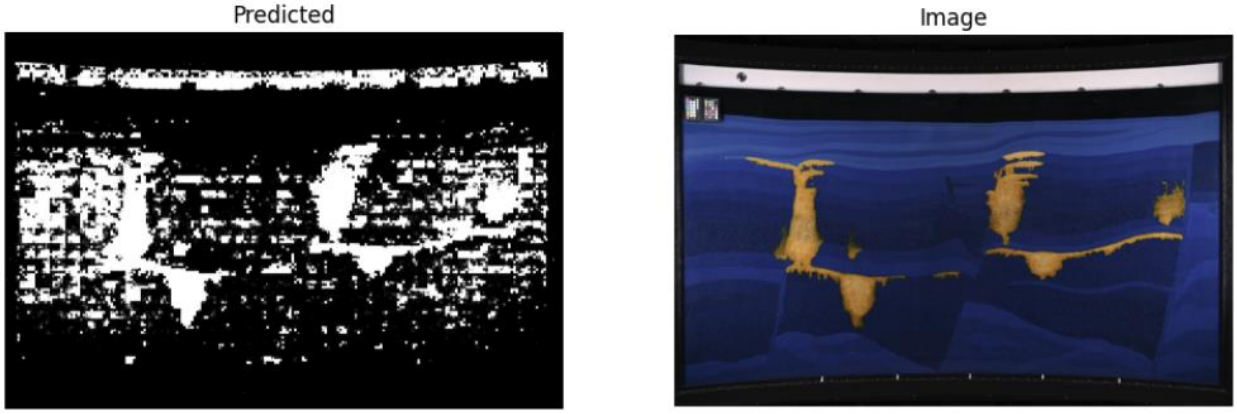
AlexNet

Nedenfor viser Figur 7-8 resultatet av prediksjonen til AlexNet, og bildet den ble testet på. Prediksjonen viser at modellen finner den overordnede strukturen relativt bra, men at den bommer på en del av detaljene i bildet. Det er også litt "støy" i bildet, der den markerer CO2 på steder der det ikke skal være CO2. Her passer det kvalitative resultatet i form av bilde-prediksjon med de kvantitative resultatene som ble presentert i 7.1. Resultatene som ble presentert viste en modell med relativt gode resultater, men ikke helt optimalt i forhold til noen av de andre modellene.



Figur 7-8 - Figuren viser prediksjon gjort av AlexNet modellen, sammen med original rød/grønn bilde den er testet på.

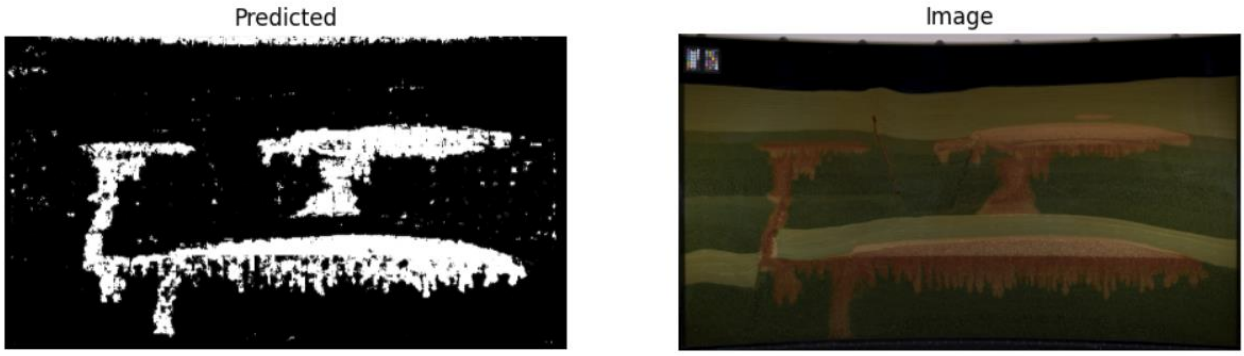
Figur 7-9 viser modellen sin prediksjon på gul/blå bilder som den ikke har trent på. Her er resultatet ganske annerledes enn på rød/grønn bildene som ble vist i Figur 7-2. Modellen predikerer CO2 veldig mange steder der det ikke er CO2 i originalbildet. Samtidig, så treffer den veldig bra der det faktisk er CO2. Den får med den overordnete CO2 strukturen, og treffer bra på detaljene.



Figur 7-9 - Figuren viser prediksjon gjort av AlexNet modellen, sammen med original gul/blå bilde den er testet på

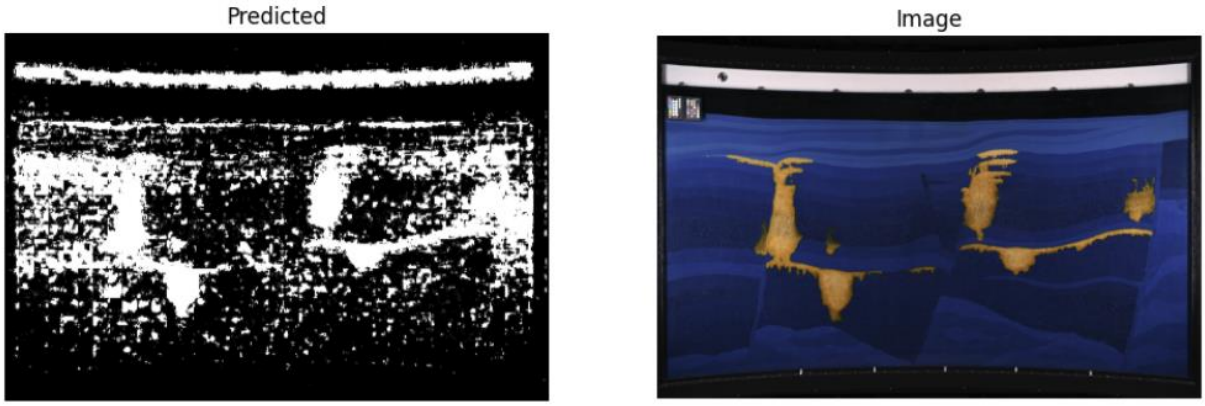
SqueezeNet

Figur 7-10 viser resultatet av SqueezeNet modellen sin prediksjon på testbildet. Som en kan se på bildene, er resultatet av denne modellen dårligere enn ResNet50. Den predikerer CO2 flere steder der det ikke er CO2, og de store feltene med CO2 er fulle av hull. Resultatet er tilsvarende AlexNet modellen sitt resultat.



Figur 7-10 - Figuren viser prediksjon gjort av SqueezeNet modellen, sammen med original rød/grønn bilde den er testet på.

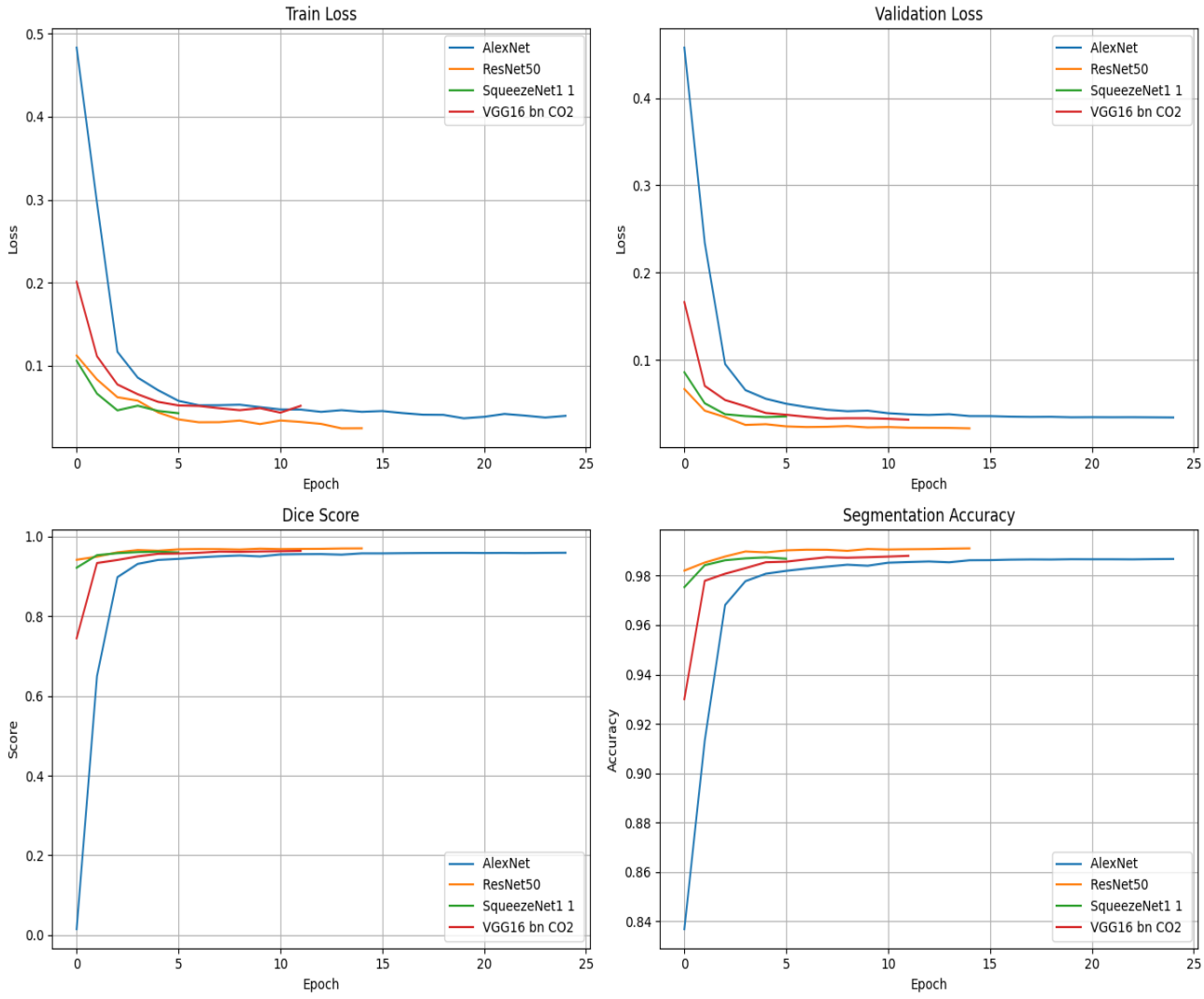
Figur 7-11 viser at SqueezeNet modellen, som de tidligere nevnte modellene, treffer bra der det er CO2 men predikerer CO2 veldig mange steder der det ikke er CO2.



Figur 7-11 - Figuren viser prediksjon gjort av SqueezeNet modellen, sammen med original gul/blå bilde den er testet på.

7.2 Sammenligning av resultater uten augmentering

Her blir det gått gjennom resultatene til de ulike modellene som er trent på datasett uten dataaugmentering. Dette innebærer både kvantitative resultater, der grafene og tabellene i kapittel 6.3 blir analysert og diskutert. Det presenteres også kvalitative resultater i form av prediksjoner gjort av de forskjellige modellene. Det er også her testet med fire forskjellige modeller; AlexNet, ResNet50, SqueezeNet og Vgg16_bn. Figur 7-12 nedenfor viser sammenligning av resultatene til modellene gjennom treningsepokene, for alle evalueringsmetodene.



Figur 7-12 - Figuren viser resultater for AlexNet, ResNet50, SqueezeNet1_1, VGG16_bn etter trening på datasett uten augmentering.

Hvorfor teste modeller uten data augmentering?

Som en del av prosjektet, ble det utdelt et datasett med bilder av CO2 for å trene opp maskinlæringsmodeller. Det var i utgangspunktet kun rød/grønn bilder i dette datasettet, der gruppen fikk vite at det kom til å komme flere bilder i andre farger senere. Den opprinnelige ideen var dermed å bruke augmenteringer for å kunne øke størrelsen på datasettet, og for å samtidig gjøre at modellen skulle kunne generalisere seg til fremtidige fargeendringer i datasettet.

Resultatene fra kapittel 7.1 viser at disse modellene greide å finne den overordnede CO2-strukturen i både rød/grønn bildene og gul/blå bildene. Prediksjonene gjort på gul/blå bildene

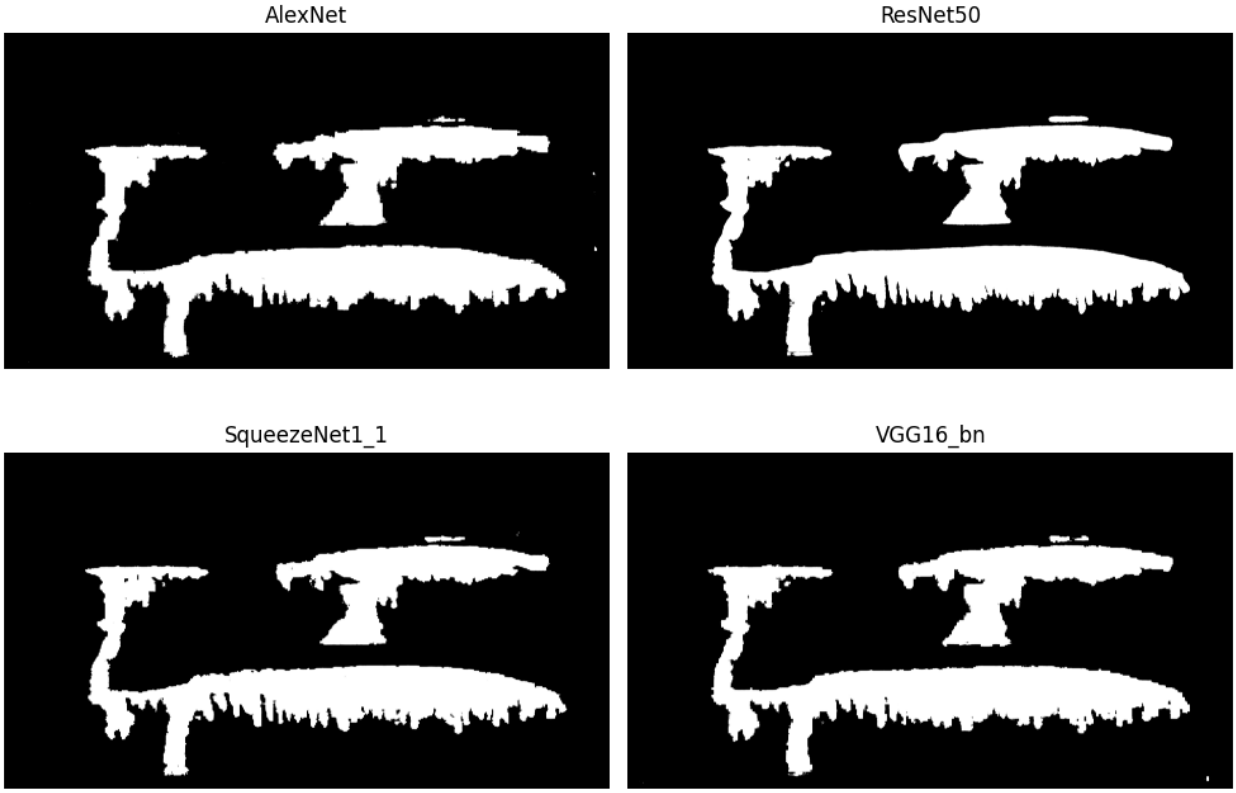
inneholder derimot også mye støy, som vil si at den feilaktig detekterer CO2 der det ikke skal være CO2, som igjen fører til at de ikke kan brukes til videre matematiske beregninger.

I et forsøk på å se på mulige forbedringer for videre arbeid, ble modellene trent på bilder uten augmenteringer. Resultatet fra disse viste seg da å være eksepsjonelt gode, ikke bare for de opprinnelige rød/grønn bildene, men også for de nye gul/blå bildene. Det at modellene var bedre på rød/grønn bildene var ikke uventet, da dette er de eneste bildene modellene har sett. Men det at modellene greide å identifisere CO2 i bilder med andre farger enn det de har blitt trent på, med vesentlig bedre resultat enn modellene som er trent på datasett med fargeaugmenteringer, var en veldig stor overraskelse.

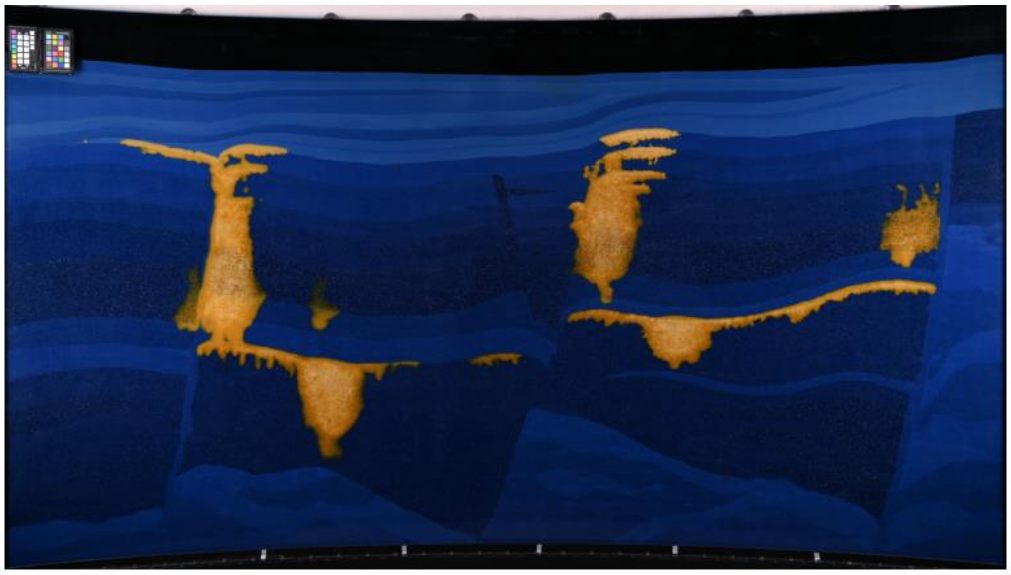
Resultater

Her blir det gått gjennom resultatet, og en sammenligning av prediksjonene til de forskjellige modellene. Først bildene som er rød/grønn, og deretter de gul/blå bildene. Figur 7-13 viser originalt rød/grønn bildet, og Figur 7-14 viser resultatet av prediksjoner gjort på dette bildet av de ulike modellene. Figur 7-15 viser originalt gult/blått bilde, og Figur 7-16 viser resultatene av prediksjonene.

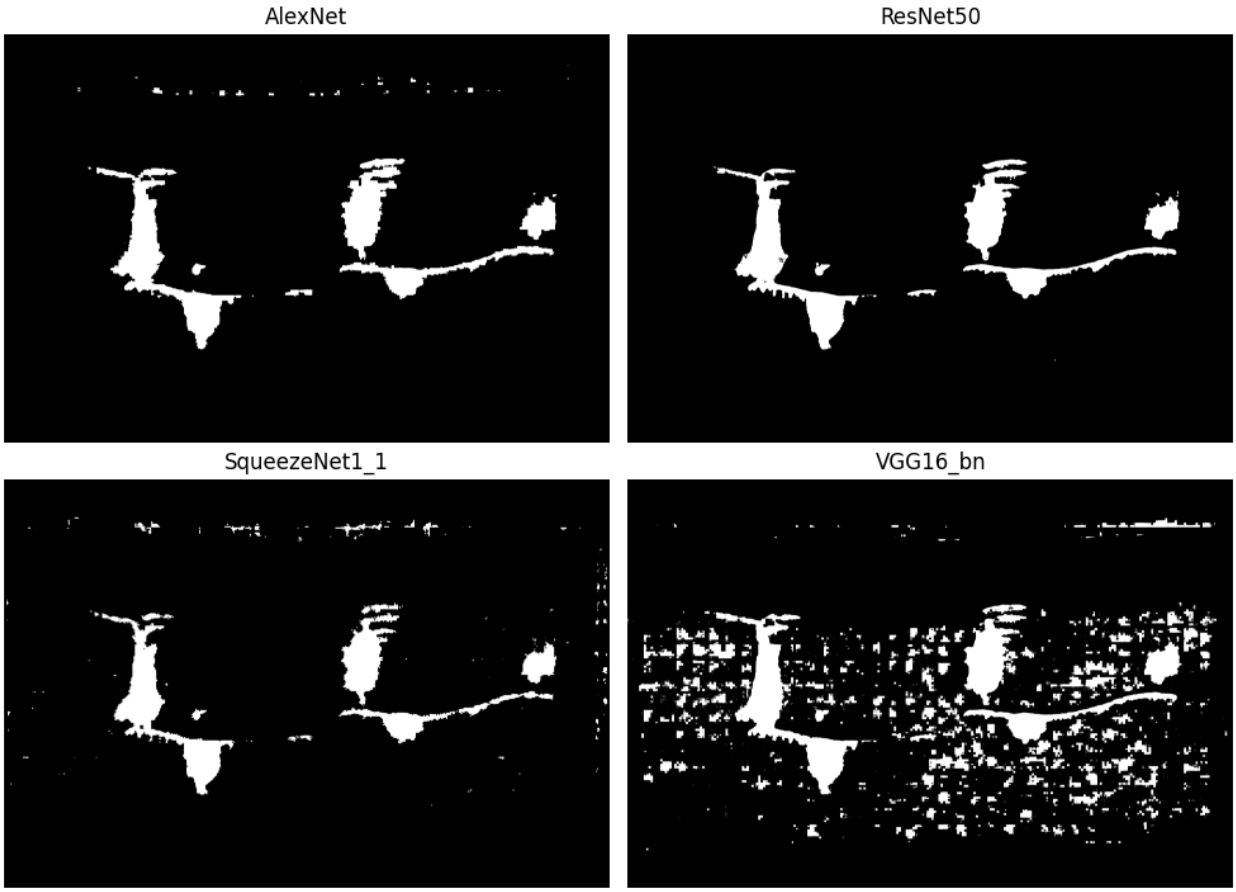
Figur 7-13 - Rød/grønn bilde som brukes til å teste modeller som er trent på datasett uten dataaugmenteringer



Figur 7-14 - Prediksjoner gjort på rød/grønn bilde av modeller trent på datasett uten augmenteringer



Figur 7-15 - Gul/blå bilde som brukes til å teste modeller som er trent på datasett uten dataaugmenteringer



Figur 7-16 - Prediksjoner på gul/blå bilde gjort av modeller på datasett uten augmentering

Felles for alle modellene er at de har fått gode resultater når de predikerer på de rød/grønne bildene. Alle har predikert riktig på hvor CO2 er, uten å ta med noe unødvendig støy. Dette blir også gjenspeilet i grafene i Figur 7-12, der resultatene til modellene som trente på datasettet uten augmenteringer blir presentert. På denne grafen kan en se at alle får veldig gode resultater på både Dice og nøyaktighet, samt at de presterer bra på trenings tap og validerings tap.
Dette er riktignok ikke så overraskende at disse modellene får gode resultater på treningen, da treningsbildene er sammensatt av kun rød/grønne bilder.

Datasettet uten augmenteringer er mindre enn det augmenterte datasettet. I det ordinære datasettet er det 18 bilder før bildet blir segmentert til mindre deler, der det augmenterte har 54 bilder før segmentering. Det augmenterte datasettet er også i forskjellige farger, der 9 av disse er rød/grønne. Derfor vil en modell trent på dette datasettet, få trent flere ganger på de samme fargekombinasjonene.

Prediksjonene på gul/blå bildene viser til et godt resultat, da alle modellene har korrekt identifisert, samt fylt ut områdene med CO2. VGG16_bn har tatt med en del støy i predikeringen sin, samtidig er den fortsatt bedre enn de andre modellene som er trent på det augmenterte datasettet.

7.3 Sammenligning av forventninger og teori

Tidlig i prosjektet ble modellene trent på et datasett bestående bilder, med tilhørende masker laget av algoritmeløsning som ble nevnt i 4.1.1. Her var forventningen at maskene produsert av denne løsningen skulle være tilstrekkelig for videre trening av modellene. Dette var ikke tilfelle, da maskene som ble produsert ikke var av god nok kvalitet for å oppnå et tilfredsstillende resultat.

Ved videre testing av modellen ble det produsert relativt gode resultater, men antagelsen var at modellen gjenkjente fargen i CO2-strukturen, og ikke strukturen i seg selv. Som følge av dette ble det bestemt å gjøre fargeaugmenteringer for at modellen skulle kunne generalisere seg godt nok til fremtidige fargeendringer.

I slutten av prosjektet ble fargene i FluidFlower-tanken endret, og vi fikk tilsendt nye bilder vi kunne teste modellene på. Derfor var overraskelsen stor da resultatene på gul/blå bildene var såpass mye dårligere enn rød/grønn bildene. Antagelsen var at modellene skulle kunne håndtere de nye bildene grunnet at de var trent på datasett med fargeaugmenteringer. Den største overraskelsen

kom midlertidig da vi testet modellene på datasett uten noen augmenteringer. Forventningene her var at modellene kom til å prestere enda dårligere enn de som var trent på augmentert datasett, da de aldri har sett andre fargevariasjoner enn rød/grønn. Dette stemte ikke i det hele tatt, da samtlige av modellene presterer over all forventning på både rød/grønn og gul/blå bildene.

En mulig årsak til at modellene som var trent på datasett med augmenteringer, presterte dårligere på både rød/grønn og gul/blå bilder enn modellene som er trent på datasett uten augmenteringer, er at augmenteringene ble for komplekse. Augmenteringene som var gjort på datasettet har mulig overkomplisert datasettet i en så stor grad at modellene ikke har greid å finne en sammenheng i bildene. En mulig løsning på dette hadde vært å ha en mindre andel augmenterte bilder, slik at det hadde vært langt flere bilder uten augmentering, enn med.

En annen mulig årsak til at modellene trent på datasett med augmentering presterte dårligere, er at Color Jitter augmentering kan ha påvirket resultatet negativt. Dette var en augmentering som var forventet å forbedre modellens evne til å generalisere til nye farger. I Figur 7-3 kan en se spesielt på det røde bildet nede til venstre i figuren, at det er flere steder i bildet der det har kommet med rødfarge utenfor CO2-strukturen. Dette kan være en mulig forklaring på hvorfor gul/blå bildene treffer bra på CO2-strukturen, men også marker CO2 flere steder der det ikke skal være CO2.

Det er mulig at alle modellene er over-tilpasset på treningsdataen, men selv om dette vanligvis er negativt, er ikke det nødvendigvis tilfelle her. Grunnen til det er at det ikke skal være store variasjoner i bildene den skal analysere, da den overordnede CO2 strukturen skal være tilnærmet lik på alle bilder.

7.4 Begrensninger og utfordringer

En begrensning som påvirket effektiviteten og resultatet til prosjektet, er manglende kunnskap om dyplæring og bildesegmentering fra tidligere, så nødvendig kunnskap har blitt tilegnet i løpet av prosjektet. Dette har gjort at det har blitt gjort en del feil og unødvendig arbeid, som kunne vært unngått dersom gruppen hadde mer kunnskap innenfor dyplæring.

En klar teknisk begrensning som gjorde at prosjektet krevde mer tid enn nødvendig, samt gjorde arbeidet mindre effektivt, er begrensningene på maskinvare. Prosjektet krevde mer og mer ytelse etter hvert som modellen ble større og mer kompleks, mens maskinvaren som var tilgjengelig ikke var egnet til å kjøre så omfattende modeller. Dette gjorde at prosjektet ble kjørt via Kaggle, som er en side som tilbyr gratis bruk av såkalte notebooks. Kaggle har begrensninger på hvor mye

maskinkraft som kan bli brukt på en gang, og i løpet av en uke. Når modellen ble såpass kompleks at den trengte i overkant av 12 timer å kjøre gjennom en læringssyklus, var ikke dette mulig å gjøre på en enkel måte via Kaggle da det de har en begrensning på maks 12 timer om gangen. Dette gjorde at det ble satt en tidsbegrensning på hvor lenge hver modell kunne kjøre, som gjorde at noen modeller kun fikk kjørt noen få epoker. Om de hadde kunne kjørt lengre, hadde muligens resultatet blitt bedre. Derimot bidro tidsbegrensninger at det ble valgt enklere modeller som eksempelvis AlexNet, som viste seg å være tidseffektiv og lite krevende i form av maskinkraft. AlexNet produserte også tilsvarende like gode resultater som de mest krevende modellene. Dette er noe som muligens ikke hadde blitt oppdaget dersom det ikke var begrensninger på tid og maskinkraft.

Siden prosjektet allerede hadde begrenset tid, og det ikke var mye tid til å lære seg nye teknologier, har dette også bidratt til at en enklere løsning som fast.ai blir valgt fremfor mulig bedre løsninger som PyTorch. Samtidig ble det produsert et resultat som oppfylte alle krav med løsning laget i fast.ai. Dette er mulig noe som ikke hadde vært oppdaget dersom det hadde blitt valgt PyTorch, som kunne ført til en mer komplisert prosess uten forbedring av resultat.

Da det ikke var tilgjengelig bilder med andre farger før i slutten av prosjektet, ble det oppdaget sent at valgte augmenteringer forverret resultatene til modellene. Hadde dette blitt oppdaget tidligere, kunne fremgangsmåten på augmenteringene blitt endret før, som kunne skapt et mer variert og generalisert datasett.

7.5 Sluttevaluering

Problemstillingen til prosjektet er beskrevet som et nettverk som det kan mates inn bilde i RGB format, for å så kunne returnere et nytt bilde som kun har verdiene 1 og 0, der forekomst av CO2 har verdien 1 og fravær av CO2 har verdien 0. Det ble også lagt frem et ønske om å på sikt kunne få ut en verdi som viser til en gitt sannsynlighet for at det er CO2 i den spesifikke pikselen. Målet for oppdragsgiver er å kunne bruke et slikt segmentert bilde til å lettere kunne gjøre videre analyser på bildet

7.5.1 Oppfyllelse av krav og forventninger

Det endelige produktet oppfyller de grunnleggende kravene satt av oppdragsgiver, da modellene som har blitt trent kan identifisere CO2 og returnere en sann/usann representasjon av det opprinnelige bildet som er matet inn. Modellene fjerner også støy i form av sandkorn. Det er

derimot ikke laget funksjonalitet som tilsier hvor stor sannsynlighet det er for at det er CO2 på en gitt piksel, resultatet er kun en sann/usann representasjon av CO2. Det ble heller ikke laget en modell som kan gjenkjenne sandlagene i FluidFlower tanken, noe som skulle blitt gjort dersom det var tilstrekkelig med tid. Oppdragsgiver har gitt tilbakemelding på at endelig resultat er godt nok til at de kan bruke det videre.

7.5.2 Styrker med prosjektet

Styrkene med dette prosjektet er at det er blitt laget flere modeller som har veldig gode resultater på segmentering og predikering av CO2 i bildene fra FluidFlower-tanken. Dette gjør at resultatet fra oppgaven kan brukes videre på forskning og matematiske beregninger.

7.5.3 Svakheter med prosjektet

Svakhetene med prosjektet er at modellene kan være over-tilpasset til datasettet, så dersom FluidFlower tanken endres drastisk, eller det bygges en ny, så vil mest sannsynligvis modellene trenes på nytt. Det skulle blitt testet en og en augmentering på datasettet, for å se effekten av hver enkelt augmentering, for å øke variasjonen i datasettet uten at det påvirker resultatet negativt. Som tidligere nevnt i diskusjonen, har augmenteringer som Color Jitter mulig forverret resultatet, og ikke gjort det bedre, som vanligvis er formålet med å augmentere bilder i et datasett. Samtidig er det ikke laget et grensesnitt for å laste opp bilder enkelt. Nå må oppdragsgiver laste opp bildene som skal testes i datasett via Kaggle. Det å lage et grensesnitt var ikke noe oppdragsgiver forventet, men det er noe som hadde gjort videre testing av modell lettere.

7.5.4 Potensielle forbedringer i resultatet

Dersom prosjektet hadde vært startet på nytt, er det et par endringer som ville vært gjort. Som en kan se ut ifra diskusjonen har effekten av dataaugmentering vært dårlig. Det skulle tidligere vært sjekket påvirkningen av hver enkel augmentasjon for å se hvilken effekt den hadde på generaliseringen av modellen. Det skulle også vært laget noen testbilder med andre farger, for å kunne teste generaliseringen. Det skulle dermed ikke vært nødvendig å vente på gul/blå bildene fra oppdragsgiver for å teste dette. Dette hadde gjort at det tidligere kunne blitt jobbet med deteksjon av sandlag og porøsitet, samt det kunne blitt bedre tid til å lage et eventuelt brukergrensesnitt.

7.6 Helhetlig perspektiv

I forskningsprosjekter kan det være lett å se for mye på detaljene i prosjektet, uten å vurdere mulige konsekvenser av produsert resultat. Derimot er det ofte etiske overveielser som må gjøres, samt at påvirkningen av prosjektet på samfunnsmessige og økonomiske aspekter burde vurderes, ut ifra et helhetlig perspektiv.

7.6.1 Etiske overveielser

Da det å fjerne CO2 fra atmosfæren og lagre det under bakken i hovedsak er en positiv handling, har det ikke vært mange etiske problemstillinger gruppen har måttet ta hensyn til under arbeidet med oppgaven. Samtidig kan det være etiske dilemmaer som må overveies fra et helhetlig perspektiv, som oppdragsgiver burde ta hensyn til ved videre arbeid. Et mulig etisk dilemma vil være at det kan bidra til å "grønnvaske" oljeindustrien, da oljeselskapene kan bruke fremtidig CO2-lagring som en unnskyldning for å kunne pumpe opp mer olje. FN's generalsekretær kritiserte også olje- og gasselskaper sine forsøk på å senke utslippene forbundet med fossile brennstoff ved å bruke karbonlagring, kontra det å begrense bruken av fossile brennstoff i seg selv (Florence, 2023). Komplekse og store maskinlæringsmodeller kan også være ekstremt energikrevende, og veksten innenfor AI krever allerede store mengder energi. Forskning viser at dersom veksten innenfor AI fortsetter i samme hastighet som det gjør nå, vil energiforbruket for AI alene være på mellom 85-134 terrawatt timer per år innen 2027 (Vincent, 2024).

7.6.2 Samfunnsmessig og økonomisk påvirkning

Tidligere forskning der det er brukt maskinlæring innen karbonlagring har blitt basert på seismisk data som er fanget opp fra oljereservoarer. Her er det ikke samme muligheter for å verifisere at antagelser som er gjort, er riktig og at dataen de får faktisk er korrekt i henhold til virkeligheten. Med FluidFlower-tanken er det mulig å se hvordan CO2 oppfører seg i simulerte geologiske omgivelser, som gjør at man enkelt kan verifisere om antagelser stemmer. Med en maskinlæringsmodell som kan fortelle hvor CO2 er og ikke er, kan forskningsteamet gjøre videre matematiske beregninger basert på hvordan CO2 oppfører seg, og slippe å regne ut for hvert enkelt bilde hvor det er CO2. Modellen vil kunne forenkle hverdagen til forskningsteamet da arbeid som tidligere har vært manuelt, vil nå være automatisk. Dette prosjektet kan dermed hjelpe til videre fremgang innen forskning på karbonlagring, som vil kunne ha store implikasjoner på hvordan CO2-utslipp blir håndtert.

Et vellykket karbonlagringsprosjekt kan ha store samfunnsmessige og økonomiske påvirkninger, da det muligens kan endre måten vi tenker på CO2 utslipp. Dersom vi kan fange opp CO2 før vi slipper det ut i atmosfæren vil det kunne føre til nullutslipp flere steder som ellers ville sluppet ut store mengder CO2. Klimaendringer har også store økonomiske konsekvenser for den globale økonomien, og forskning anslår at klimaendringene vil koste oss mellom 1.7 og 3.1 trillion dollar innen 2050. Samtidig vil et vellykket prosjekt kunne bidra til å begrense klimaendringene, og samtidig hjelpe oss å nå FNs mål som blant annet er at klimagassutslipp blir netto-null innen 2050 (FN, u.å.).

Karbonlagring vil spesielt være nyttig i land som har kullbasert energi, da utslippet som er skapt av å brenne kull kan fanges opp, komprimeres og sendes via rør til underjordiske lagrings depoter (British Geological Survey, u.å.).

8. Konklusjon

Bachelorprosjektet hadde som mål å effektivisere og automatisere analysen av CO2 forekomst i bilder tatt av FluidFlower-tanken til UiB, da det blir produsert tusenvis av bilder som må analyseres. Formålet med dette er at oppdragsgiver kan bruke resultatet av analysen til å gjøre videre matematiske beregninger for å bedømme hvordan CO2 oppfører seg i geologiske formasjoner. Målet med oppgaven var dermed følgende:

I prosjektet skal det utnyttes maskinlæringsalgoritmer for å automatisk identifisere hvor CO2 befinner seg i bildene tatt av FluidFlower-tanken.

Det ble først trent modeller på augmentert datasett, men resultatet på andre fargekombinasjoner viste seg å være utilstrekkelig i forhold til kravene satt av oppdragsgiver. Som følge av dette ble modellene trent på et datasett uten augmenteringer, og resultatet her var vesentlig bedre. Gjennom videre analyse av de kvalitative og kvantitative resultatene fra trening av modellene, ble det bestemt at modellene som var trent på datasett uten augmenteringer var vesentlig bedre på å predikere CO2 forekomst i rød/grønn bilder, men også å generalisere til nye farger som gul/blå bildene.

Disse resultatene kan indikere at augmenteringene som er utført har gjort at modellen har lært seg uønskede mønstre og har plukket opp støy som ikke nødvendigvis kan eksistere i et reelt scenario.

Prosjektet kan dermed demonstrere at valg av dataforberedelse og databehandling kan ha store konsekvenser for maskinlæringsmodellens ytelse. Augmentering av datasett kan, og ofte er, fordelaktig for å øke generaliseringsevnen til modellen. Samtidig viser resultatene av dette prosjektet at kvalitet i datasettet kan være viktigere enn kvantitet. Disse funnene er ikke bare relevante for dette prosjektet, men har også bredere implikasjoner for andre prosjekter innen maskinlæring og dyplæring.

Avslutningsvis konkluderes det fra både prosjektgruppen og oppdragsgiver, at resultatene av dette prosjektet oppfyller de stilte kravene til en ønsket løsning.

8.1 Anbefalinger for fremtidige prosjekter

Modellene som er utarbeidet og trent i dette prosjektet kan ikke brukes direkte av andre enn oppdragsgiver, da modellene kun er trent på FluidFlower tanken som står på UiB. Det som sannsynligvis kan brukes av andre lignende prosjekter, både innen karbonlagring og andre

lignende bilde segmenteringsoppgaver er fremgangsmåten i prosjektet og hvordan modellene har blitt trent og hvordan treningsdata har blitt laget.

En anbefaling til grupper eller personer som skal jobbe med bildesegmentering generelt, er å vurdere hvilke, om noen, augmenteringer som blir gjort basert på kompleksiteten på datasettet. I prosjekter der det er stor variasjon i bildene og eksempelvis lysstyrke, kvalitet og format kan endre seg fra bilde til bilde, er det nok viktigere med mer komplekse og flere augmenteringer. I motsetning til for eksempel dette prosjektet, der det alltid er samme vinkel på bildene, samme kvalitet, og generelt veldig lite variasjoner, så kan det være fordelaktig med mindre augmenteringer. Det samme gjelder valg av modell, der det ikke alltid er nødvendig med de mest komplekse og største modellene om det ikke jobbes med veldig komplekse datasett.

8.2 Forslag til videre arbeid

Dersom det skal jobbes videre med dette prosjektet i ettertid, er det anbefalt å utbedre et par aspekter med løsningen som er presentert i dette prosjektet. Det vil være å anbefale å lage et brukergrensesnitt der bilder tatt av FluidFlower-tanken automatisk lastes opp til modellen, slik at den kan returnere en prediksjon direkte. Det vil også være hensiktsmessig å justere modellen slik at den kan returnere konsentrasjon av CO2 på gitte steder i bildet, og ikke bare sant/usant prediksjoner. For å predikere sandlag og porøsitet vil det anbefales å trene en ny modell, eller utvide trening på eksisterende modell.

Optimalt sett burde det blitt laget et nytt datasett med forskjellige farger i FluidFlower-tanken, slik at modellen kan trene på flere mulige variasjoner og sikre en god nok generalisering. Derimot er det naturlig å tro, basert på resultatet av dette prosjektet, at modellen burde kunne håndtere endringer i tanken, da den overordnete strukturen i utgangspunktet ikke skal endre seg nevneverdig med tiden.

ResNet50 var den modellen som presterte best på predikering av CO2. Derimot betyr ikke det at det nødvendigvis er den modellen som anbefales å bruke videre, da alle modellene ga tilnærmet like gode resultater. ResNet50 krever langt mer ressurser og lagringskapasitet å trene enn eksempelvis AlexNet. Dette gjør at AlexNet tar vesentlig kortere tid å trene enn ResNet50. Så her burde det vurderes i eventuelt videre arbeid om de minimale forbedringene i ResNet50 modellen er verdt det på bekostning av maskinkraften det tar å kjøre den.

9. Referanser

Alqahtani, A., Xupeng H., Bicheng, Y. & Hussein H., (2023). *Uncertainty Analysis of CO_2 Storage in Deep Saline Aquifers Using Machine Learning and Bayesian Optimization.* Tilgjengelig fra: https://dx.doi.org/10.3390/en16041684 (Hentet: 14.04.2024)

Audagnotto, M., Czechtizky, W., De Maria, L. *et al. Machine learning/molecular dynamic protein structure prediction approach to investigate the protein conformational ensemble* (2022). Tilgjengelig fra: https://doi.org/10.1038/s41598-022-13714-z (Hentet: 25.04.2024)

Awan, A. (2022) *A Complete Guide to Data Augmentation.* Tilgjengelig fra: https://www.datacamp.com/tutorial/complete-guide-data-augmentation (Hentet: 05.03.2024).

Baeldung (2024) *Training and Validation Loss in Deep Learning*. Tilgjengelig fra: https://www.baeldung.com/cs/training-validation-loss-deep-learning (Hentet: 20.04.2024).

Berchane, N(2018) *artificial intelligence, machine learning, deep learning [digital illustration]* Tilgjengelig fra: https://master-iesc-angers.com/artificial-intelligence-machine-learning-and-deep-learning-same-context-different-concepts/ (Hentet: 04.04.2024)

British Geological Survey (u.å) *Understanding carbon capture and storage*. Tilgjengelig fra: https://www.bgs.ac.uk/discovering-geology/climate-change/carbon-capture-and-storage/ (Hentet: 07.05.2024).

Chhikara, P. (2022) *Understanding Morphological Image Processing and Its Operations*. Tilgjengelig fra: https://towardsdatascience.com/understanding-morphological-image-processing-and-its-operations-7bcf1ed11756 (Hentet: 05.04.2024).

CIOpages (2023) *Solving Unstructured Data Challenges with AI and ML.* Tilgjengelig fra: https://www.ciopages.com/solving-unstructured-data-challenges-with-ai-and-ml/. (Hentet: 23.04.24)

Datagen (u.å) *ResNet-50: The Basics and a Quick Tutorial*. Tilgjengelig fra: https://datagen.tech/guides/computer-vision/resnet-50/ (Hentet: 10.03.2024).

Datascientest (2023) *U-NET: Computer Vision's neural network*. Tilgjengelig fra: https://datascientest.com/en/u-net-computer-visions-neural-network (Hentet: 20.04.2024).

Hotz, N. (2024) *What is CRISP DM?*. Tilgjengelig fra: https://www.datascience-pm.com/crisp-dm-2/ (Hentet: 01.05.2024).

Dvergsdal, H. (u.å) *Python.* Tilgjengelig fra: https://snl.no/Python_-_programmeringsspr%C3%A5k (Hentet: 20.02.2024).

Fast.ai (u.å a) *Vision.Models*. Tilgjengelig fra: https://fastai1.fast.ai/vision.models.html (Hentet: 13.03.2024).

Fast.ai (u.å b) W*elcome to fastai*. Tilgjengelig fra: https://docs.fast.ai/ (Hentet: 20.02.2024).

Fernø, M., Haugen, M., Eikehaug, K., Folkvord, O., Parlius, O., Benali, B., Both, J., Storvik, E., *et al* (2024) *Room-Scale CO2 Injections in a Physical Reservoir Model with Faults*. Tilgjengelig fra: https://bora.uib.no/bora-xmlui/handle/11250/3095493 (Hentet: 03.03.2024). (Hentet: 03.03.2024).

Florence, J. (2023) *UN secretary-general calls out carbon capture as greenwashing*. Tilgjengelig fra: https://www.offshore-technology.com/news/un-secretary-general-calls-out-carbon-capture-as-greenwashing/ (Hentet: 08.05.2024.).

FN (u.å) *FNs bærekraftsmål*. Tilgjengelig fra: https://fn.no/om-fn/fns-baerekraftsmaal (Hentet:28.04.24).

GeeksForGeeks (2024) *OpenCV Tutorial in Python*. Tilgjengelig fra: https://www.geeksforgeeks.org/opencv-python-tutorial/ (Hentet: 11.05.2024).

Gibson, P.B., Chapman, W.E., Altinok, A. *et al.* (2021), *Training machine learning models on climate model output yields skillful interpretable seasonal precipitation forecasts.* Tilgjengelig fra: https://doi.org/10.1038/s43247-021-00225-4 (Hentet: 28.04.2024)

GoodDay(u.å) *GoodDay*. Tilgjengelig fra: https://www.goodday.work/ (Hentet: 16.01.2024).

Grøn, Ø (2022). *Superkritisk fluid.* Tilgjengelig fra: https://snl.no/superkritisk_fluid. (Hentet: 21.02.24).

Hanlin Sheng, Xinming Wu, Xiaoming Sun, Long Wu (2023) *Deep learning for characterizing CO2 migration in time-lapse seismic images* Tilgjengelig fra: https://www.sciencedirect.com/science/article/pii/S0016236122036304 (Hentet: 24.02.2024).

Howard, J. (2017) *Introducing Pytorch for fast.ai*. Tilgjengelig fra: https://www.fast.ai/posts/2017-09-08-introducing-pytorch-for-fastai.html (Hentet: 28.02.2024)

IBM (u.å. a) *What is Artificial Intelligence (AI)?* Tilgjengelig fra: https://www.ibm.com/topics/artificial-intelligence (Hentet: 14.04.24)

IBM (u.å. b) *What is Machine Learning (ML)?* Tilgjengelig fra: https://www.ibm.com/topics/machine-learning (Hentet: 15.04.24)

IBM (u.å. c) *What is Deep Learning?* Tilgjengelig fra: https://www.ibm.com/topics/deep-learning (Hentet: 15.04.24)

IBM (u.å. d) *CRISP-DM Help Overview*. Tilgjengelig fra: https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview (Hentet: 07.05.2024).

IBM (u.å. e) *What is computer vision?*. Tilgjengelig fra: https://www.ibm.com/topics/computer-vision (Hentet: 07.04.2024).

IITK(2023), Machine Learning in Computer Vision and Image Processing. Tilgjengelig fra: https://ifacet.iitk.ac.in/knowledge-hub/machine-learning/machine-learning-in-computer-vision-and-image-processing/ Hentet(23.04.24)

InfoSysBPM (u.å) *Quality of datasets defines the quality of AI projects*. Tilgjengelig fra: https://www.infosysbpm.com/blogs/annotation-services/quality-of-datasets-defines-the-quality-of-ai-projects.html (Hentet: 14.04.2024)

LabelStudio (u.å) *LabelStudio.* Tilgjengelig fra: https://labelstud.io/ (Hentet: 10.04.2024).

Lutkhevich, B (u.å) 3 V's (volume, velocity and variety). Tilgjengelig fra: https://www.techtarget.com/whatis/definition/3Vs (Hentet: 14.04.2024).

Malhotra, A., Agarwal, R., Khandhar, H. & Mathur, R. (2023) *SqueezeNet: The Key to Unlocking the Potential of Edge Computing.* Tilgjengelig fra: https://medium.com/sfu-cspmp/squeezenet-the-key-to-unlocking-the-potential-of-edge-computing-c8b224d839ba . (Hentet: 01.05.2024).

Mavropalias, K. (2019) *Understanding Fastai's fit_one_cycle method*. Tilgjengelig fra: https://iconof.com/1cycle-learning-rate-policy/ (02.05.2024).

MargCampusSoft (2023) *Understanding the PIL Format*: A Powerful Tool for Image Processing. Tilgjengelig fra: https://margcompusoft.com/m/pil-format/ (Hentet: 11.05.2024).

Matplotlib (u.å) Matplotlib: Visualization with Python. Tilgjengelig fra: https://matplotlib.org/ (Hentet: 11.05.2024).

Neville (2023) *Dice Loss In Medical Image Segmentation*. Tilgjengelig fra: https://cvinvolution.medium.com/dice-loss-in-medical-image-segmentation-d0e476eb486 (Hentet: 20.04.2024).

National Energy Technology Laboratory (u.å) *Carbon Storage FAQs*. Tilgjengelig fra: https://netl.doe.gov/carbon-management/carbon-storage/faqs/carbon-storage-faqs (Hentet: 17.04.24)

Nazaré, T., da Costa, G., Contato, W., Ponti, M. (2018). *Deep Convolutional Neural Networks and Noisy Images.,*. Tilgjengelig fra: https://link.springer.com/chapter/10.1007/978-3-319-75193-1_50, (Hentet: 15.04.2024)

NHO (u.å) *CO2-fangst og lagring (CCS)*. Tilgjengelig fra: https://www.nho.no/tema/energi-miljo-og-klima/artikler/co2-fangst-og-lagring-ccs/) (Hentet: 14.04.24).

Notion (u.å) Notion. Tilgjengelig fra: https://www.notion.so/ (Hentet: 16.01.2024).

Nvidia (u.å) *PyTorch*. Tilgjengelig fra: https://www.nvidia.com/en-us/glossary/pytorch/ (Hentet: 20.04.2024)

Opentext (u.å) *What is Agile Development?*. Tilgjengelig fra: https://www.opentext.com/what-is/agile-development (Hentet: 07.05.2024).

Pragya Shukla (2024) *CNN vs DNN for visual understanding of images*. Tilgjengelig fra: https://medium.com/@pragyashukla2580/cnn-vs-dnn-for-visual-understanding-of-images-3b3f35f493de (Hentet: 08.05.2024).

Rastogi, V. (2023) *Performance metrics*. Tilgjengelig fra: https://medium.com/@vaibhav1403/performance-metrics-ffc6a54a141c (Hentet: 13.05.2024).

Rørvik, A. (2013) *Den Vitenskapelige Metode*. Tilgjengelig fra: https://www.friskogfunksjonell.no/vitenskapelig-metode/ (Hentet: 02.05.2024).

Sabat´es de la Huerta, J. (2022) *Classifying astronomical sources with machine learning*. Tilgjengelig fra: https://diposit.ub.edu/dspace/bitstream/2445/189820/1/SABAT%C3%89S%20DE%20LA%20HU

ERTA%20JORDI_5181450_assignsubmission_file_TFG_Sabat%C3%A9s_Jordi_final.pdf (Hentet: 20.04.24).

Salomão, A. (2024) *Unveiling the Influence of Machine Learning in Science*. Tilgjengelig fra: https://mindthegraph.com/blog/machine-learning-in-science/ (Hentet: 02.05.2024).

Sarita (2023) *Basic understanding of Neural Network Structure.* Tilgjengelig fra: https://medium.com/@sarita_68521/basic-understanding-of-neural-network-structure-eecc8f149a23. (Hentet: 15.04.24)

Schumacher, D. (u.å) *Dice Loss*. Tilgjengelig fra: https://serp.ai/dice-loss/ (Hentet: 20.04.2024).

Simonyan, K. & Zisserman, A. (2015) *Very Deep Convolutional Networks For Large-Scale Image Recognition*. Tilgjengelig fra: https://arxiv.org/pdf/1409.1556/ (Hentet: 04.05.2024).

Sokkeldirektoratet (u.å). *CO2-lagring.* Tilgjengelig fra: https://www.sodir.no/fakta/co2-lagring/. (Hentet 24.02.24)

Staff, C (2023) *What Is Kaggle and What Is It Used For?*. Tilgjengelig fra: https://www.coursera.org/articles/kaggle/ (Hentet: 20.02.2024).

Szalvay, V. (2004) *An Introduction to Agile Software Development*. Tilgjengelig fra: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2efe4d840631ebf026fede741e85195e36f8b134 (Hentet: 12.01.2024)

Torchvision (u.å) *torchvision*. Tilgjengelig fra: https://pytorch.org/vision/stable/index.html (Hentet: 16.03.2024).

UiB (u.å) *FluidFlower.* Tilgjengelig fra: https://fluidflower.w.uib.no/ (Hentet: 12.02.2024)

USGS (u.å). *What is carbon sequestration?* Tilgjengelig fra: https://www.usgs.gov/faqs/what-carbon-sequestration (Hentet: 20.02.24)

Vincent, J. (2024) *How much electricity does AI consume?*. Tilgjengelig fra: https://www.theverge.com/24066646/ai-electricity-energy-watts-generative-consumption (Hentet: 05.05.2024).

WHO (2023). *Climate change.* Tilgjengelig fra: https://www.who.int/news-room/fact-sheets/detail/climate-change-and-health (Hentet 20.02.24)

W3Schools (u.å. a) NumPy Introduction. Tilgjengelig fra: https://www.w3schools.com/python/numpy/numpy_intro.asp (Hentet: 11.05.2024).

W3Schools (u.å. b) Pandas Introduction. Tilgjengelig fra: https://www.w3schools.com/python/pandas/pandas_intro.asp (Hentet: 11.05.2024).

Yan, Y., Borhani, T., Subraveti, S., Pai, K., Prasad, V., Rajendran, A., Nkulikiyinka, P., Asibor, J., et. al (2021) *Harnessing the power of machine learning for carbon capture, utilisation, and storage (CCUS) – a state-of-the-art review*. Tilgjengelig fra: https://pubs.rsc.org/en/content/articlelanding/2021/ee/d1ee02395k (Hentet: 20.04.24).

You, J. & Lee, K., (2021). *Pore-Scale Study to Analyze the Impacts of Porous Media Heterogeneity on Mineral Dissolution and Acid Transport Using Darcy–Brinkmann–Stokes Method. Transport in Porous Media*., Tilgjengelig fra: https://link.springer.com/article/10.1007/s11242-021-01577-3. (Hentet: 10.03.2024)

Zappone, A., Rinaldi, A. P., Grab, M., Wenning, Q. C., Roques, C., Madonna, C., Obermann, A. C., Bernasconi, S. M., Brennwald, M. S., Kipfer, R., Soom, F., Cook, P., Guglielmi, Y., Nussbaum, C., Giardini, D., Mazzotti, M., and Wiemer, S. (2021) *Fault sealing and caprock integrity for CO2 storage: an in situ injection experiment*. Tilgjengelig fra: https://se.copernicus.org/articles/12/319/2021/, (Hentet: 27.02.2024)

10. Vedlegg

Vedlegg 1 Prosjekthåndbok