Høgskulen på Vestlandet

BACHELOROPPGAVE

Utnytting av symmetri til å kunstig øke mengden treningsdata for klassifisering av mikroskopiske sorte hull og sfaleroner

– Ved bruk av simulerte kollisjonsdata fra ATLAS (HVL)

Utilizing Symmetry to Artificially Augment Training Data for Classification of Microscopic Black Holes and Sphalerons

– Using Simulated Collision data from ATLAS (HVL)

Sunniva Storetvedt Lothe

Vladimirs Sergejevics Civilgins

Bachelor, Dataingeniør

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

Carsten Gunnar Helgesen

22. mai 2023

Fakultet for ingeniør- og naturvitskap
Institutt for datateknologi, elektroteknologi og realfag

TITTELSIDE FOR HOVEDPROSJEKT

Rapportens tittel: Utnytting av symmetri til å kunstig øke mengden treningsdata for klassifisering av mikroskopiske sorte hull og sfaleroner ved bruk av simulerte kollisjonsdata fra ATLAS (HVL)	*Dato:* 22.05.2023
Forfatter(e): Sunniva Storetvedt Lothe og Vladimirs Sergejevics Civilgins	*Antall sider u/vedlegg:* 107 *Antall sider vedlegg:* 109
Studieretning: Dataingeniør	*Antall disketter/CD-er:* 0
Kontaktperson ved studieretning: Carsten Gunnar Helgesen	*Gradering:* Ingen
Merknader: Ingen	

Oppdragsgiver: HVL ATLAS Group	*Oppdragsgivers referanse:* Ingen
Oppdragsgivers kontaktperson: Trygve Buanes Steffen Mæland	*Telefon:* 55 58 70 87 55 58 77 94

Sammendrag:
Dette bachelorprosjektet har som mål å forbedre analysestrategier for kollisjonsdata fra ATLAS-detektoren ved Large Hadron Collider (LHC) på CERN ved å utnytte teknikker innen maskinlæring. Prosjektet er et samarbeid med Høgskulen på Vestlandet sin ATLAS-gruppe, som undersøker bruken av maskinlæring til å besvare uløste spørsmål innen partikkelfysikk og om universet. Prosjektet har fokusert på studiet av simulerte proton-proton kollisjoner i ATLAS-eksperimentet og den påfølgende analysen av dataene. Spesifikt har prosjektet utforsket bruken av maskinlæring for å forbedre klassifiseringen av mikroskopiske sorte hull og sfaleroner ved å trene en dyplæringsmodell til å gjenkjenne mønstre og symmetrier i detektordataene.

Stikkord:

Fundamental partikkelfysikk	ATLAS, CERN	Maskinlæring og nevrale nettverk

Høgskulen på Vestlandet, Fakultet for ingeniør- og naturvitskap
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

I denne rapporten dokumenteres arbeidet bak bachelorprosjektet *Utnytting av Symmetri til å kunstig øke mengden treningsdata for klassifisering av Mikroskopiske sorte hull og Sfaleroner ved bruk av simulerte kollisjonsdata fra ATLAS (HVL).* Dette prosjektet er en avsluttende del av vår treårige utdannelse på dataingeniør med spesialisering innen maskinlæring og kunstig intelligens ved Høgskulen på Vestlandet (HVL).

Vi ønsker å takke alle som har bidratt til den vellykkede gjennomføringen av dette prosjektet. Først og fremst ønsker vi å rette en stor takk til vår veileder, Carsten Gunnar Helgesen, for sine konstruktive og innsiktsfulle tilbakemeldinger gjennom prosjektløpet. Like fullt ønsker vi å takke våre kontaktpersoner fra HVL sin ATLAS gruppe, Trygve Buanes og Steffen Mæland, for umåtelig veiledning og støtte gjennom forskningsprosessen. Uten deres ekspertise og tålmodighet ville fullføringen av dette prosjektet vært mye mer krevende. Vi ønsker også å takke doktorgradsstipendiat fra HVL sin ATLAS gruppe, Aurora Singstad Grefsrud, for sitt bidrag med datasett og startkode.

Avslutningsvis ønsker vi å takke hverandre for et godt og gjensidig samarbeid. Ved å lære om hverandres styrker og svakheter, har vi klart å utfylle hverandre på en god måte. Gruppe-dynamikken og vår felles interesse for fagfeltet har vært en nøkkel til suksess, og vi er takknemlige for den verdifulle lærdommen vi har fått.

Abstract

This bachelor's project aims to enhance analysis strategies for collision data from the ATLAS detector at the Large Hadron Collider (LHC) in CERN by utilizing machine learning techniques. The project is a collaboration with the ATLAS research group at the Western Norway University of Applied Sciences, which investigates the use machine learning to answer unresolved questions in particle physics and the universe. The project has focused on the study of simulated proton-proton collisions in the ATLAS experiment and the subsequent analysis of the data. Specifically, the project has explored the use of machine learning to improve the classification of microscopic Black holes and Sphalerons by training a deep learning model to recognize patterns and symmetries in the detector data

Ordliste partikkelfysikk

Ord	Forklaring / definisjon
Bosoner	Er en klasse av subatomære partikler som kan eksistere i samme kvantetilstand samtidig, som for eksempel fotoner og Higgs-boson.
Fermioner	Fermioner er små partikler som inkluderer elektroner, og de adlyder regler som sier at de ikke kan være på samme sted samtidig og har forskjellige egenskaper.
Higgs-boson	Higgs-bosonet er en subatomær partikkel som spiller en nøkkelrolle i å forklare hvordan andre partikler får sin masse, og ble bekreftet ved Large Hadron Collider-eksperimentet i 2012.
Kvarker	Kvarker er en type subatomære partikler som utgjør byggesteinene til hadroner, som protoner og nøytroner.
Leptoner	Leptoner er en type elementærpartikler som ikke er påvirket av den sterke kjernekraften og inkluderer elektroner, myoner og tau-partikler samt deres tilsvarende nøytrinoer.
Materie	Materie er ethvert stoff som har masse og tar opp plass.
Mikroskopiske sorte hull	Mikroskopiske sorte hull er hypotetiske sorte hull som er antatt å ha svært liten størrelse, på størrelsesorden av nanometer eller mindre, og som ville fordampe raskt.
Mørk energi	Mørk energi er en ukjent form for energi som antas å være til stede i hele universet, og som påvirker utvidelsen av universet.
Mørk materie	Mørk materie er en form for materie som ikke interagerer med lys eller annen elektromagnetisk stråling, og ingen kjente partikler kan utgjøre den mørke materien.
Sfaleroner *(Sphalerons)*	Sfaleroner er en hendelse som potensielt kan skje i kollisjoner ved LHC. De kan beskrives som dannelsen av en type topologisk defekt som kan oppstå når kvarker og gluoner kolliderer med høy energi, og kan bidra til å studere asymmetrien mellom materie og antimaterie i universet.
Standardmodellen	Standardmodellen er en teoretisk ramme for å beskrive de grunnleggende byggesteinene i naturen, inkludert elementærpartikler og deres krefter, som elektromagnetisk kraft, sterk kjernekraft og svak kjernekraft.
Subatomære partikler	Subatomære partikler er partikler som er mindre enn atomer, og inkluderer elementærpartikler som kvarker, leptoner og bosoner, som er byggesteinene i atomer og alle andre typer materie.

Ordliste maskinlæring

Merknad: De fleste begrepene er gitt på både norsk og engelsk. Dette skyldes at de engelske begrepene er betydelig mer utbredt og brukt enn de norske oversettelsene. Det finnes imidlertid noen få begreper som ikke har blitt oversatt på grunn av mangelen på tilfredsstillende norske oversettelser.

Ord	Forklaring/definisjon
AdaDelta	AdaDelta er en forbedring av optimeringsmetoden, Adagrad.
Adagrad (*Adaptive Gradient*)	Adagrad er en optimeringsmetode som justerer læringsraten for hver parameter basert på historiske gradientverdier. Dette gjør det mulig å ha forskjellige læringsrater for sjeldne og hyppige funksjoner.
Adam (*Adaptive Moment Estimation*)	Adam er en adaptiv optimeringsmetode som kombinerer elementer fra RMSprop og Adagrad, og beregner individuelle adaptive læringsrater for forskjellige parametere basert på deres gradienthistorie.
Adamax	Adamax er en forbedring av optimeringsmetoden, Adam.
Assosiasjonsregler (A*ssociation rules*)	Assosiasjonsregler er en teknikk i maskinlæring som avdekker relasjoner mellom variabler i store datasett. Et vanlig eksempel er markedskurv-analyse, der man søker å finne regler som indikerer at hvis en kunde kjøper et bestemt produkt, er det sannsynlig at de også kjøper et annet produkt.
Batch	*Batch* er en samling datapunkter som behandles sammen i en enkelt iterasjon av treningsprosessen. Batching brukes for å effektivisere treningen og redusere støy i oppdateringene.
Batch-normalisering *(Batch normalization)*	Batch-normalisering er en teknikk i nevrale nettverk som standardiserer inndata, forbedrer hastighet og stabilitet ved å skalere og justere utdata i hvert lag.
Bildegjenkjenning *(Image recognition)*	Bildegjenkjenning går ut på å identifisere og klassifisere objekter og mønstre i visuelle data, som eksempelvis bilder.
Delphes	Delphes er et rammeverk som simulerer responsen til partikkeldetektorer i partikkelfysikk, og brukes til å analysere og tolke eksperiment data ved å generere en simulert utgang basert på partikkel informasjon fra kollisjon
Egenskapskart (*Feature map*)	Egenskapskart er en samling av aktiveringsverdier fra et bestemt konvolusjonslag som representerer egenskaper eller mønstre som er lært av nettverket. Egenskapskartene hjelper til med å visualisere hva modellen har lært.
Epok *(Epoch)*	En epoke innen maskinlæring refererer til en enkelt gjennomkjøring av hele treningssettet gjennom et nevralt nettverk.

Ord	Forklaring/definisjon
Etiketter *(Labels)*	Etiketter er tildelte svar eller kategorier for datapunkter som brukes i veiledet læring. Etiketter fungerer som grunnlag for sammenligning og evaluering av modellens forutsigelser.
Forskyvningsverdi *(Bias)*	Forskyvningsverdi er en ekstra parameter i et nevralt nettverk som hjelper til med å justere utgangen ved å legge til en konstant forskyvning. Bias hjelper modellen med å tilpasse seg mer fleksibelt til data og er avgjørende for å lære komplekse mønstre.
Forsterket læring *(Reinforcement learning)*	Forsterket læring er en maskinlæringsmetode der agenter lærer fra en sekvens av handlinger og tilhørende belønninger i et miljø. Målet er å maksimere den samlede belønningen over tid.
Forsvinnende gradient *(Vanishing gradient)*	Forsvinnende gradient er en utfordring i dype nevrale nettverk, preget av gradienter som krymper eksponentielt under bakoverpropagering, noe som forsinker læring.
Forvirringsmatrise *(Confusion matrix)*	Forvirringsmatrise er en tabell som viser antallet korrekte og feilaktige prediksjoner basert på faktiske og forutsagte klasser. Den gir innsikt i modellens ytelse og feilklassifiseringer.
Fremoverpropagering (*Forward propagation*)	Fremoverpropagering er prosessen der informasjon i et nevralt nettverk flyter fra inngangslaget, gjennom de skjulte lagene og til utgangslaget. Dette skjer ved å anvende aktiveringsfunksjoner og vektede forbindelser mellom lagene for å produsere den endelige utgangen.
Fullt tilkoblet lag *(Fully connected layer)*	Et fullt tilkoblet er et lag i et nevralt nettverk der hvert nevron er tilkoblet alle nevronene i forrige og neste lag. Fullt tilkoblet lag brukes ofte for å produsere endelige prediksjoner eller klassifiseringer i et nettverk.
Generaliseringskompleksitet	Generaliseringskompleksitet går ut på en modell sin evne til å tilpasse til nye og ukjente data, basert på det den har lært fra trening av kjent data.
GitHub	GitHub er en webbasert plattform som brukes til samarbeid og deling av kodeprosjekter blant utviklere. Det gir funksjoner som versjonskontroll, styring av oppgaver og kodeanmeldelser.
Gjennomsnittlig pooling *(Average pooling)*	Gjennomsnittlig pooling er en type pooling som tar gjennomsnittet av verdiene i et gitt område av inndataen. Gjennomsnittlig pooling bidrar til å redusere støy og glatte egenskapskartene.
Gradientnedstigning *(Gradient descent)*	Gradientnedstigning er en optimeringsalgoritme som brukes i maskinlæring for å minimere tapsfunksjonen ved å justere modellens parametere. Den gjør dette ved å beregne

Ord	Forklaring/definisjon
	gradienten av tapet med hensyn til hver parameter, og deretter ta små skritt i motsatt retning av gradienten for å finne den laveste tap-verdien.
Grafikkort (GPU) *(Graphic Processing unit)*	Grafikkort er en grafisk prosessor for raske parallelle beregninger.
Gruppering *(Clustering)*	Gruppering er en teknikk innenfor maskinlæring som tar sikte på å gruppere datapunkter basert på likheter i deres attributter. Dette er en form for ikke-overvåket læring, ettersom modellen lærer mønstre uten å bli gitt eksplisitte kategorier.
Ikke-veiledet læring *(Unsupervised learning)*	Ikke-veiledet læring er en maskinlæringsmetode som bruker data uten etiketter for å lære underliggende mønstre og strukturer. Eksempler er klyngeanalyse og dimensjonsreduksjon.
Inngangslag *(Input layer)*	Inngangslag i et nevralt nettverk er det første laget som mottar inndataene og sender dem videre til de påfølgende lagene. Dette laget er ansvarlig for å konvertere rådata til en form som kan behandles av nettverket.
Klassifisering	Klassifisering er en maskinlæringsoppgave som består i å tilordne en bestemt klasse eller kategori til et gitt inndata eksempel.
Konvolusjonelle nevrale nettverk *(CNN)*	Konvolusjonelle nevrale nettverk er en type dyp læring algoritme som er spesielt egnet for å analysere data som bilder eller lyd. De består av flere konvolusjons- og reduksjonslag for å ekstrahere høyere nivå av funksjoner og mønstre.
Konvolusjonsfilter	Konvolusjonsfilter, også kjent som kjerne eller egenskaps-detektor, er en liten matrise som brukes i konvolusjonelle nevrale nettverk for å analysere og lære lokale mønstre i bilder.
Kryssentropi *(Cross entropi)*	Kryssentropi er en populær tap-funksjon som brukes i klassifiseringsoppgaver. Den måler forskjellen mellom den virkelige sannsynlighetsfordelingen og modellens forutsagte sannsynlighetsfordeling, og er spesielt nyttig når man jobber med flerklassifiseringsproblemer.
L1- og L2-regularisering	Regulariseringsteknikker som legger til straff på modellvektene basert på deres absolutte (L1) eller kvadratiske (L2) verdier. Dette hjelper til med å kontrollere modellkompleksitet og forhindre overtilpasning.
Læringshastighet *(Learning rate)*	Lærings hastighet er en hyperparameter i maskinlæring som bestemmer hvor raskt en modell oppdaterer sine parametere, som vekter og bias, under treningsprosessen. En høy

Ord	Forklaring/definisjon
	læringshastighet kan føre til at modellen raskt konvergerer, men risikerer å hoppe over det optimale punktet, mens en lav læringshastighet sikrer en mer nøyaktig konvergens, men krever mer tid og ressurser.
Maksimum pooling *(Max pooling)*	Maks. pooling er en teknikk innen bildebehandling og datavitenskap som brukes for å redusere dimensjonaliteten til bilder eller andre data uten å miste viktig informasjon. Det gjøres ved å dele opp bildet i mindre rektangulære områder og velge det største elementet i hvert område.
Metrikker *(Metrics)*	Metrikker er mål som brukes til å evaluere en modells ytelse på forskjellige aspekter, som nøyaktighet, presisjon og gjenvinningsrate. Metrikker hjelper til med å kvantifisere og sammenligne modellresultater.
Minimum pooling *(Minimum pooling)*	Min. pooling en form for pooling som tar den minste verdien i et gitt område av dataene. Min. pooling kan bidra til å understreke de minst fremtredende funksjonene i et egenskapskart.
Modellkompleksitet	Modellkompleksitet refererer til mengden informasjon en modell kan representere eller hvor fleksibel den er for å tilpasse seg data. Høyere kompleksitet kan føre til overtilpasning, der modellen blir for spesifikk for treningsdataene og presterer dårlig på nye data.
Nøyaktighet *(Accuracy)*	Nøyaktighets er andelen av korrekte prediksjoner blant totalt antall prediksjoner. Nøyaktighet er en enkel og vanlig metode for å vurdere modellens ytelse, men kan være villedende i ubalanserte datasett.
Overtilpasning *(Overfitting)*	Overtilpasning oppstår når en modell i maskinlæring er for kompleks og lærer for mye fra trening dataene, slik at den ikke generaliserer godt til nye data. Dette kan føre til at modellen presterer dårligere på nye data enn på trening dataene.
Planlegger *(Scheduler)*	En planlegger justerer dynamisk parametere som læringshastighet under trening av en maskinlæringsmodell for å optimalisere ytelse og forbedre generaliseringsevnen.
Presisjon *(Precision)*	Presisjon andelen korrekte positive prediksjoner blant de positive prediksjonene. Presisjon måler modellens evne til å korrekt identifisere positive tilfeller uten falske positive.
ReduceLROnPlateau – planlegger	ReduceLROnPlateau- planlegger overvåker tapet på valideringssettet under treningen og reduserer læringshastigheten med en faktor når tapet ikke forbedres i løpet av et visst antall epoker for å hjelpe modellen med å finne en bedre optimeringsbane.

Ord	Forklaring/definisjon
Reduksjonslaget *(Pooling layer)*	Reduksjonslaget er et lag i et nevralt nettverk som reduserer dimensjonene til innkommende data, noe som hjelper til med å redusere beregningskostnaden og fokusere på de mest informative funksjonene.
Regresjon	Regresjon er en maskinlæringsoppgave som består i å predikere en kontinuerlig verdi fra inngangsvariablene. For eksempel kan man bruke regresjon for å forutsi hus priser basert på ulike attributter som areal, antall soverom og nabolag.
ReLU (*Rectified Linear Unit*)	ReLU er en aktiviseringsfunksjon som ofte brukes i dyp læring. Det gir en enkel beregning og effektiv trening av nevrale nettverk ved å eliminere negative verdier og beholde positive verdier.
RMSprop (*Root Mean Square Propagation*)	En optimeringsmetode som bruker en løpende gjennomsnittlig kvadratisk gradient for å tilpasse læringsraten for hver parameter. Dette hjelper til med å forhindre at læringsraten blir for stor eller liten.
Sensitivitet *(Sensitivity/ Recall)*	Sensitivitet måler andelen korrekte positive prediksjoner blant de faktiske positive. Sensitivitet er også kjent som gjenvinningsrate og brukes til å evaluere modellens evne til å identifisere positive tilfeller.
Sigmoid	Sigmoid er en aktiveringsfunksjon i nevrale nettverk som transformerer inngangsverdier til en skala fra 0 til 1. Ofte brukt i binær klassifisering.
Skjevhet (*Bias*)	I maskinlæring refererer skjevhet *(bias)* til en modells tendens til å lage systematiske feil ved å forenkle problemet den forsøker å lære. En høy-skjevhet-modell kan under-tilpasse seg dataene og ha dårlig generaliseringsytelse.
Skjulte lag (*Hidden layer*)	Mellomliggende lag i et nevralt nettverk som behandler informasjonen som kommer fra inngangslaget og videreformidler den til utgangslaget. Disse lagene kan være flere og er ansvarlige for å lære komplekse mønstre og representasjoner i dataene.
Steglengde (*Stride*)	Steglengden refererer til hvor mange enheter et filter eller en kjerne beveger seg over inndataene i en konvolusjonell nevralnett-operasjon. En større steglengde vil resultere i en mindre størrelse på utdata og mindre overlapping, mens en mindre stride-verdi gir en større størrelse på utdata og mer overlapping.
Tanh	Tanh er en aktiveringsfunksjon i nevrale nettverk som kartlegger inngangsverdier til et område mellom -1 og 1. Gir null-sentrert utdata, noe som kan forbedre læring.

Ord	Forklaring/definisjon
Tapsfunksjon (L*oss function*)	En tapsfunksjon er en funksjon som måler forskjellen mellom prediksjonene en modell gjør og de faktiske verdiene. Under trening søker modellen å minimere denne funksjonen for å forbedre nøyaktigheten av sine prediksjoner.
Tilbakepropagering (*Back propagation*)	Tilbakepropagering en algoritme som brukes i nevrale nettverk for å minimere feilen mellom modellens prediksjoner og de faktiske målverdiene. Den beregner gradienten av tapet med hensyn til hver vektparameter og justerer deretter vektene for å redusere feilen.
Undertilpasning (*Underfitting*)	Undertilpasning oppstår når en modell i maskinlæring ikke lærer tilstrekkelig fra treningsdataene, slik at den ikke generaliserer godt til nye data. Dette kan føre til at modellen presterer dårlig både på treningsdataene og på nye data.
Utfylling (*Padding*)	Padding brukes for å legge til kunstige data, ofte nuller, rundt inndataene for å justere størrelsen til ønsket format. Dette er nyttig i konvolusjonelle nevrale nettverk for å hindre at inndatadimensjonene blir redusert for mye under prosessering.
Utgangslag (*Output layer*)	Utgangslaget i et nevralt nettverk genererer den endelige utgangen eller prediksjonen for nettverket. Denne utgangen kan være kontinuerlig for regresjonsoppgaver, eller diskret for klassifiseringsoppgaver.
Utkobling (*Dropout*)	Utkobling er en teknikk for å forhindre overtilpasning hvor tilfeldige nevroner blir "slått av" under trening, noe som tvinger nettverket til å lære mer robuste representasjoner. Dropout øker generaliseringen av modellen.
Veiledet læring (*Supervised learning*)	Veiledet læring er en maskinlæringsmetode der modellen «lærer» fra etiketterte data under trening. Dette brukes ofte for klassifisering og regresjonsoppgaver.
Vekt (*Weights*)	Vektene i et nevralt nettverk er de tilkoblede styrkene mellom nevronene i forskjellige lag. Disse verdiene justeres under trening for å minimere feilen og lære de underliggende mønstrene i dataene.

Akronymer

Akronym	Betydning
ALICE	A Large Ion Collider Experiment
ATLAS	A Toroidal LHC ApparatuS
CERN	European Organization for Nuclear Research
CMS	Compact Muon Solenoid
CNN	Convolutional Neural Network
CRISP-DM	Cross-industry standard process for data mining
FN	False negative
FP	False positive
GPU	Graphic Processing Unit
HVL	Høgskulen på Vestlandet
IBM	International Business Machines
LHC	Large Hadron Collider
LHCb	Large Hadron Collider beauty
ReLU	Rectified Linear Unit
RGB	Color model Red Green Blue
SCT	Semiconductor tracker
TN	True negative
TP	True positive
EMA	Exponential Moving Averages
TRT	Transition Radiation Tracker
VSCode	Visual Studio Code
RAM	Random Access Memory

INNHOLDSFORTEGNELSE

FIGURLISTE

TABELLISTE

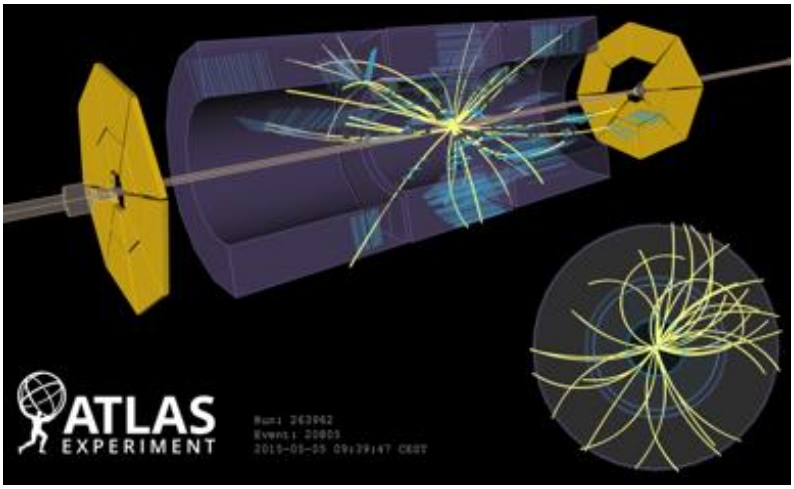
1 INNLEDNING

Dette kapittelet gir en sammenfatning av bakgrunnen og motivasjonen for prosjektet, presenterer oppdragsgiveren og beskriver det overordnede problemet som prosjektet tar sikte på å løse. Til slutt blir strukturen i rapporten presentert.

1.1 Kontekst

ATLAS-gruppen ved Høgskulen på Vestlandet (HVL) er en forskningsgruppe bestående av engasjerte fysikere og dataforskere. Gruppen ønsker å utforske hvorvidt oppdagelser innen partikkelfysikk kan være med på å finne svar på de ubesvarte spørsmålene om universet (HVL Atlas Group, 2023b). Ved å anvende ulike teknikker innen maskinlæring, en gren innen kunstig intelligens, arbeider de med simulerte data fra en detektor kalt *A Toroidal LHC ApparatuS* (ATLAS).

ATLAS er et av de store partikkelfysikkeksperimentene som er installert ved *Large Hadron Collider* (LHC) på *European Organization for Nuclear Research* (CERN). CERN er verdens største og mest anerkjente organisasjon for vitenskapelig forskning, inkludert partikkelfysikk- og kjernefysisk forskning (CERN, 2023a). I ATLAS-eksperimentet kolliderer subatomære partikler med hverandre, typisk protoner og ioner, med en hastighet nær lysets hastighet. Partikkelkollisjonene medfører en omfordeling av energi og materie, der eksisterende partikler kan bli omdannet til andre partikler eller ødelegges, og nye partikler kan skapes. Detektorer i ATLAS fanger opp produktene av kollisjonene og rekonstruerer samspillet mellom dem, som illustrert i Figur 1-1.



Figur 1-1 – Rekonstruksjon av en av de første partikkelkollisjonene som ble registrert av ATLAS den 5. mai 2015 (CERN Document Server [CDS], 2015). De gule strålene representerer partikler som skapes i partikkelsammenstøtene.

1.2 Motivasjon

ATLAS-detektoren ved CERN produserer enorme mengder data fra partikkelfysikk-eksperimentene. Med opptil 1,7 milliarder kollisjoner per sekund, genererer detektoren over 60 millioner megabyte data hvert sekund (ATLAS Collaboration, 2023e). For å håndtere denne enorme datamengden, kreves det omfattende databehandling og sofistikerte analysemetoder.

God databehandling og analyse spiller en avgjørende rolle i oppdagelsen av nye fysiske fenomener og teste vitenskapelige teorier. I henhold til den vitenskapelige metoden, er det essensielt å observere, danne hypoteser, teste og analysere data for å validere eller avvise hypotesene. En god analyse av data gir forskerne muligheten til å justere eksperimentelle metoder for å øke relevansen av dem, og dermed forbedre resultatene. Ved å avdekke de viktigste parameterne som påvirker resultatene, kan man justere og optimalisere dem for å få de beste resultatene, samtidig som man kan avsløre eventuelle feil eller usikkerheter i de eksperimentelle metodene. Dette fører til forbedrede eksperimentelle metoder og mer pålitelige resultater. En god analyse i fysiske eksperimenter spiller dermed en sentral rolle i å gjøre fremskritt innen fysikk.

Teknologisk utvikling ved CERN kan også overføres til andre forskningsområder og fagfelt. Siden 1970-tallet har CERN arbeidet med å finne medisinske anvendelser for teknologier som er utviklet innen eksperimentell partikkelfysikk (Cirilli, 2021). I dette arbeidet er det blant annet utviklet metoder og verktøy for behandling av kreft ved bruk av partikkelstråler for å skade kreftceller. Partikkelfysikk-inspirerte detektorer har også vært nyttige i utviklingen av metoder for medisinsk avbildning. På samme måte har utviklingen av medisinsk teknologi inspirert arbeidet innen eksperimentell partikkelfysikk. Dette vitner om en dualitet mellom disse feltene, ettersom arbeid fra ett område kan overføres til det andre. Forskningen ved CERN viste seg også å være viktig under Covid-19 pandemien, da laboratoriet utviklet ventilatorer og masker som var viktig for behandling av pasienter, samt sensorer og metoder for å håndtere data som var viktige for å begrense smittespredning. Ved å utvikle nye teknologier og metoder, bidrar forskning innen partikkelfysikk til å løse utfordringer innen en rekke vitenskapelige og samfunnsmessige områder.

1.3 Oppdragsgiver

Oppdragsgiveren for dette bachelorprosjektet er forskningsgruppen kalt *HVL ATLAS Group*. Gruppen består av fire førsteamanuenser, en postdoktor, to doktorgradsstudenter og flere masterstudenter, og en stor del av deres aktiviteter er finansiert av Norges forskningsråd (HVL ATLAS Group, 2023a, 2023b).

ATLAS-gruppen på HVL ledes av et team av fysikere og dataforskere som hovedsakelig arbeider med analyse av data fra ATLAS-eksperimentet ved CERN (HVL ATLAS Group, 2023b). Hovedfokuset deres ligger på å undersøke mørk materie og å utforske bruken av maskinlæringsteknikker for å analysere høyenergipartikkelkollisjoner fra ATLAS-detektoren. Deltakelsen i disse eksperimentene har ført til at forskerne har blitt en del av et omfattende internasjonalt nettverk, samt et nasjonalt nettverk der de samarbeider med andre høyenergifysikkgrupper ved andre universiteter.

1.4 Overordnet problembeskrivelse

Den konvensjonelle metoden for innsamling og analyse av kollisjonsdata fra ATLAS består av tre trinn. Det første trinnet, kjent som *trigger og dataregistrering*, spiller en avgjørende rolle for databehandlingen av partikkelkollisjonene. På grunn av den enorme mengden data som produseres i ATLAS-detektoren, klarer ikke dagens kraftigste maskiner å lese ut, lagre og prosessere den raskt nok i forhold til kollisjonsraten (ATLAS Collaboration, 2023e). Dessuten er ikke alle kollisjonene av interesse eller relevante for vitenskapelige analyser. Derfor blir mengden data redusert i to omganger, slik at antall kollisjoner går fra opptil 1,7 milliarder per sekund til omtrent 1000 kollisjoner per sekund. Utvelgelsen av kollisjoner baseres på raske analyser av kollisjonshendelsene, som videresendes til et datalagringssystem for nærmere analyse.

De to neste trinnene i studiet av partikkelsammenstøt kalles *rekonstruksjon* og *analyse*. Tradisjonelt sett behandles disse som separate steg, men oppdragsgiver ønsker at det skal bli utviklet en ny analysestrategi som kombinerer dem i et forsøk på å forbedre sensitivitet i analysen (Buanes og Mæland, 2023). Datasettet som er gitt av oppdragsgiver består av simulerte kollisjonsdata fra ATLAS-detektoren. Simulerte kollisjonsdata brukes i stedet for innsamlede data fra detektoren av flere grunner, men i dette prosjektet er den viktigste grunnen

at det muliggjør veiledet maskinlæring. Dette skyldes at maskinlæringsmodellen[1] har evnen til å gjenkjenne karakteristiske mønstre i fenomenene den skal klassifisere, da den gjennom trening utvikler en forståelse av hva treningsdataene i datasettet representerer. Betydningen av dette i veiledet maskinlæring blir nærmere beskrevet i punkt 2.4.1. Den nye analysestrategien går ut på å behandle kollisjonsdata som bilder, og bruke maskinlæring for å identifisere mønstre og strukturer for å klassifisere bildene. Slik vil bildene representere rekonstruert data i rekonstruksjons-trinnet, og maskinlæringen tar seg av den nærmere analysen i analyse-trinnet.

I dette prosjektet vil det bli brukt en maskinlæringsmodell av typen, konvolusjonelt nevralt nettverk, for å klassifisere bilder som representerer mikroskopiske sorte hull og sfaleroner. Nærmere forklaringer av mikroskopiske sorte hull og sfaleroner er gitt i punkt 2.1.2, og maskinlæring og konvolusjonelle nevrale nettverk blir beskrevet i punkt 2.4. Mer detaljert problemstilling med tilhørende mål og forskningsspørsmål er gitt i punkt 3.1, etter innføringen av teoretisk bakgrunn.

1.5 Oppbygging av rapporten

Frem til nå er det gitt en kort introduksjon av konteksten og motivasjonen for bachelor-prosjektet, samt en overordnet problembeskrivelse. I kapittel 2 gis det en innføring i den teoretiske bakgrunnen for prosjektet, hvilket innbefatter fagfeltet partikkelfysikk, aktuelle konstruksjoner ved CERN, og maskinlæring. I kapittel 3 presenteres den praktiske bakgrunnen for prosjektet, der problembeskrivelsen utdypes med målformuleringer og relevante forskningsspørsmål. Kapittelet tar også for seg kravspesifikasjon fra oppdragsgiver og initiell løsningsidé, samt prosjektets begrensninger og avgrensninger. Videre oppsummeres prosjektets ressurser og litteratur om problemstillingen. I kapittel 4 presenteres designet av prosjektet, inkludert forslag til løsninger, valgt løsning og valg av verktøy og metodikk. I kapittel 5 blir den detaljerte løsningen for prosjektet gjennomgått. De påfølgende resultatene fra løsningen blir presentert i kapittel 6. I kapittel 7 diskuteres resultatene og arbeidet som har blitt utført, hvilket leder frem til en konklusjon i kapittel 8. I dette kapittelet blir det også gitt en introduksjon til videre arbeid for prosjektet.

[1] I dette bachelorprosjektet vil begrepene «maskinlæringsmodell» (eventuelt bare «modell») og «nevrale nettverk» bli brukt om hverandre. Begge begrepene refererer til den matematiske representasjonen av algoritmen som brukes til å trene en maskin til å gjenkjenne mønstre og ta beslutninger basert på inndata.

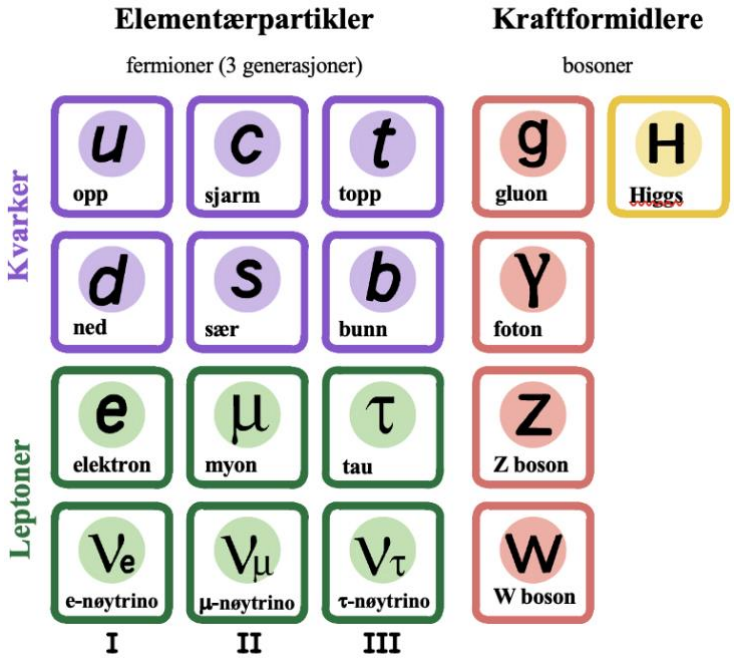
2 Teoretisk bakgrunn

Dette kapittelet gir en innføring i den teoretiske bakgrunnen som er nødvendig for å forstå prosjektet. Først introduseres relevante temaer innen partikkelfysikk, der standardmodellen forklarer hva universet består av og kreftene som styrer det. Videre beskrives to hypotetiske fenomener, mikroskopiske sorte hull og sfaleroner, som vil være kategoriene for klassifiseringen i dette prosjektet. Deretter beskrives partikkelakseleratoren, Large Hadron Collider, og ATLAS-eksperimentet i mer detalj. Avslutningsvis, blir det gitt en generell oversikt over maskinlæring og dens underkategorier. Dette innbefatter nevrale nettverk, dyp læring og konvolusjonelle nevrale nettverk, samt trening og evaluering av slike nettverk.

2.1 Partikkelfysikk

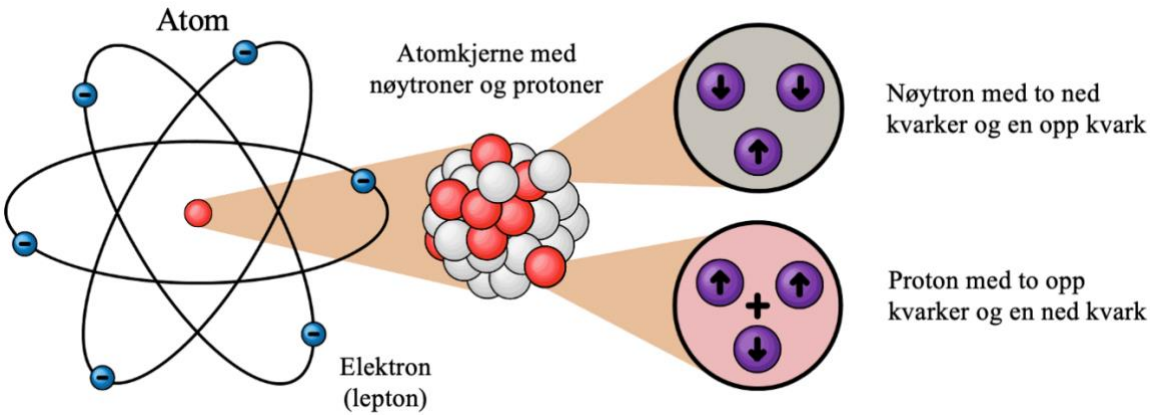
2.1.1 Standardmodellen

Standardmodellen er en av de mest vellykkede teoriene i fysikkhistorien og forklarer de grunnleggende kreftene og partiklene som utgjør universet (CERN, 2023d). Som illustrert i Figur 2-1, identifiserer standardmodellen tre grunnleggende partikkeltyper: kvarker (lilla) og leptoner (grønn) som tilhører undergruppen fermioner, og bosoner (rød og gul).



Figur 2-1 – Standardmodellen beskriver elementærpartikler og kraftpartikler (inspirert av figuren til Wikimedia, 2020).

Elementærpartiklene, kvarker og leptoner, er de minste og «udelelige» byggesteinene som er kjent per i dag. Figur 2-2 viser et eksempel på hvordan atomer kan deles inn i stadig mindre bestanddeler.



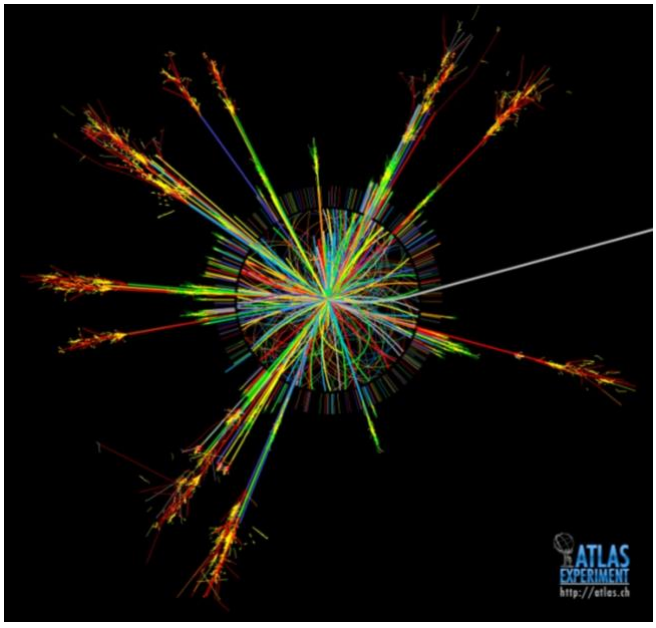
Figur 2-2 – Atomer kan deles inn i en atomkjerne med elektroner rundt. Elektronet er et av flere typer leptoner, mens atomkjernen kan deles inn i nøytroner og protoner. Videre kan nøytronene og protonene deles inn i ulike typer kvarker.

Standardmodellen beskriver også tre av de fire grunnleggende naturkreftene: sterk kjernekraft, svak kjernekraft og elektromagnetisk kraft (CERN, 2023d). Gravitasjonskraften er den fjerde og svakeste av de fundamentale kreftene, men den er ikke en del av standardmodellen. De andre kreftene har egne partikler som formidler dem, kalt bosoner. Den sterke kjernekraften virker på kvarker og holder atomkjernen sammen (Lopes og Perrey, 2022, s. 8-9). Den svake kjernekraften påvirker kvarker og leptoner, og sørger for at de tunge partiklene henfaller til lettere partikler. Den elektromagnetiske kraften virker på partikler med ladning og holder atomer sammen i molekyler.

Selv om standardmodellen har vist seg å være en nøyaktig teori gjennom omfattende testing, er den ikke i stand til å forklare alle de ubesvarte spørsmålene i fysikken. For å finne svar på disse spørsmålene, bruker forskere eksperimenter som ATLAS for å teste standardmodellen og søke etter bevis på ulike fenomener. Ved å kombinere teori og eksperimenter, arbeider forskere for å utvikle en mer fullstendig teori som kan forklare alle fenomener og egenskaper i universet, deriblant gravitasjonskraften.

2.1.2 Mikroskopiske sorte hull og sfaleroner

Partikkelfysikere har lenge vært interessert i å undersøke muligheten for dannelse av mikroskopiske sorte hull i høyenergikollisjoner i LHC (CERN, 2011). Dersom slike sorte hull skulle dannes i partikkelkollisjoner, ville de vært betydelig mindre enn sorte hull i verdensrommet. Det mikroskopiske sorte hullet ville også henfalt øyeblikkelig til standardmodell-partikler, som ville ha vært lett å oppdage i ATLAS-detektoren (CERN, 2023b). Figur 2-3 viser en simulering av et mikroskopisk sort hull, slik det ville sett ut i ATLAS-detektoren, der alle de fargerike strekene representerer ulike partikler. Til tross for at mikroskopiske sorte hull ville ha hatt en kort «levetid», ville deres eksistens kunne bekrefte teorier om flere dimensjoner i universet og være med på å forklare hvorfor tyngdekraften er mye svakere enn de andre naturkreftene som beskrives i standardmodellen. Oppdagelsen av mikroskopiske sorte hull kan dermed bidra til å utvide den menneskelige forståelsen av universet.

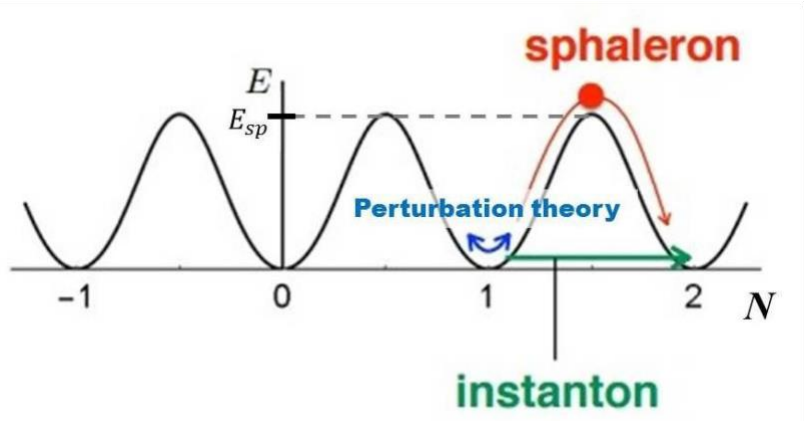


Figur 2-3 – Simulering av et mikroskopisk sort hull i ATLAS (CDS, 2018).

På samme måte som partikkelfysikkforskere har spekulert om eksistensen til mikroskopiske sorte hull, er sfaleroner en hypotetisk prosess som antas å kunne oppstå i partikkelkollisjoner i LHC. Det er en type prosess som kan oppstå når det samles nok energi i et lite område til at det oppstår en overgang fra et bunnpunkt til et annet, i en potensialfunksjon[2] slik Figur 2-4 viser

[2] En potensialfunksjon er en matematisk funksjon som brukes i fysikken for å beskrive potensiell energi eller potensielle krefter i et system.

(Ellis og Sakurai, 2016; Daria, u.å.). Sfaleroner kan potensielt føre til at partikler endrer noen av sine kvantetall[3], som elektrisk ladning og baryon- og leptontall[4]. Dette kan bryte med bevaringslover[5] i fysikken som sier at disse kvantetallene skal bevares i alle fysiske prosesser. På tross av at sfaleroner ikke er direkte observert, er deres eksistens teoretisk mulig. Hvis de blir oppdaget, kan de også være med på å bekrefte teorier om ekstra dimensjoner og bidra til en bedre forståelse av de grunnleggende kreftene som holder universet sammen.



Figur 2-4 – Et sfaleron fremstilt som en overgang fra et bunnpunkt til et annet i en potensialfunksjon. Energien, E, til sfaleronet vises langs y-aksen og ladningsnummeret, N, står langs x-aksen (Daria, u.å.).

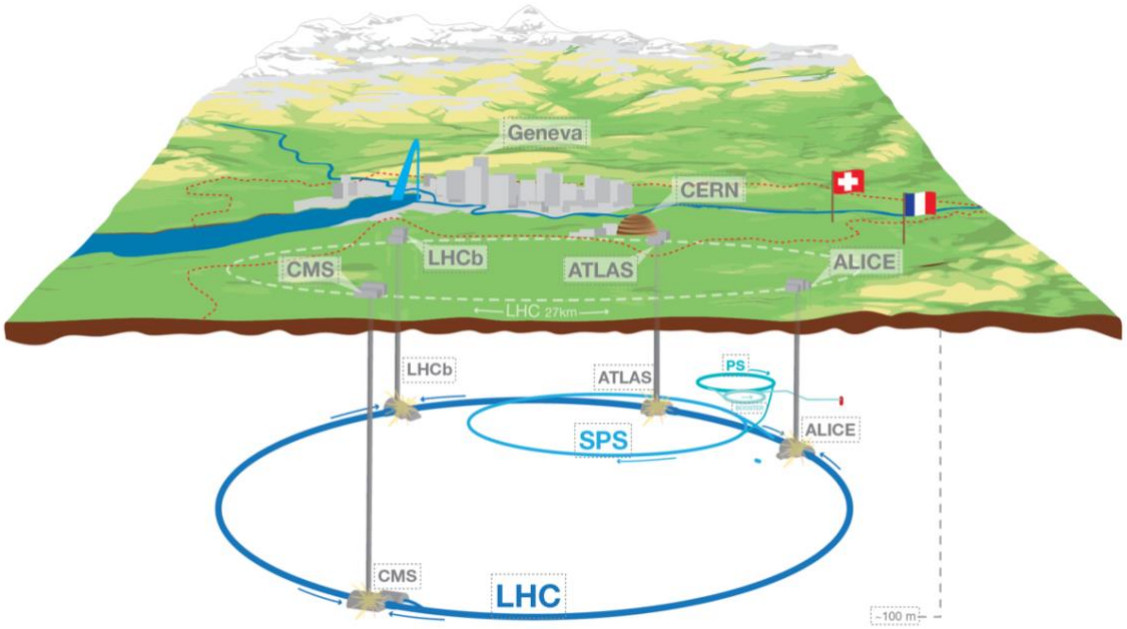
Mikroskopiske sorte hull og sfaleroner er eksempler på hypotetiske fenomener beskrevet av fysikk utover standardmodellen. ATLAS-detektoren ved CERN brukes til å søke etter slike fenomener ved å observere høyenergipartikkelkollisjoner og se etter spesifikke signaturer som etterlates av disse fenomenene. På tross av at både mikroskopiske sorte hull og sfaleroner vil etterlate nye partikler som er unike for det respektive fenomenet, kan signaturene være ganske like. Det kan derfor være utfordrende å skille mellom disse. I dette prosjektet vil maskinlæring brukes for å klassifisere og identifisere disse signaturene, slik et fingeravtrykk kan brukes for å identifisere et menneske.

[3] Kvantetall er en verdi som beskriver en egenskap ved et atom eller molekyl.
[4] Hadroner er partikler som består av kvarker (se Figur 2-1). Hadroner som består av tre kvarker kalles baryoner, som protoner og nøytroner er eksempler på. Leptoner og kvarker er elementærpartikler, der elektronet er et eksempel på et lepton. Lepton- og baryontall er to kvantetall som er bevart i alle prosesser observert til nå.
[5] I alle prosesser med elementærpartikler skal følgende bevaringslover gjelde: totalenergien, bevegelsesmengden, den elektriske ladningen, baryontallet og leptontallet, og angulært moment.

2.2 Large Hadron Collider og ATLAS

2.2.1 Large Hadron Collider

Large Hadron Collider (LHC) er en partikkelakselerator som befinner seg på grensen mellom Frankrike og Sveits, like utenfor byen Genève. Akseleratoren er bygget som en 27 kilometer lang ringtunnel, og er den største og mest kraftfulle partikkelakseleratoren i verden (CERN, 2023c). Selve ringtunnelen befinner seg under bakkenivå, slik Figur 2-5 viser, og ble designet og bygget av CERN.

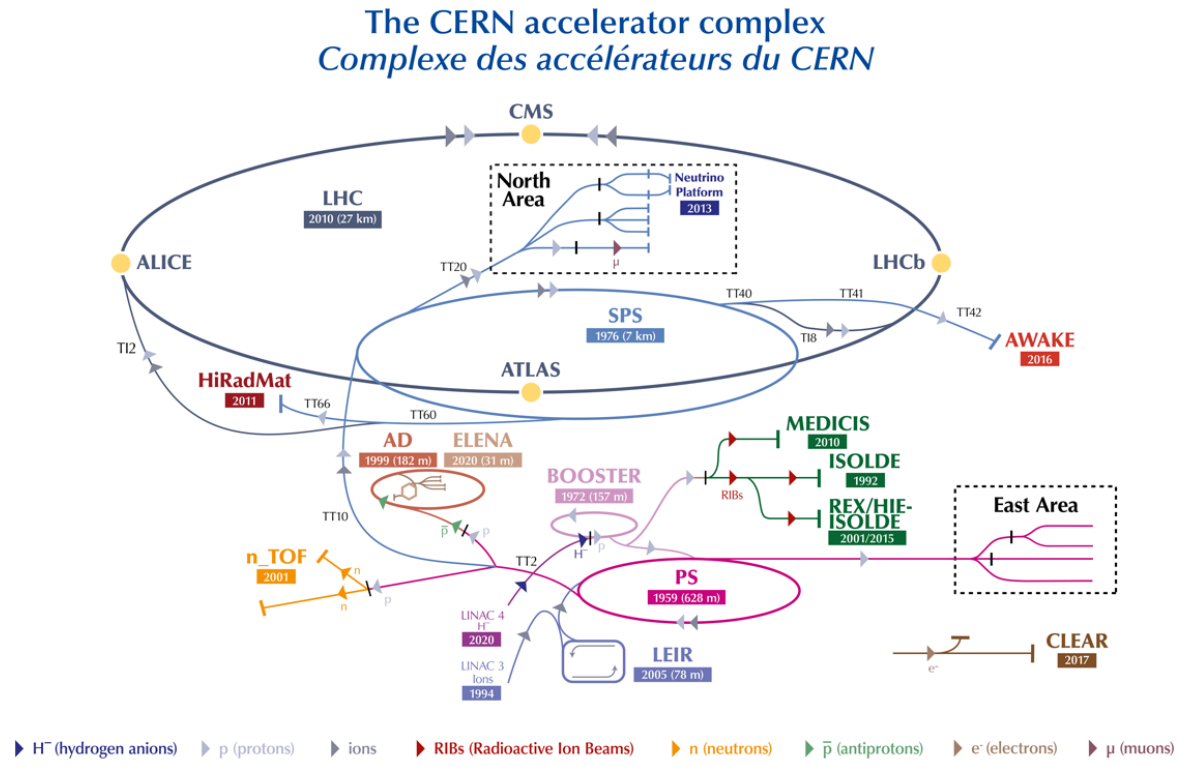


Figur 2-5 – LHC-anlegget under bakken i Genève (Zanini, 2022).

I LHC sendes to partikkelstråler med høy energi i motsatte retninger av hverandre (CERN, 2023c). Vanligvis består partikkelstrålene av protoner som akselereres nær lysets hastighet, og på andre tidspunkt kan partikkelstrålene bestå av tunge ioner. Videre føres strålene rundt tunellen ved hjelp av kraftige, superledende elektromagneter som skaper et sterkt magnetfelt. Strålene beveger seg gjennom separate strålerør med høyvakuum, for å unngå kollisjoner med molekyler i luften, før de ledes sammen for å kollidere ved fire punkter.

Ved hvert av de fire kollisjonspunktene i LHC er det plassert partikkelfysikkeksperimenter som utforsker partiklenes opprinnelse, sammensetning og interaksjoner (Lopes og Perrey, 2022, s. 38). De fire eksperimentene er: *A Toroidal LHC ApparatuS* (ATLAS), *Compact Muon Solenoid* (CMS), *A Large Ion Collider Experiment* (ALICE) og *Large Hadron Collider beauty* (LHCb).

ATLAS og CMS er generelle detektoreksperimenter som søker etter nye partikler og fenomener, mens ALICE undersøker kollisjoner av tunge ioner og LHCb fokuserer på studier av partikkelantimaterie. En skjematisk illustrasjon av det totale komplekset av akseleratorer er gitt i Figur 2-6



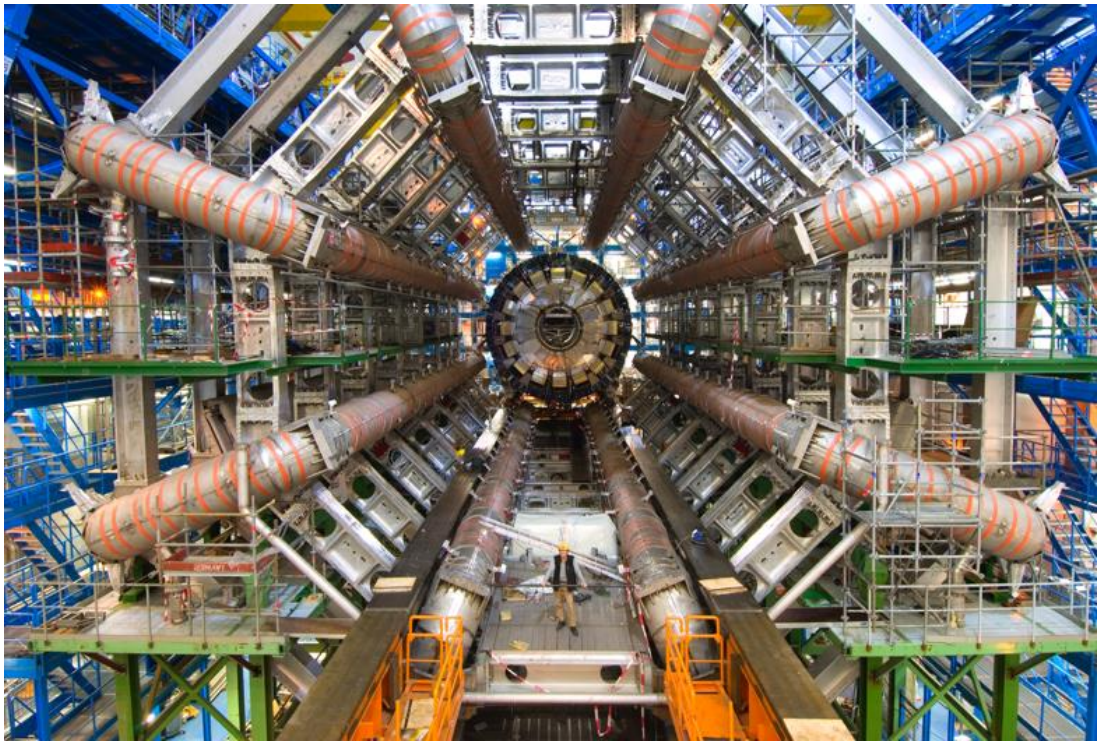
Figur 2-6 – Partikkelakseleratorkomplekset ved CERN der LHC utgjør den siste og største delen, markert med en mørkeblå oval i figuren. De mindre akseleratorene i komplekset øker partiklenes energi gradvis før de til slutt når sin endelige energi i LHC (CDS Server, 2019).

Hovedmålet til LHC er å få en mer komplett forståelse for universet. Dermed deler de samme mål som HVL sin ATLAS gruppe, nevnt i punkt Kontekst1.1, om å finne svar på ubesvarte spørsmål om universet. Følgelig er det blitt formulert spørsmål som konkret beskriver hva de ønsker å finne ut av (Lopes og Perrey, 2022, s. 22-25):

1. «Hva er opprinnelsen til masse?»
2. «Vil vi oppdage bevis for supersymmetri?»
3. «Hva er mørk materie og mørk energi?»
4. «Hvorfor er det langt mer materie enn antimaterie i universet?»
5. «Hvordan gir kvark-gluon plasma opphav til partiklene som utgjør materien i universet vårt?»

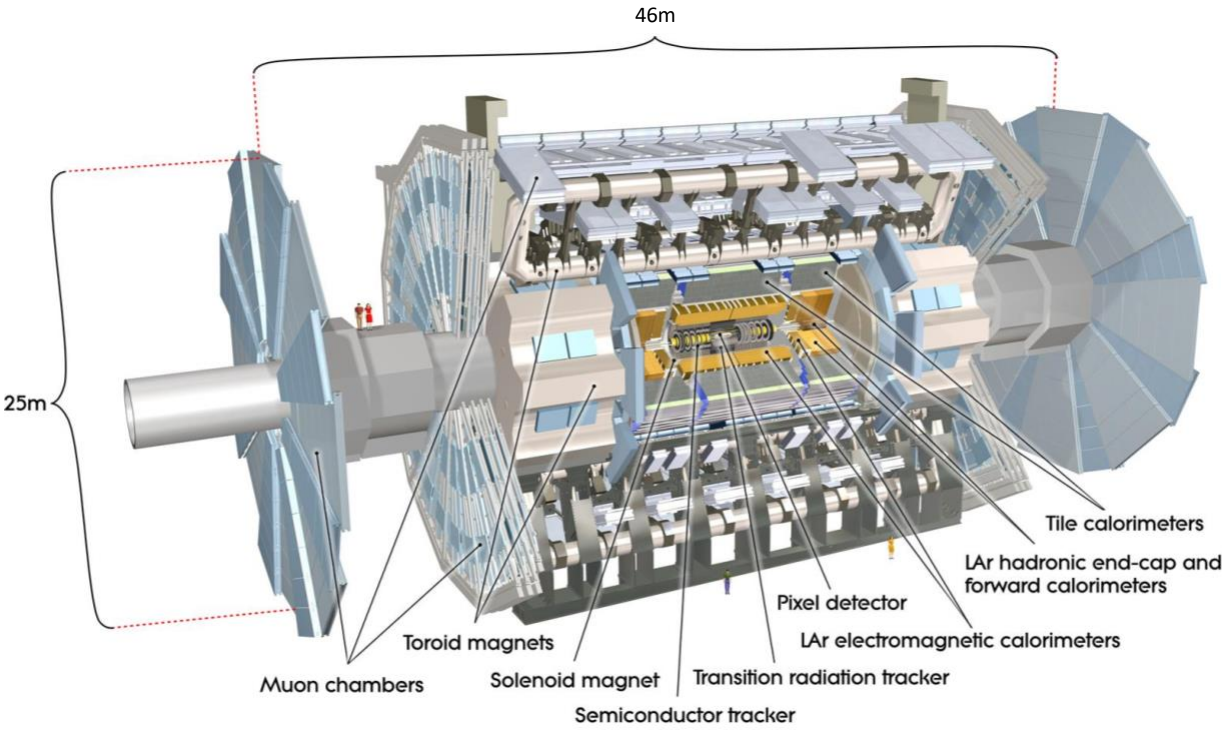
2.2.2 ATLAS

Som nevnt i punkt 2.2.1 er ATLAS er en av de fire hovedpartikkeldetektorene som befinner seg på LHC. Konstruksjonen av detektoren ble formelt godkjent i 1997 og den var ferdig bygget i 2008 (ATLAS Collaboration, 2023b). Figur 2-7 viser hvordan ATLAS så ut under konstruksjon. Den er designet for å måle og spore banene til partikler som produseres i høyenergikollisjoner i LHC, slik at forskere kan studere egenskapene til disse partiklene og de fundamentale kreftene som styrer deres oppførsel.



Figur 2-7 – Bilde av ATLAS-konstruksjonen under bygging. Bildet viser også størrelsesforskjellen mellom et menneske (nederst) og detektoren (CERN, 2005).

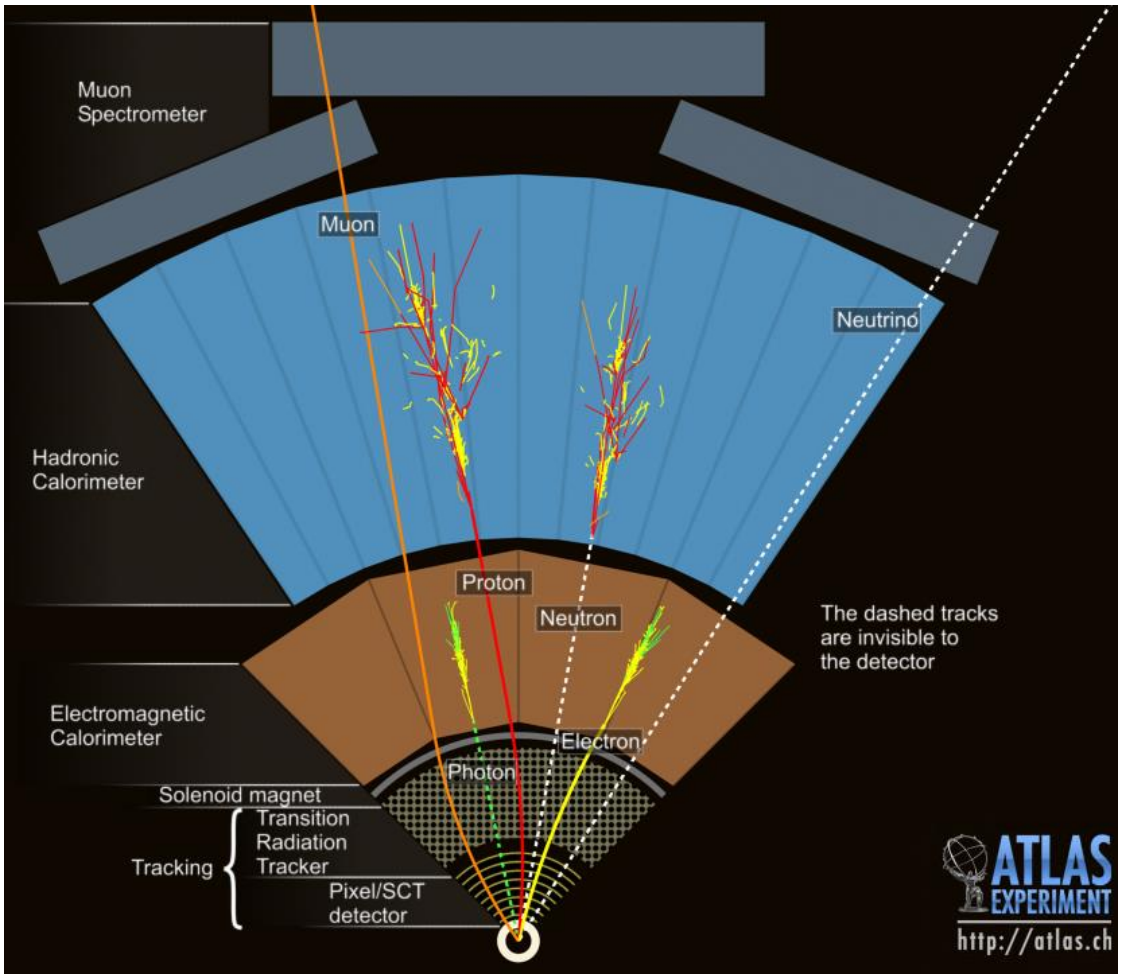
ATLAS-detektoren, som illustrert i Figur 2-8, har en sylindrisk form som måler 46 meter i lengde og 25 meter i diameter (Lopes og Perrey, 2022, s. 41). Den veier rundt 7000 tonn og befinner seg i en hule som ligger 100 meter under bakkenivå (CERN, 2023a). ATLAS består av en rekke underdetektorer, som hver er spesialisert på å måle bestemte aspekter ved det som kommer ut av kollisjonene (Lopes og Perrey, 2022, s. 48-50). Figur 2-8 viser også hvordan underdetektorene er strukturert, og disse kan kategoriseres som sporingssystemer og kalorimetere.



Figur 2-8 – Skjematisk illustrasjon av oppbyggingen av ATLAS (Zanini, 2022).

Figur 2-9 viser et tverrsnitt av ATLAS som illustrerer hvilke underdetektorer som identifiserer de ulike partiklene fra partikkelkollisjonene. Innerst er det en indre detektor som består av tre sporingssystemer: *Pixel-detektoren*, *Semiconductor tracker* (SCT) og *Transition Radiation Tracker* (TRT). Disse brukes sammen for å spore banene til ladede partikler som beveger seg gjennom detektoren (ATLAS Collaboration, 2023a). Ved hjelp av magnetfeltet i detektoren kan avansert programvare måle partiklenes kurve, og dermed bestemme partiklenes bevegelsesmengde[6]. Rundt den indre detektoren finnes det to hovedtyper av kalorimeter. Først er det den elektromagnetiske kalorimeteren som måler energien fra elektroner og fotoner som dannes når partikler kolliderer. Deretter er det hadroniske kalorimeteret som måler energien fra de tyngre partiklene som produseres i kollisjonene, som eksempelvis protoner og nøytroner. Kalorimetrene måler disse egenskapene ved å stoppe partiklene og måle energien som frigjøres. Helt ytterst finner man myondetektorene som detekterer myoner[7].

[6] Bevegelsesmengden kan gi verdifull informasjon om partiklenes hastighet, retning og masse.
[7] Myon er en elementærpartikkel som tilhører den hovedgruppen av partikler som kalles leptoner. Se Figur 2-1 i punkt 2.1.1

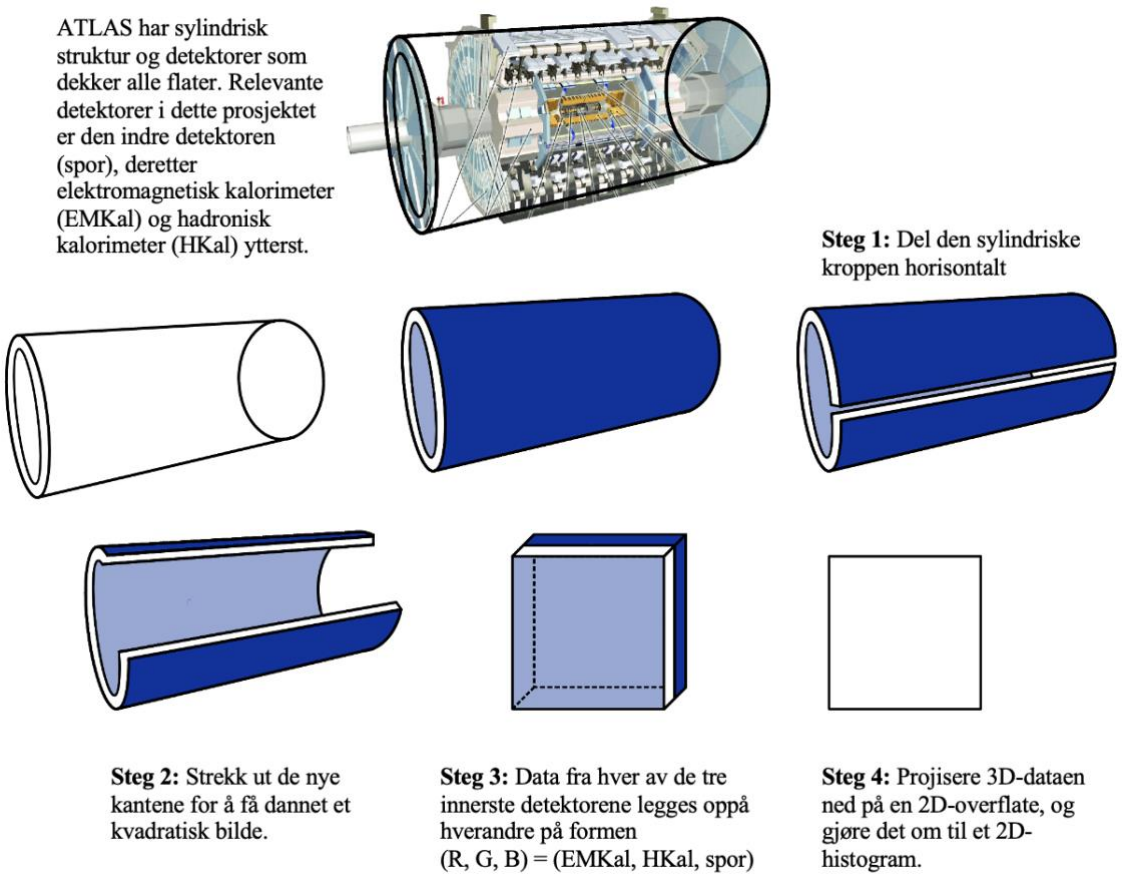


Figur 2-9 – Tverrsnitt av ATLAS som illustrerer hvilke under detektorer som identifiserer hvilke partikler fra partikkelsammenstøt / hvordan partiklene gir opphav til nye partikler i kollisjonene, som kan detekteres av ulike sub-detektorer (CDS, 2008).

Informasjonen fra underdetektorene gjør det mulig for datamaskiner å rekonstruere hendelsene som oppstår under partikkelsammenstøtene, og dermed kan man også oppdage nye partikler ved å identifisere nye signaturer som ikke er kjent fra før av. Et eksempel på dette er oppdagelsen av Higgs-bosonet, som resulterte i et stort gjennombrudd innen partikkelfysikk og bidro med å bekrefte standardmodellen for partikkelfysikk (ATLAS Collaboration, 2023c). Forskere søker også etter å bekrefte hypotetiske fenomener som mikroskopiske sorte hull og sfaleroner gjennom analyser av deres respektive signaturer, som beskrevet i punkt 2.1.2.

2.3 Utlevert datasett

ATLAS-detektoren er en sylindrisk struktur som detekterer kollisjoner langs alle sider og flater. På et sted i detektoren deles den sylindriske kroppen horisontalt, og de nye kantene strekkes ut for å danne et kvadratisk bilde, som illustrert i Figur 2-10. Datasettet i prosjektet består av slike rekonstruerte kollisjonsbilder fra ATLAS, generert av Aurora S. Grefsrud fra HVL sin ATLAS-gruppe ved hjelp av simuleringspakken og rammeverket, *Delphes*.

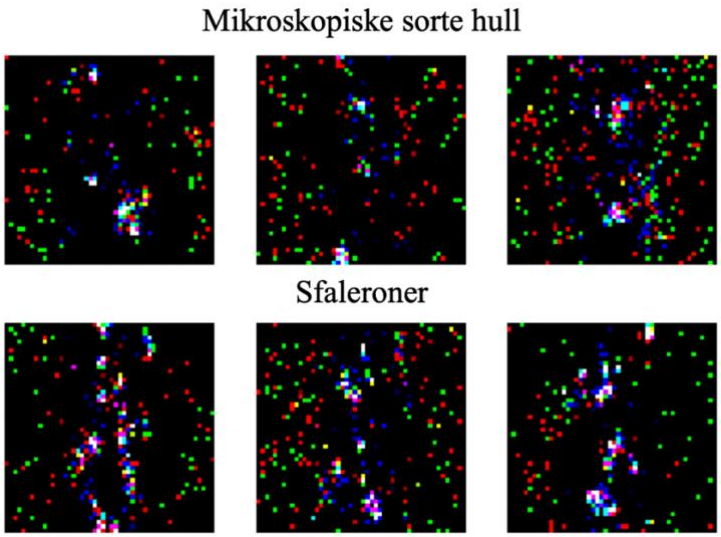


Figur 2-10 – Illustrasjon som viser prosessen for å lage 2D-histogram fra ATLAS-detektoren.

Datasettet er en samling av to filer, der hver fil består av 15000 kollisjonsbilder. Kollisjonsbildene har blitt generert fra simulerte kollisjonsdata fra ATLAS-detektoren, hvor den ene filen inneholder data som representerer mikroskopiske sorte hull og den andre filen inneholder data som representerer sfaleroner. Bildene har en oppløsning på 50x50 piksler og er av typen 2D-histogram[8]. Hvert kollisjonsbilde er sammensatt av tre fargekanaler plassert «oppå» hverandre, av typen RGB-farger, som vist i Figur 2-11. Det røde laget i bildene

[8] Et 2D-histogram er en grafisk representasjon av data som viser fordelingen av observasjoner eller verdier i to dimensjoner.

representerer data fra den elektromagnetiske kalorimeteren, det grønne laget representerer data fra den hadroniske kalorimeteren og det blå laget representerer data fra den indre detektoren av ATLAS.



Figur 2-11 – Et tilfeldig utvalg bilder fra datasettet som representerer mikroskopiske sorte hull (øverst) og sfaleroner (nederst)..

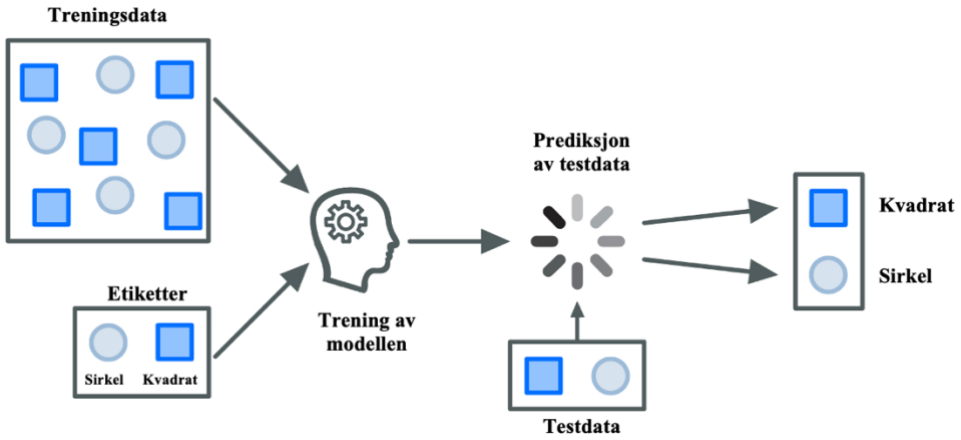
2.4 Maskinlæring

Teknologi har revolusjonert måten mennesker lever og arbeider på, og en spesielt spennende teknologi som har fått mye oppmerksomhet de siste årene er maskinlæring. Maskinlæring er en gren innen kunstig intelligens som gjør det mulig for datamaskiner å lære og forbedre seg selv ved å analysere mønstre og sammenhenger i data, uten å være eksplisitt programmert for hver enkelt oppgave (Brown, 2021). På denne måten kan man si at maskiner etterligner menneskelig intelligens. Maskinlæring kan videre deles inn tre hovedgrupper: veiledet maskinlæring (*Supervised machine learning),* ikke-veiledet maskinlæring *(Unsupervised machine learning)* og forsterket læring (*Reinforcement learning*).

2.4.1 Veiledet maskinlæring

Veiledet maskinlæring er en av de mest brukte metodene innen maskinlæring og utgjør hovedfokuset for dette prosjektet (Singh, 2019). På samme måte som mennesker lærer ved å se eksempler og tilhørende forklaringer, lærer også maskinlæringsmodeller i veiledet maskinlæring ved å bli gitt treningsdata med tilhørende etiketter (*labels*) (Singh, 2019a). Etikettene gir modellen informasjon om den riktige løsningen for hvert enkelt datapunkt i

treningssettet. Modellen bruker disse eksemplene til å justere sine egne parametere for å kunne gjøre de beste prediksjonene på ny og ukjent data. Figur 2-12 gir et eksempel på hvordan en maskin kan lære å klassifisere ulike figurer gjennom veiledet maskinlæring.



Figur 2-12 – Treningsdata er gitt med tilhørende etiketter for å muliggjøre veiledet maskinlæring.

Klassifisering og regresjon er to hovedgrupper innenfor veiledet maskinlæring (Singh, 2019). Klassifisering brukes når målet er å tilordne inndata til en bestemt kategori eller klasse, som i dette prosjektet vil være å forutsi om et bilde representerer et mikroskopisk sort hull eller et sfaleron. Regresjon, på den annen side, brukes når målet er å forutsi en numerisk verdi, som eksempelvis å forutsi prisen på et hus basert på de egenskapene det har.

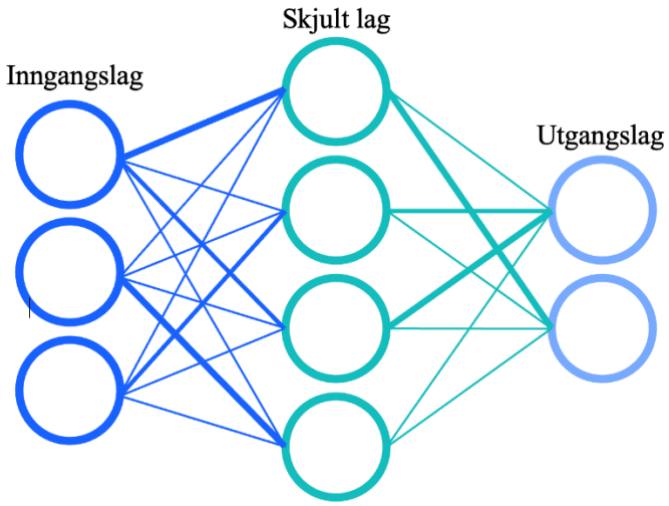
2.4.2 Ikke-veiledet maskinlæring og forsterket læring

Ikke-veiledet maskinlæring skiller seg fra veiledet maskinlæring ved at det ikke er gitt etiketter sammen med treningsdataen (Singh, 2019b). I stedet søker modellene å identifisere mønstre, relasjoner eller sammenhenger i data. Gruppering (*Clustering*) og assosiasjonsregler (*association rules*) er to hovedgrupper innen ikke-veiledet maskinlæring. Gruppering grupperer datapunkter med liknende egenskaper sammen, basert på avstanden mellom dem eller likheter den selv trekker ut fra treningsdataen. Assosiasjonsregler identifiserer assosiasjoner mellom variabler ved å analysere graden av avhengighet mellom dem.

Forsterket læring er en type maskinlæringsteknikk der en datamaskin lærer å ta beslutninger ved å utføre handlinger og observere resultatene i et miljø. Datamaskinen får positiv tilbakemelding for gode handlinger og negativ tilbakemelding for dårlige handlinger. Dermed fungerer det som et belønningssystem der en belønningsfunksjon maksimeres over tid.

2.4.3 Nevrale nettverk

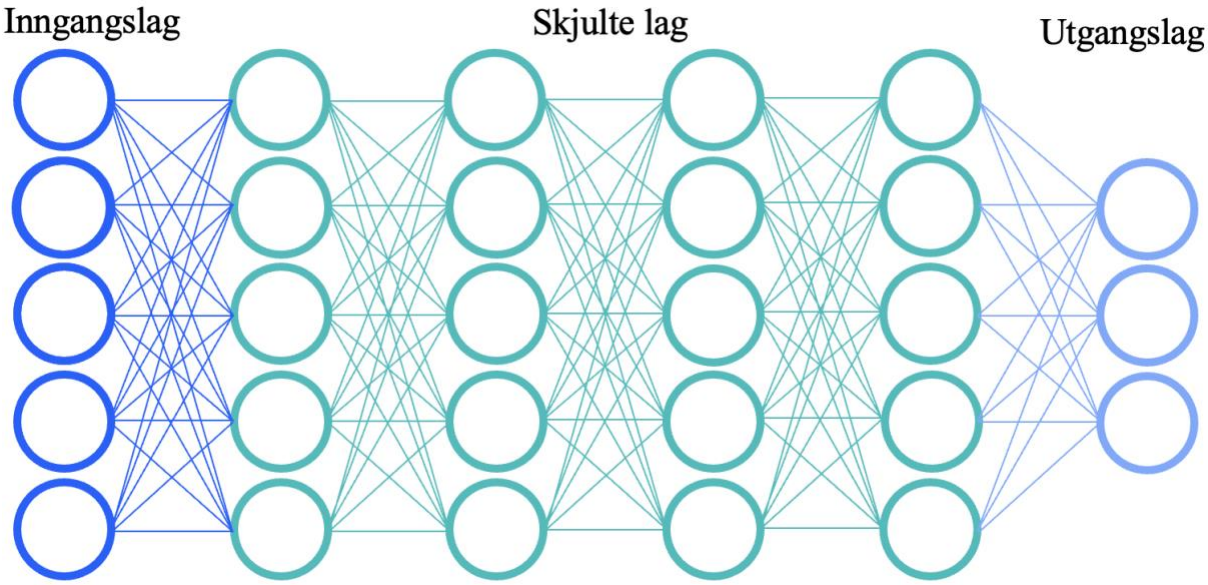
Et nevralt nettverk er en type maskinlæringsmodell som er inspirert av strukturen og funksjonaliteten til den menneskelige hjernen (IBM, 2023). I hjernen er det nevronene som har ansvaret for å motta og sende signaler, og disse signalene overføres via synapser, en type forbindelsessti, fra nervecelle til nervecelle (Faiz, 2023). I nevrale nettverk representeres nevronene som noder og synapsene som koblinger, slik Figur 2-13 illustrerer (Han mfl., 2018). Figur 2-13 viser også at nodene er organisert i flere lag. For å overføre informasjon fra et lag til et annet i nettverket, er hver node koblet til alle nodene i det foregående laget. Et nevralt nettverk består vanligvis av et inngangslag (*input layer*), et eller flere skjulte lag (*hidden layers*) og et utgangslag (*output layer*). Punkt 2.4.5 gir en mer detaljert beskrivelse av hvordan disse lagene samarbeider med hverandre for å oppnå læring av det nevrale nettverket.



Figur 2-13 – Et enkelt / grunt nevralt nettverk.

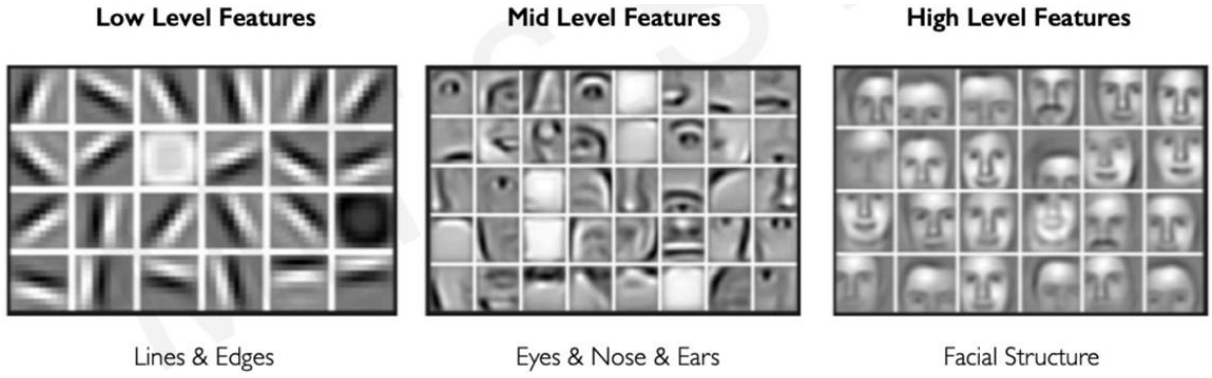
2.4.4 Dyp læring

Et nevralt nettverk kan være grunt, slik Figur 2-13 er et eksempel på, eller dypt, slik Figur 2-14 viser (IBM, 2023). Et grunt nettverk har kun ett til tre skjulte lag, mens et dypt nevralt nettverk har flere enn tre skjulte lag. Dyp læring er en gren av maskinlæring som fokuserer på å lære dype nevrale nettverk å utføre komplekse oppgaver fra store datasett. Dyp læring presterer godt innen bilde- og talegjenkjenning, klassifisering, og naturlig språkbehandling.



Figur 2-14 – Et dypt nevralt nettverk (inspirert av IBM, 2023).

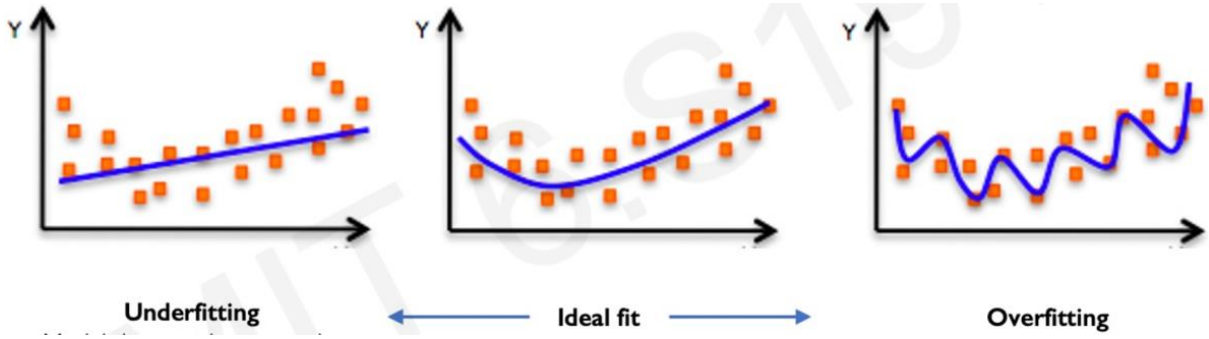
En av forskjellene mellom maskinlæring og dyp læring er hvordan modellen lærer å gjenkjenne mønstre i treningsdataen. I tradisjonell maskinlæring beskriver mennesker de karakteristiske egenskapene som modellen bruker til å gi prediksjoner (Amini, 2023). I dyp læring derimot, er det det dype nevrale nettverket som lærer å trekke ut slike karakteristiske egenskaper gjennom en automatisert læringsprosess. Læringsprosessen beskrives nærmere i avsnitt 2.4.5. Figur 2-15 illustrerer eksempler av stadig mer komplekse mønstre som kan trekkes ut fra treningsdataen i forbindelse med ansiktsgjenkjenning. I vanlig maskinlæring måtte mennesker ha beskrevet disse mønstrene, mens i dyp læring klarer den å lære disse selv.



Figur 2-15 – For at en modell skal kunne gjenkjenne ansikter, må den først lære seg enkle egenskaper i datasettet før den kan gå videre til mer avanserte egenskaper (Amini, 2023).

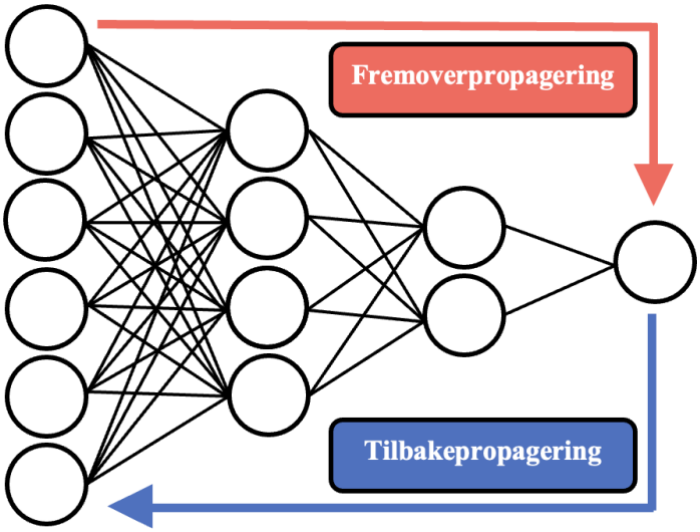
2.4.5 Trening og optimering

Når man trener et nevralt nettverk, er det flere konsepter og teknikker som spiller en viktig rolle for å oppnå gode resultater, og forhindre vanlige problemer som over- og undertilpasning (Raschka, 2020). Overtilpasning oppstår når modellen blir for kompleks og lærer å huske treningsdataene, i stedet for å generalisere og gjøre nøyaktige prediksjoner på ny data (Shorten og Khoshgoftaar, 2019). Undertilpasning oppstår når modellen er for enkel og ikke klarer å lære de underliggende mønstrene i treningsdataene. Både over- og undertilpasning fører til dårlig ytelse på ukjent data. For å unngå disse problemene, er det viktig å finne en balanse mellom modellkompleksitet og generaliseringskompleksitet. Figur 2-16 illustrerer problemene med over- og undertilpasning, der de røde firkantene representerer datapunkter og de blå strekene representerer maskinlæringsmodellen.



Figur 2-16 – Grafene viser undertilpasning t.v., ideal trening i midten og overtilpasning t.h. (Amini, 2023).

Læringsprosessen til et nevralt nettverk består av to hovedprosesser: fremoverpropagering (*Forward propagation*) og tilbakepropagering (*Backpropagation*), slik Figur 2-17 viser (Amini, 2023).



Figur 2-17 – Retningen til fremover- og tilbakepropagering i et nevralt nettverk.

Under fremoverpropagering overføres inndataen i inngangslaget til utgangslaget via de skjulte lagene. Koblingene mellom nodene i nettverket tildeles tilfeldige verdier for å konfigurere og klargjøre det for trening. Prediksjonene som produseres i utgangslaget brukes i tilbakepropagering til å trene nettverket. Ved å sammenligne de predikerte resultatene med de ønskede resultatene, justerer nettverket verdien av koblingene mellom nodene med sikte på å oppnå forbedrede resultater. Denne iterative prosessen fortsetter i flere sykluser, kjent som epoker (*epoch*), for å forbedre nettverkets ytelse.

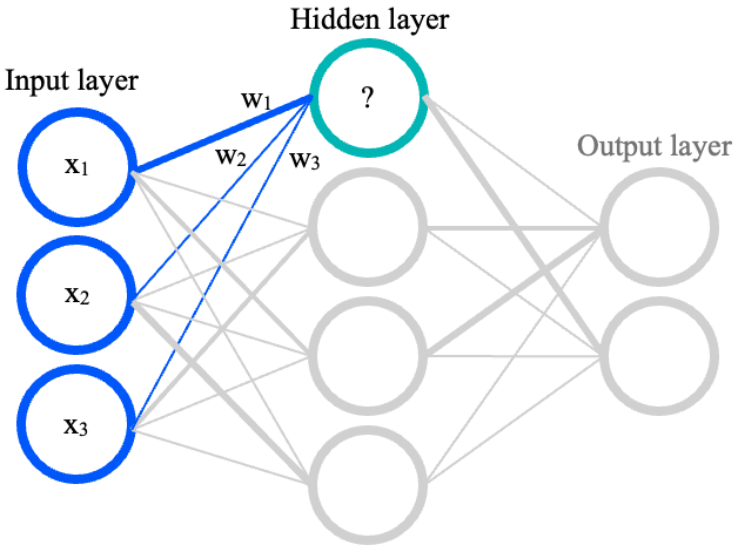
Antall epoker som brukes i maskinlæring er en hyperparameter som kan justeres av utvikleren (Brownlee, 2018). Å kjøre flere epoker kan bidra til å forbedre modellens nøyaktighet og generaliseringsevne. Imidlertid kan for mange epoker føre til overtilpasning til treningssettet.

2.4.5.1 Fremoverpropagering: Inngangslag

I inngangslaget mottar nodene inndata som nettverket skal lære av. Dette kan være tekst i kodet form, numeriske verdier, pikselverdier i bilder eller andre typer data. Nodene i inngangslaget er koblet til det neste laget i nettverket via koblinger (Amini, 2023). Koblingene er ansvarlige for å overføre informasjon fra den ene noden til den neste og har derfor en vekt (*weight*) som bestemmer styrken på forbindelsen mellom dem (Han mfl., 2018; IBM, 2023). Det vil si at det finnes en numerisk verdi som angir hvor mye signalet som overføres langs koblingen, påvirker den neste noden i nettverket. Jo høyere vekt, desto større innflytelse har signalet som overføres, som illustrert i Figur 2-18 med ulik tykkelse på koblingene. Hver node i det nevrale nettverket blir også tildelt en forskyvningsverdi (*bias*). Før nettverket er trent, vil vekten til koblingene og forskyvningsverdiene bli gitt som tilfeldige verdier. I begynnelsen har ikke disse verdiene særlig betydning for nettverket, da de er læringsverdier (*learning values*) som vil bli forbedret senere i læringsprosessen. Overgangen fra nodene i inngangslaget til det første skjulte laget foregår ved hjelp av vektet sum. Summen av vektet inndata blir beregnet ved å legge sammen summen av alle nodene, *x*, i inngangslaget multiplisert med sine tilhørende vekter, *w*:

$$(w_1 x_1 + w_2 x_2 + \cdots + w_n x_n)$$

Figur 2-18 viser et eksempel på hvilke deler av nettverket som bidrar i å bestemme inndataen til den første noden i det første skjulte laget. Tilsvarende beregninger skjer for alle nodene i nettverket.



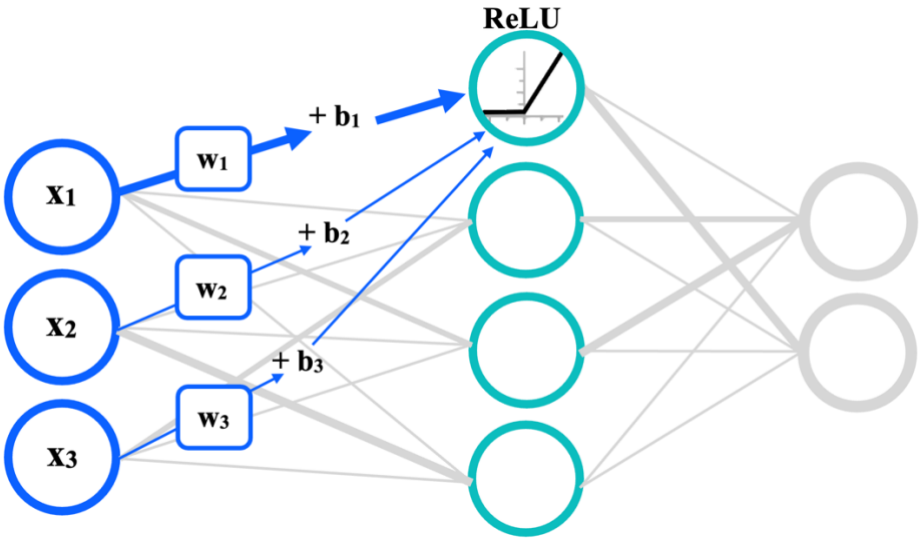
Figur 2-18 – Beregning av vektet inndata til noden markert med spørsmålstegn.

2.4.5.2 Fremoverpropagering: De skjulte lagene

Hvert nevron i det skjulte laget mottar en vektet sum av inndata fra det forrige laget, som deretter overføres til en aktiveringsfunksjon (Han mfl., 2018; Amini, 2023). Før den vektede summen overføres til aktiveringsfunksjonen, g, legges forskyvningsverdien, b, til:

$$g(w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b)$$

Figur 2-19 viser en oppdatert versjon av Figur 19 som inkluderer forskyvningsverdien og ReLU-aktiveringsfunksjon, i tillegg til vektene.



Figur 2-19 – Oppdatert nettverk der forskyvningsverdier og aktiveringsfunksjonen, ReLU, er lagt til.

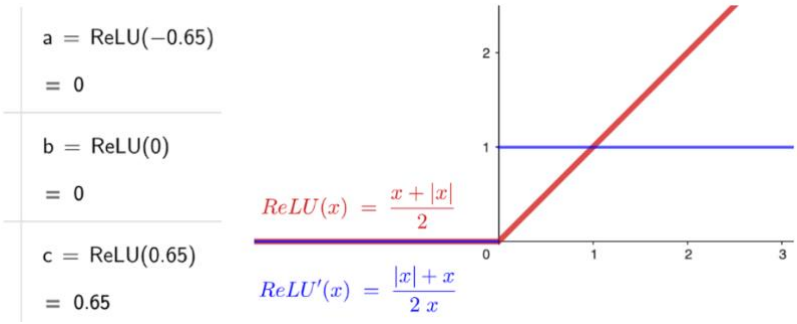
Dersom den vektede summen i noden overskrider en viss terskelverdi, vil aktiverings-funksjonen bli trigget (IBM, 2023). Aktiveringsfunksjoner brukes for å bestemme om noden skal aktiveres eller ikke, og dermed bestemme om noden skal kunne sende signalet videre til det neste laget i nettverket. Det er denne prosessen med å aktivere og overføre signaler mellom nodene som gjør at nevrale nettverk kan lære å gjenkjenne mønstre og utføre oppgaver. Formålet med å legge til en forskyvningsverdi i den vektede summen er å justere hvor mye påvirkning de vektede inndataene har på aktiveringsfunksjonen (Goodfellow, Bengio og Courville, 2016, s. 226; Howard og Gugger, 2020, s. 164-165). En høy forskyvningsverdi vil føre til en lavere terskelverdi for aktivering, og omvendt. Forskyvningen medfører økt fleksibilitet i nettverket i forbindelse med tilpasning av parameterne senere i læringsprosessen, fordi den får et bredere spekter for hvilke verdier som blir ansett som aktivert og ikke.

Rectified Linear Unit (ReLU) er et eksempel på en aktiveringsfunksjon som brukes i nevrale nettverk (Zeiler mfl., 2013; Nwankpa mfl., 2018). Den fungerer ved å sette alle negative inndata til null, mens den lar positiv inndata passere uendret. Dette gjelder for alle typer inngangsdata, ikke bare bilder. ReLU er ofte det foretrukne valget for mange applikasjoner innen dyp læring, og dette skyldes flere faktorer. En av hovedgrunnene er enkelheten i funksjonsuttrykket, som gjør den lett å forstå og implementere. Den krever ikke komplekse beregninger eller eksponentielle funksjoner, noe som gjør den beregningsmessig effektiv og rask å evaluere. En annen fordel er dens evne til å redusere overtilpasning. Ved å sette node med negativ inndata til null, vil det deaktivere noden slik at den ikke kan sende signal videre til nettverket. Dette fører til en mer sparsommelig representasjon i det nevrale nettverket som igjen er med på å redusere kompleksiteten og overtilpasningen. ReLU tilbyr også en løsning på et problem i dyp læring kjent som forsvinnende gradient (*vanishing gradient*). Dette problemet oppstår når gradientene blir svært liten eller avtar betydelig under tilbakepropagering, noe som resulterer i ineffektiv oppdatering av vektene i tidligere lag under trening. Ved å sette negative verdier til null, forhindrer aktiveringsfunksjonen at gradienten blir forsvinnende liten i de dype lagene av nettverket.

ReLU er definert som maksimum av null og inndataen, *x*, og kan derfor uttrykkes gjennom følgende funksjonsuttrykk (Nwankpa mfl., 2018):

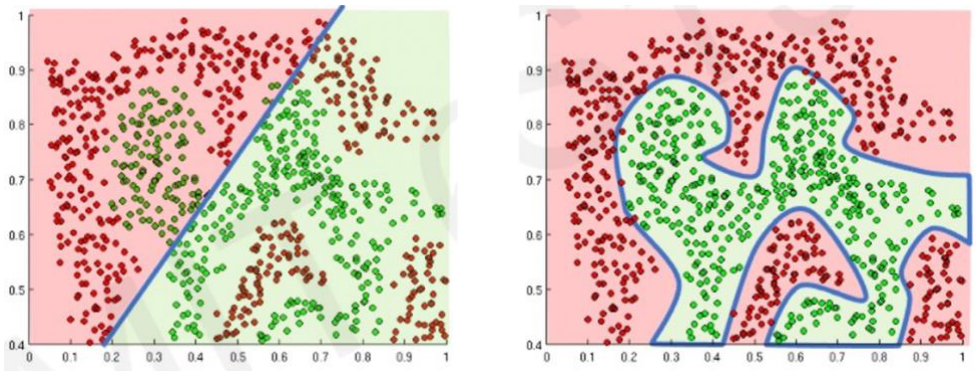
$$ReLU(x) = max(0, x) = \begin{cases} x_i, if\ x_i > 0 \\ 0, if\ x_j < 0 \end{cases}$$

Figur 2-20 viser grafen til ReLU-funksjonen (rød). Her kan man se en «knekk» i koordinatpunktet (0,0), som indikerer at funksjonen er ikke-deriverbar i dette punktet. Dette kan bli et problem for de senere beregningene som er avhengige av derivasjon (se punkt 2.4.5.4). Derfor er den deriverte av funksjonsuttrykket (blå graf) i dette punktet definert som enten 0 eller 1. Dette sikrer at læringen kan fortsette på tross av diskontinuiteten.



Figur 2-20 – Noen eksempler ved bruk av ReLU-funksjonen (t.v.), samt grafen til ReLU-funksjonen og den deriverte av ReLU.

Aktiveringsfunksjoner brukes også for å introdusere ikke-linearitet i modellen (Amini, 2023). Dette er nødvendig for å modellere komplekse sammenhenger mellom inndata og utdata, ettersom disse sammenhengene ikke kan beskrives med en rett linje eller matematiske funksjoner. I stedet kreves det mer komplekse funksjoner som kan tilpasse seg ulike mønstre og trekk i inndataene. Figur 2-21 gir en visuell fremstilling av hvorfor det er nødvendig med ikke-linearitet i nevrale nettverk.



Figur 2-21 – Lineære aktiveringsfunksjoner (t.v.) produserer lineære valg, mens ikke-lineære funksjoner (t.h.) tillater modellen å tilnærme vilkårlige komplekse funksjoner (Amini, 2023).

2.4.5.3 Fremoverpropagering: Utgangslaget

Utgangslaget i et nevralt nettverk består av en gruppe noder som produserer utdata basert på informasjonen som samles opp gjennom nettverket (Teuwen og Moriakov, 2020; IBM, 2023). Hver node i utgangslaget er koblet til alle noder i det siste skjulte laget, og henter ut informasjon fra disse nodene gjennom vektede forbindelser. Avhengig av hva nettverket er trent til å gjøre, kan de predikerte verdiene eksempelvis være tall, kategorier eller sannsynligheter. I dette prosjektet skal det nevrale nettverket brukes til å klassifisere inndata, mens i andre tilfeller kan det brukes til å forutsi verdier eller generere respons.

I første epoken av læringen av det nevrale nettverket, er det fortsatt ikke trent. Det betyr at den første utgangsverdien som man får i dette steget vanligvis ikke gir noen særlig god prediksjon på inndataen som ble gitt. I den neste fasen av læringsprosessen vil nettverket bli trent.

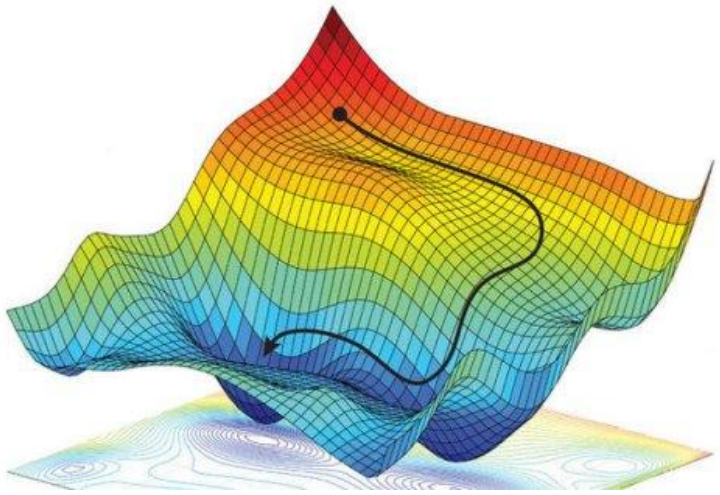
2.4.5.4 Tilbakepropagering

Etter fremoverpropagering blir det utført en evaluering av den predikerte verdien fra utgangslaget (Murugan, 2017). Hensikten med evalueringen er å fastslå om nettverket har gjort feil eller ikke. Hvis det viser seg at de predikerte verdiene fra nettverket ikke stemmer overens med forventet verdi, blir denne informasjonen sendt tilbake til de skjulte lagene slik at vektene og forskyvningsverdiene kan justeres. Dette er årsaken til at disse verdiene kunne bli tildelt tilfeldig før nettverket var trent, ettersom verdiene justeres etter behov gjennom tilbakepropagering. Tilbakepropagering er derfor årsaken til at nevrale nettverk kan forbedre («lære») seg selv uten menneskelig innblanding. Omfanget av feilen er gitt av tapet (*loss*) eller kostnaden som beregnes av tapsfunksjonen (*loss function*) (Howard og Gugger, 2020, s. 159). Tapsfunksjonen er en matematisk funksjon som beregner nettverkets feilaktige prediksjoner ved å sammenligne disse med de forventede verdiene for nettverket. Funksjonen fungerer dermed som et verktøy for å evaluere ytelsen til det nevrale nettverket.

Det finnes ulike typer tapsfunksjoner avhengig av hvilket problem det nevrale nettverket forsøker å løse. Kryssentropi (*cross entropy loss*) er en tapsfunksjon som er mye brukt i klassifiseringsoppgaver (Kline og Berardi, 2005). Funksjonen måler hvor godt nettverket tilordner riktige klasser til inndataen. Kvadratisk tap (*quadratic loss* eller *mean squared error*) er en tapsfunksjon som ofte blir brukt i regresjonsoppgaver (Howard og Gugger, 2020, s. 159).

Funksjonen måler avstanden mellom den faktiske verdien og nettverkets predikerte verdi, og er en essensiell del av læringsprosessen.

Felles for alle tapsfunksjoner er at de skal forsøke å minimere[9] feilen for alle datapunkter i nettverkets datasett. Det innbefatter å finne det optimale settet av vekter mellom nodene som gir det minst mulige tapet over hele datasettet. Det minste tapet tilsvarer den globale minimumsverdien i 3D-grafen gitt i Figur 2-22, hvilket er det mest optimale punktet som er ønskelig å finne.



Figur 2-22 – 3D-grafen viser hvordan gradientnedstigning brukes for å bevege seg i flerdimensjonalt rom av parametere for å minimere en tapsfunksjon (Amini, Soleimany, Karaman, og Rus, 2018).

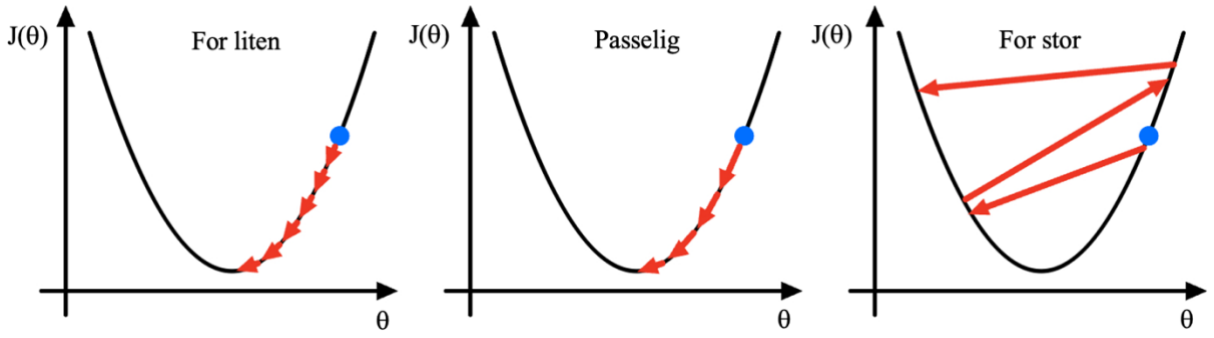
For å finne den globale minimumsverdien på tapsfunksjonen brukes en optimeringsalgoritme kalt gradientnedstigning (*gradient descent*). Det første trinnet i algoritmen er å velge en startposisjon på funksjonen, ofte tilfeldig valgt eller basert på tidligere kunnskap om funksjonen. Deretter beregnes tapet i det nevrale nettverket på denne plassen ved å regne ut gradienten til tapsfunksjonen. Gradienten er den partielle deriverte av tapsfunksjonen med hensyn på alle modellparameterne, inkludert både vektene og forskyvningsverdiene.

Gradienten til tapsfunksjonen gir informasjon om hvor raskt tapet øker i hver retning fra gjeldende vektverdier. Følgelig viser den også retningen som gir størst økning i tapet. Ved å gå

[9] Å minimere en funksjon betyr å finne det laveste punktet på funksjonskurven.

i motsatt retning av gradienten vil man dermed kunne bevege seg ett steg nærmere bunnpunktet. Prosessen med å beregne gradienten på den nye plassen og ta ett steg i den motsatte retningen, gjentas frem til tapsfunksjonen konvergerer til et minimum eller et forhåndsbestemt stopp-kriterium er oppfylt. Å konvergere til et minimum betyr at man justerer vektene og forskyvingsverdiene slik at tapet gradvis reduseres til et minimum. Dette kan sees som å finne et sted i parameterrommet der tapsfunksjonen ikke kan reduseres ytterligere ved å justere parametere i nettverket.

Størrelsen på steget man tar i motsatt retning av gradienten bestemmes av læringshastighet (*learning rate*) (Howard og Gugger, 2020, s. 156-157). Det kan være mange utfordringer knyttet til å sette en passende læringsrate, som visualisert i Figur 2-23. Dersom læringsraten er for liten, vil man konvergere sakte og man kan bli sittende fast i et lokalt minimum. Dersom læringsraten er for stor kan det være man «hopper» rett forbi et globalt minimum og dermed avvike fra den optimale løsningen. Det mest optimale er å finne en slags middelverdi der læringsraten er stor nok til at man kommer seg forbi de lokale minimum, men ikke for stor til at man begynner å avvike fra løsningen.



Figur 2-23 – Representasjoner av ulik læringsrate.

Mens gradientnedstigning i teorien søker å finne den globale minimumsverdien av tapsfunksjonen, vil den ofte konvergere til en lokal minimumsverdi i praksis. En lokal minimumsverdi representerer den laveste verdien et lokalt område av parameterrommet. Likevel er det mulig å oppnå en løsning som er tilstrekkelig nær den globale minimumsverdien for de fleste praktiske formål. Dessuten finnes det en rekke andre optimeringsmetoder som kan brukes til å finne globalt minimum i komplekse funksjonsrom. I punkt 2.4.5.5 diskuteres noen alternative optimeringsmetoder til gradientnedstigning.

Oppsummert lærer nevrale nettverk gjennom fremoverpropagering, beregning av tapet ved hjelp av en tapsfunksjon, gradientnedstigning og tilbakepropagering. Disse prosessene samarbeider for å justere vektene og forskyvningsverdiene i nettverket basert på inndataene og forventet utgang. Ved å gå gjennom treningsdataene flere ganger og justere parameterne etter behov, kan nevrale nettverk lære å klassifisere eller predikere resultater med høy nøyaktighet. Avanserte teknikker og optimeringsalgoritmer som beskrives i de påfølgende punktene har potensial til ytterligere forbedre ytelsen, og hjelpe med å finne optimale løsninger for problemet nettverket prøver å løse.

2.4.5.5 Andre optimeringsmetoder

Gradientnedstigning er en av de vanligste optimeringsalgoritmene som brukes for å optimere nevrale nettverk (Ruder, 2017). Imidlertid er det flere andre optimeringsmetoder som kan brukes for å forbedre treningen av et nettverk. Optimeringsalgoritmene har som mål å finne den optimale parameterkombinasjonen som gir den beste prediksjonen av utdataene fra inndataene. Det vil si å finne de parameterne som gir minst feil i modellen.

En av de mest populære optimeringsmetodene er Adam (*Adaptive Moment Estimation*) (Ruder, 2017). Adam tilpasser læringsraten til hver parameter i nettverket basert på informasjon om hvordan tidligere gradienter har endret seg over tid. Matematisk sett beregner Adam et eksponentielt glidende gjennomsnitt[10] (EMA) av både gradientene og kvadratene av gradientene. Dette gjør at metoden kan tilpasse seg endringer i gradienten og justere læringsraten på en adaptiv måte for å forbedre konvergenshastigheten til det nevrale nettverket. Det vil si at Adam kan øke læringsraten for parametere som har en høyere gradientverdi, og redusere læringsraten for parametere med lavere gradientverdi. Dette kan føre til raskere trening av modellen og redusere risikoen for at nettverket kan «sette seg fast» i et lokalt minimum.

En annen optimeringsmetode er RMSprop (*Root Mean Square Propagation*) (Ruder, 2017). RMSprop justerer også læringsraten for hver parameter individuelt, men basert kun på et EMA av kvadratene av gradientene og utelater EMA av gradientene selv. Det vil si at metoden kun

[10] Et glidende gjennomsnitt er snittet av flere målinger av en verdi. I denne sammenhengen er verdien gradientene til en bestemt parameter i modellen. Gradientene varierer fra treningssekvens til treningssekvens, og et glidende gjennomsnitt kan gi en jevnere måling av hvordan gradientene endres over tid. Et eksponentielt glidende gjennomsnitt regner også ut snittet av flere målinger av en verdi, men det vekter de siste verdiene mer fremfor å vekte alle verdier like mye.

tar hensyn til endringene i gradientenes størrelse over tid, og ikke gradientenes absoluttverdier. Likevel bidrar også RMSprop til å forbedre konvergenshastigheten og treningen av modellen.

En tredje optimeringsmetode er Adagrad (*Adaptive Gradient*), som også tilpasser læringsraten for hver parameter i nettverket, men på en annen måte enn både Adam og RMSprop (Ruder, 2017). I Adagrad beregnes et akkumulert kvadrat av gradientene, hvilket betyr at kvadratene til tidligere beregnede gradientverdier summeres. For å unngå at læringsraten blir for stor og fører til ustabil trening, vil læringsraten til en parameter gradvis reduseres etter hvert som den akkumulerte kvadratsummen av gradienten øker. Dette betyr at parametere som har blitt oppdatert sjeldnere vil ha en lavere kvadratsum og dermed en høyere læringsrate enn parametere som har blitt oppdatert oftere. Adagrad kan bidra til å balansere læringen mellom ulike parametere i modellen, hvilket kan forbedre stabiliteten til nettverket og konvergenshastigheten.

I tillegg til «generelle» optimeringsmetoder, finnes det også optimeringsmetoder som skal forbedre disse igjen. AdaDelta er en forbedring av Adagrad, som skal sørge for at læringsraten ikke blir for redusert etter hvert som treningen skrider frem, hvilket kan føre til at modellen til slutt stopper å lære (Ruder, 2017). Adamax er en forbedring av Adam, som kan være med på å forhindre ustabil trening grunnet for store gradientverdier.

2.4.5.6 Regularisering

Regularisering (*regularization*) er en teknikk som brukes for å redusere overtilpasning i nevrale nettverk. Dette oppnås ved å legge til en straff i tapsfunksjonen, med sikte på å begrense modellens kompleksitet for å unngå overtilpasning. Straffen kan beregnes ved å ta summen av kvadratene av vektene (L2-regularisering, også kjent som vekt fall *weight decay*) eller summen av absoluttverdiene av vektene (L1-regularisering). Ved å legge til denne straffen, tvinges modellen til å velge enklere vekter og dermed øke generaliseringsevnen.

En annen populær teknikk som kan hjelpe til med å redusere overtilpasning er tilfeldig utkobling (*dropout*). Denne teknikken går ut på å tilfeldig deaktivere noen av nodene i nettverket under trening. Tilfeldig utkobling kan bidra til å tvinge modellen til å lære mer robuste og uavhengige mønstre og egenskaper, som kan forbedre generaliseringsmulighetene. Denne teknikken kan også bidra til å unngå at nettverket blir for avhengig av enkelte noder, og dermed øke variasjonen i de egenskapene som modellen lærer.

2.4.6 Evaluering

Evaluering er en viktig del av utviklingsprosessen for maskinlæringsmodeller, ettersom det hjelper med å vurdere ytelsen til en trent modell (Raschka, 2020). Det finnes forskjellige måter å evaluere et nevralt nettverk på, avhengig av oppgaven det er ment å løse.

I binære klassifiseringsoppgaver, slik dette prosjektet er, finnes det fire typer utfall av maskinlæringsmodellen (Ghosh mfl., 2022). De fire utfallene er vanligvis referert til som *true positive* (TP), *false positive* (FP), *true negative* (TN) og *false negative* (FN):

- TP betyr at modellen har korrekt predikert en positiv tilstand, for eksempel at modellen klassifiserer et bilde som et mikroskopisk sort hull, og at det faktisk er et mikroskopisk sort hull.
- FP betyr at modellen feilaktig predikerte en positiv tilstand, for eksempel at modellen klassifiserer et bilde til å være et mikroskopisk sort hull, men at det i grunnen er et sfaleron.
- TN betyr at modellen har korrekt predikert en negativ tilstand, for eksempel at modellen klassifiserer et bilde som et sfaleron, og at det faktisk er et sfaleron.
- FN betyr at modellen feilaktig predikerte en negativ tilstand, for eksempel at modellen klassifiserer et bilde som et sfaleron, men at det i grunnen er et mikroskopisk sort hull.

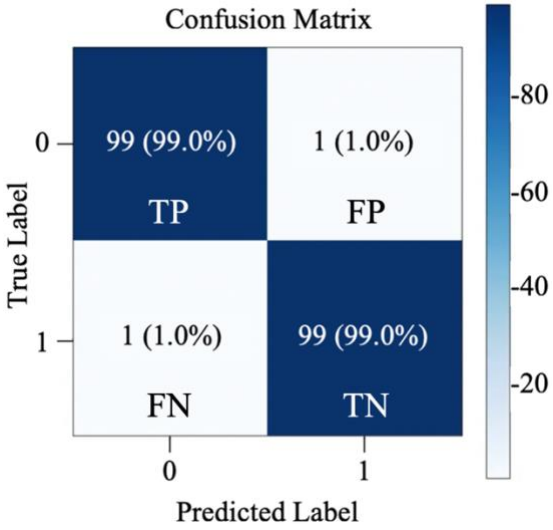
De fire utfallene fra en binær klassifikasjon kan organiseres i en matrise for å bedre visualisere hva disse betyr. Figur 2-24 viser eksempelet brukt i forklaringene av de fire utfallene, der mikroskopisk sort hull (blå) representerer positiv tilstand og sfaleron (rød) representerer negativ tilstand (ikke mikroskopisk sort hull).

	Actual Values	
Predicted Values	Pred: microscopic black hole Actual: microscopic black hole **True positive**	Pred: microscopic black hole Actual: sphaleron **False positive**
	Pred: sphaleron Actual: microscopic black hole **False negative**	Pred: sphaleron Actual: sphaleron **True negative**

Figur 2-24 – De fire utfallene i en binær klassifisering, visualisert med eksempelet om mikroskopiske sorte hull (positiv tilstand) og sfaleroner (negativ tilstand).

En slik matrise som vist i Figur 2-24 ovenfor, kalles en forvirringsmatrise (*confusion matrix*) (Howard og Gugger, 2020, s. 76). Forvirringsmatriser brukes til å evaluere ytelsen til en maskinlæringsmodell i en klassifiseringsoppgave, og vanligvis visualiseres de slik Figur 2-25

viser. Den gir en oversikt over antall riktige og antall feilaktige prediksjoner for hver klasse i problemet, og viser hvordan modellen klarer å skille mellom de forskjellige klassene. En forvirringsmatrise kan videre brukes til å beregne forskjellige metrikker (*metrics*).



Figur 2-25 – En forvirringsmatrise.

Sensitivitet (*sensitivity*) eller *(recall)* måler modellens evne til å korrekt identifisere alle positive tilfeller i et datasett og utrykkes med et tall fra 0 til 1.00, hvor en høyere andel indikerer bedre resultater. Sensitivitet kan være en viktig metrikk i tilfeller hvor det er viktig å unngå falske negative prediksjoner. Et slikt tilfelle kan eksempelvis være i medisinsk diagnostikk, der en negativ prediksjon kan føre til at en sykdom ikke blir oppdaget og behandlet. Sensitiviteten beregnes ved å dele antallet riktige positive prediksjoner (TP) på summen av riktige positive og falske negative prediksjoner (FN), og den matematiske formelen ser slik ut:

$$Sensitivitet = \frac{TP}{TP + FN}$$

Nøyaktighet (*accuracy*) uttrykkes vanligvis i prosent, hvor høyere prosentandel indikerer bedre resultater. Den måler andelen korrekte prediksjoner av den totale mengden prediksjoner. Det er viktig å merke seg at nøyaktighet kan være misvisende i tilfeller hvor datasettet er ubalansert. Det vil si at en klasse har mye flere eksempler enn den andre. Derfor brukes nøyaktighet ofte sammen med andre prestasjonsmål for å evaluere modellen. Nøyaktigheten beregnes ved å dele antall korrekte prediksjoner (TP + TN) på summen av antall riktige og feilaktige (TP + FP + TN + FN), og den matematiske formelen ser slik ut:

$$Nøyaktighet = \frac{TP + TN}{TP + FP + TN + FN}$$

Presisjon (*precision*) måler andelen korrekte positive prediksjoner i forhold til totalt antall positive prediksjoner og utrykkes med et tall fra 0 til 1.00, hvor en høyere andel indikerer bedre resultater.. Presisjon kan være en viktig metrikk i tilfeller hvor det er viktig å unngå falske positive prediksjoner. Et slikt tilfelle kan være i medisinsk diagnostikk, da det er uønsket å få falske positive resultater. Sensitivitet og presisjon vil derfor ofte være i konflikt med hverandre, ettersom en økning i sensitivitet kan medføre en reduksjon i presisjon og omvendt. Presisjon beregnes ved å dele antall riktige positive prediksjoner (TP) med summen av riktige positive og falske positive prediksjoner (TP + FP), og den matematiske formelen ser slik ut:

$$Presisjon = \frac{TP}{TP + FP}$$

2.4.7 Data augmentering

Data augmentering er en teknikk innen maskinlæring som brukes til å øke mengden tilgjengelig treningsdata (Shorten og Khoshgoftaar, 2019). Dette oppnås ved å lage nye eksempler basert på eksisterende data, men med små endringer som bevarer de grunnleggende egenskapene til originaldataene.

En av fordelene med data augmentering er at den gir modellene muligheten til å lære mer effektivt ved å eksponere dem for forskjellige varianter av treningsdata (Shorten og Khoshgoftaar, 2019). Gjennom å øke mangfoldet og variasjonen i treningsdataene, kan data augmentering bidra til å redusere overtilpasning og forbedre modellens generelle ytelse.

En annen fordel med data augmentering er at den øker modellens robusthet mot støy og andre forstyrrelser i inndataene. Ved å introdusere flere variasjoner i treningsdataen, kan modellen lære å gjenkjenne mønstre og egenskaper som er felles for ulike varianter av treningsdata, og dermed forbedre dens evne til å generalisere til nye og ukjente data.

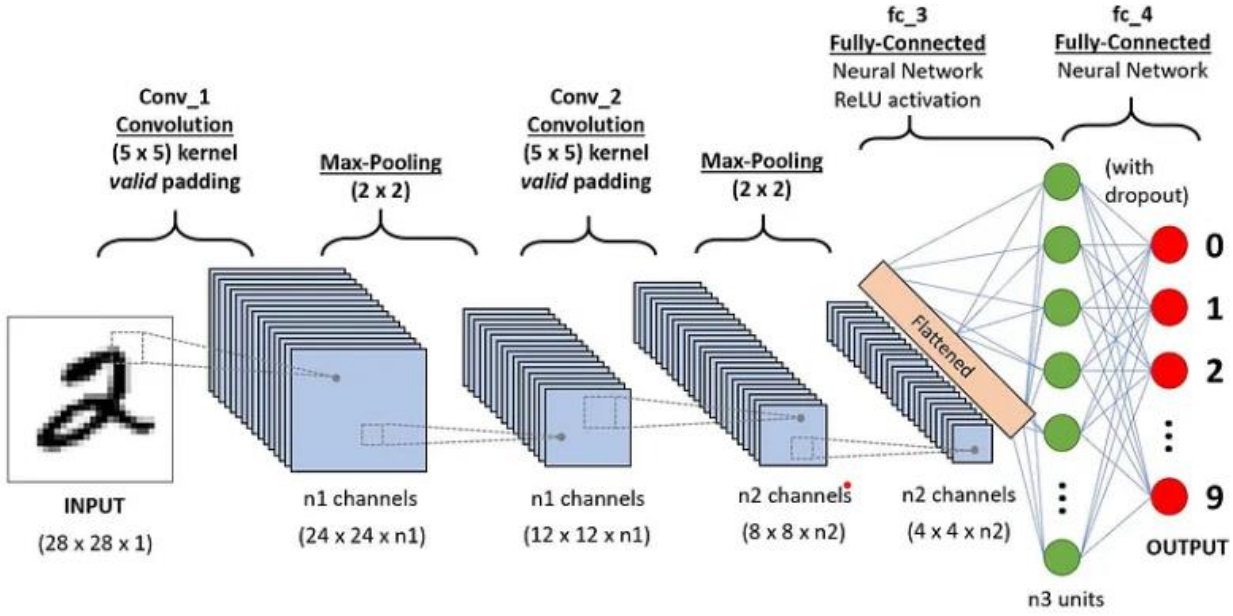
Det finnes flere forskjellige teknikker for data augmentering som er egnet for ulike typer data. Noen teknikker inkluderer å bruke geometriske transformasjoner, som eksempelvis rotasjon, skalering, speiling, beskjæring og endring av perspektiv for å tilføye flere eksempler som modellen kan trenes på. Andre teknikker går ut på å gjøre fargeendringer, som for eksempel å endre lysstyrke, kontrast, metning og fargetone. I tillegg til disse grunnleggende teknikkene kan også mer avanserte metoder brukes, som støytilsetning, elastiske deformasjoner og generative modeller. De avanserte teknikkene kan hjelpe modellen å lære mer om de underliggende

mønstrene i et datasett, og dermed forbedre dens evne til å generalisere til nye og ukjente data. Det er også mulig å kombinere ulike teknikker for data augmentering for å skape en mer robust og effektiv modell som er i stand til å takle et bredere spekter av utfordringer.

Imidlertid kan data augmentering også innføre en skjevhet (*bias*) i modellen hvis det ikke er utført på en hensiktsmessig måte. Når en modell har skjevhet betyr det at den har en tendens til å produsere unøyaktige eller feilaktige resultater på grunn av systematiske feil eller tendenser i datasettet eller modellen. Data augmentering kan introdusere skjevhet i modellen ved at den får modellen til å fokusere på andre ting enn selve innholdet. Slik kan en teknikk som rotering av bilder føre til at modellen lærer å klassifisere bilder basert på orienteringen, og ikke på innholdet i bildet. På samme måte kan fargeendringer føre til at modellen blir mer opptatt av bestemte fargetoner enn andre. Det er derfor viktig å være oppmerksom på potensiell skjevhet som kan innføres ved bruk av data augmentering, og å velge teknikker som ikke skaper skjevhet i modellens læring.

2.4.8 Konvolusjonelle nevrale nettverk

Konvolusjonelle nevrale nettverk (*Convolutional Neural Networks*) er avhengige av konsepter fra lineær algebra for å kunne identifisere mønstre i bilder og løse komplekse oppgaver (IBM, 2023). Derfor er det en populær dyplæringsmodell for å løse komplekse oppgaver som eksempelvis objekt-deteksjon, bildesegmentering og ansiktsgjenkjenning. Figur 2-26 viser strukturen til en CNN-modell, og i de etterfølgende punktene beskrives de viktigste komponentene.

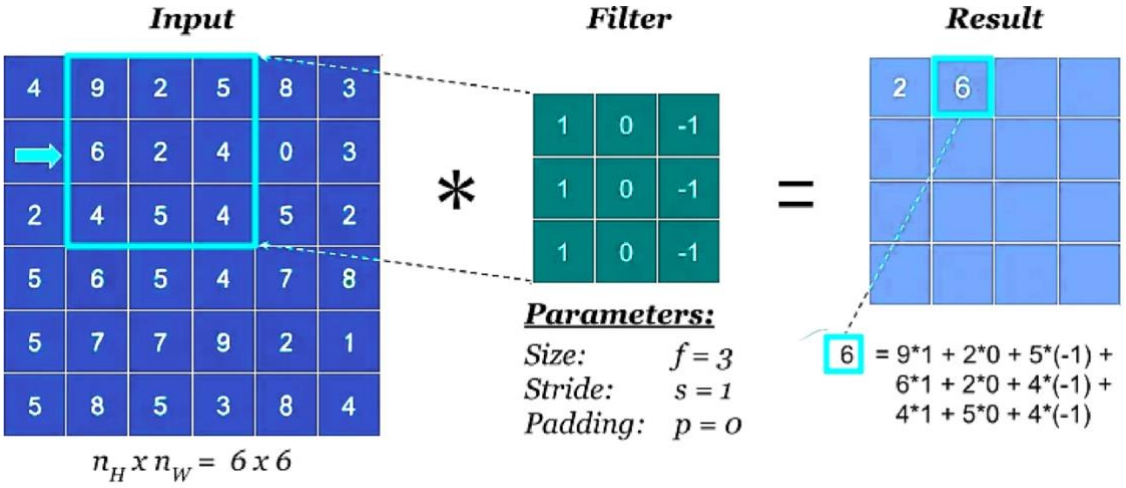


Figur 2-26 – Strukturen til en standard CNN-modell (Saha, 2018).

2.4.8.1 Konvolusjonslag

Konvolusjonelle nevrale nettverk bruker konvolusjonslag (*convolutional layers*) for å bearbeide inndataen. Justerbare *filtre* eller *kjerner*, illustrert av den lyseblå firkanten i Figur 2-27, er små matriser som representerer ulike mønstre den kan trekke ut fra treningsdataen. Disse matrisene beveger seg over hver pikselverdi i inndataen og utfører indreprodukt på hver posisjon. Dette resulterer i et egenskapskart (f*eature map*) over egenskaper som inneholder informasjon om mønstrene og strukturene i inndataene.

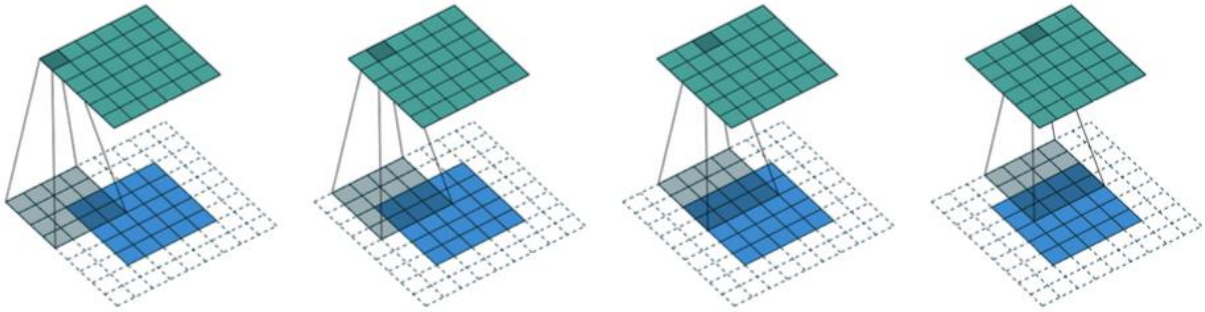
Indreproduktet mellom pikselverdiene og filtermatrisen i inndataen er en sentral operasjon i konvolusjonslagene. Figur 2-27 gir et eksempel på denne prosessen. Indreproduktet beregner en skalarverdi ved å multiplisere hver pikselverdi i inndataen med en verdi i filtermatrisen, og deretter summere produktene. Resultatet av denne summeringen er en enkelt verdi i egenskapskartet.



Figur 2-27 – Filteret flyttes fra venstre til høyre for å produsere et kart over spesifikke egenskaper funnet i det originale bildet (Convolutional Neural Networks Explained [CNN Visualized], 2020).

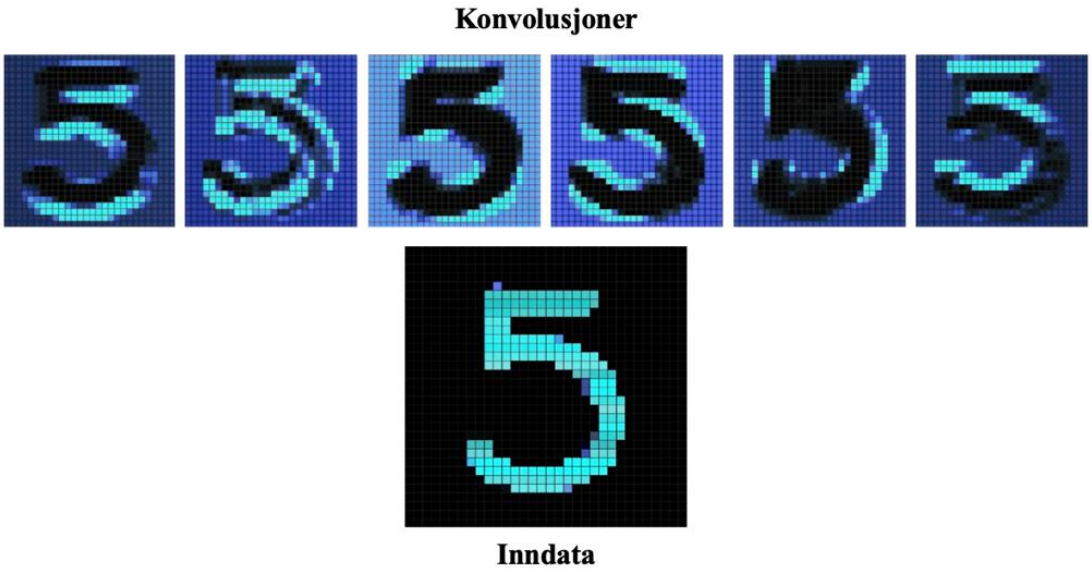
Filtermatrisen har tre parametere som er med på å bestemme størrelsen og egenskapene til egenskapskartet. Disse parameterne inkluderer størrelsen på filteret (*size*), steglengde (*stride*) og utfylling (*padding*). Størrelsen på filteret refererer til dimensjonene til filteret som beveger seg over inngangsbildet. Jo større filteret, desto mer komplekse funksjoner kan det lære. Likevel vil beregningskostnadene øke, og det kan føre til overtilpasning fordi den husker treningsdataen fremfor å gjøre nye prediksjoner. Steglengde refererer til avstanden mellom hvert trinn som filteret tar når det beveger seg over inngangsbildet. Som Figur 2-28 viser, vil filteret bevege seg én piksel om gangen hvis steglengden er satt til 1. Steglengden bestemmer hvor mye overlapp det vil være mellom ulike deler av inngangsbildet, og dermed hvor mye informasjon som

trekkes ut. Utfylling-parameteren legge til ekstra nuller rundt inngangsbildet, slik at filteret kan bevege seg over kantene av inngangsbildet. Dette kan bidra til å bevare informasjonen langs kantene av inngangsbildet, og kan dermed føre til høyere nøyaktighet i modellen. Imidlertid vil utfylling øke størrelsen på egenskapskartet og beregningskostnadene.



Figur 2-28 – Inndata er på 5x5 piksler, filteret er av størrelse 4x4 piksler, steglengden er 1 piksel og utfyllingen «rundt» inndataen er på 2 piksler (Howard og Gugger, 2020, s. 411).

Hvert filter i et konvolusjonslag bidrar til å produsere et kart over egenskaper, og hvert kart er ansvarlig for å gjenkjenne et spesifikt visuelt trekk i inndataene. Ved å ha flere slike kart i et konvolusjonslag kan modellen lære å gjenkjenne flere og mer komplekse trekk i inndataene, som vist i Figur 2-29. Det vil derimot være en grense for hvor mange lag som er hensiktsmessig å legge til. På et tidspunkt vil ikke flere lag bidra til å hente ut mer informasjon fra inndataene, men heller føre til overtilpasning av modellen. Dette betyr at modellen blir så godt tilpasset til treningsdataene at den mister evnen til å generalisere til ny og usett data. Å finne riktig balanse mellom antall konvolusjonslag og antall filtre i hvert lag er en viktig del av å utvikle en effektiv CNN-modell.

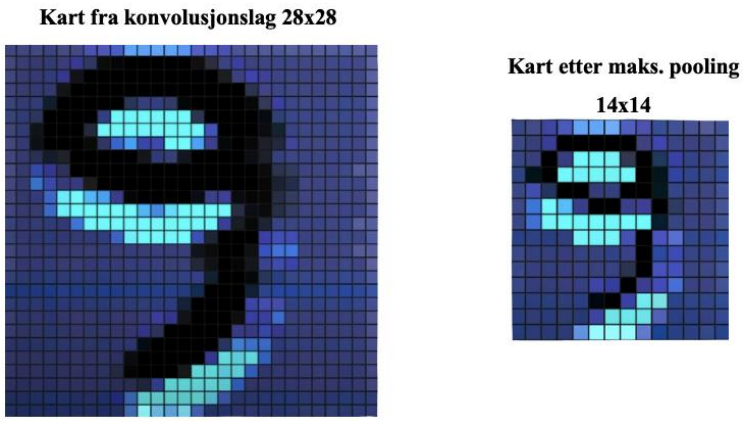


Figur 2-29 – Det er ikke uvanlig å ha flere filtre som trekker ut forskjellig informasjon fra inndataen (CNN Visualized, 2020).

Etter at egenskapskartene er generert gjennom konvolusjonslagene, blir det vanligvis ført inn i et eller flere lag med aktiveringsfunksjoner. Som beskrevet i punkt 2.4.5.2, bidrar funksjonene til å innføre ikke-linearitet i modellen og gjør modellen mer fleksibel og i stand til å modellere komplekse sammenhenger i datasettet.

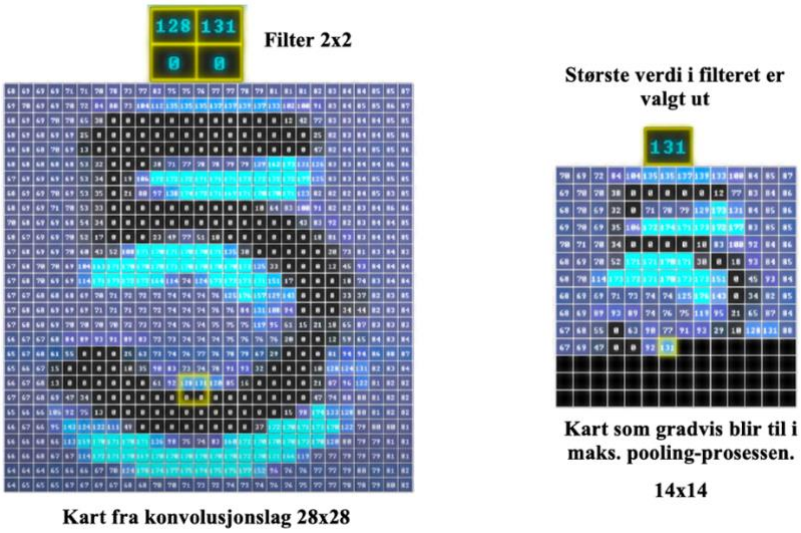
2.4.8.2 Reduksjonslag

Reduksjonslaget *(pooling layer*) er det neste steget i et CNN etter konvolusjonslaget. Hensikten med reduksjonslaget er å redusere størrelsen til egenskapskartene produsert i konvolusjonslaget, slik Figur 2-30 viser. På denne måten vil viktig informasjon bevares og uviktig informasjon forkastes. Dette medfører en lavere risiko for overtilpasning, ettersom den reduserte dimensjonaliteten vil forhindre at modellen fokuserer for mye på støy og unødvendig informasjon i treningsdataen. Slik vil modellen bli mer generell og bedre i stand til å generalisere til nye data. Reduksjonslaget vil også være med på å akselerere beregningene i senere lag, på bakgrunn i at den får færre datapunkter å behandle. Dermed kan de senere lagene kunne beregne raskere og mer effektivt. Dette kan føre til en betydelig hastighetsforbedring og mindre beregningskraft som kreves for å trene og evaluere modellen.



Figur 2-30 – Egenskapskart før og etter maks. pooling (CNN Visualized, 2020).

Nedskaleringen av størrelsen på egenskapskartene foregår ved at et filter glir over inndataen, slik som i konvolusjonslaget, og trekker ut én spesifikk verdi som beholdes i det nye laget. Figur 2-31 viser en 2x2 filteret som flyttes over egenskapskartet for å velge ut den høyeste verdien som skal beholdes i den nedskalerte versjonen.



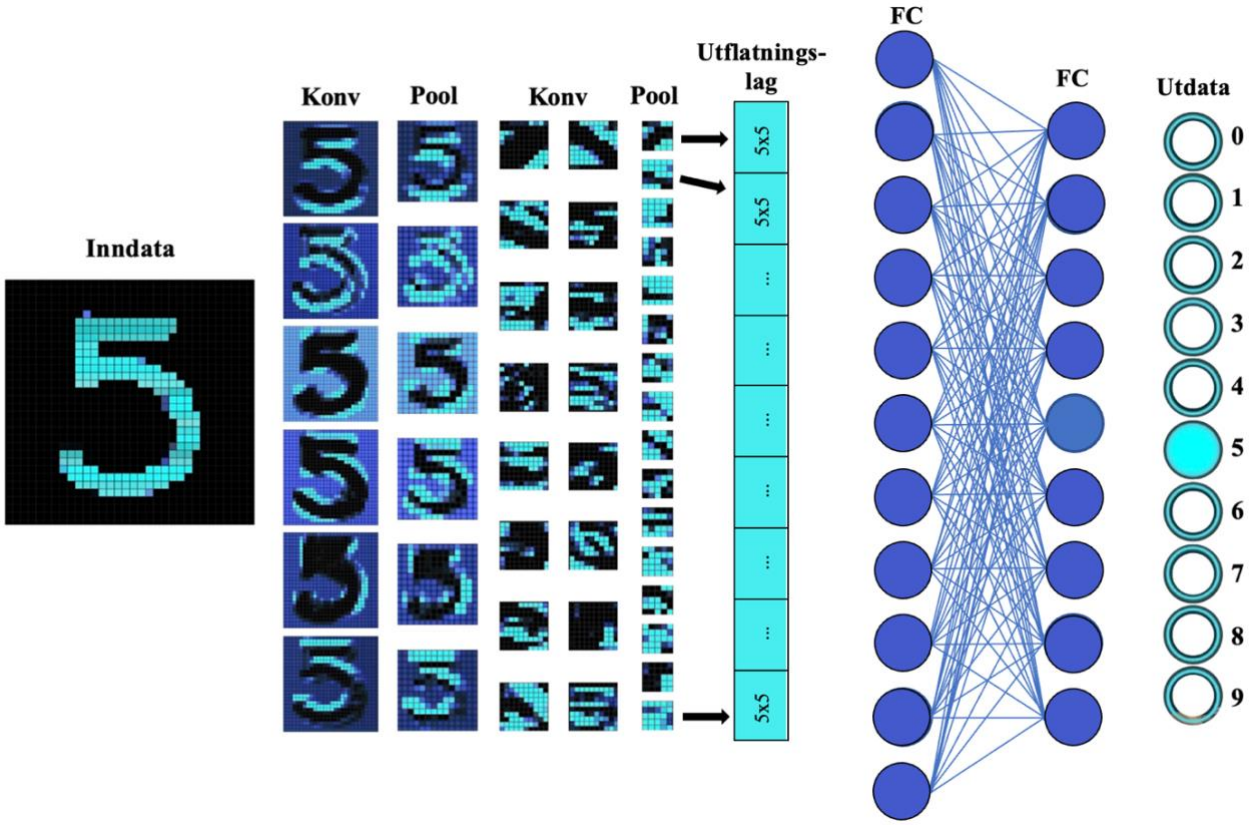
Figur 2-31 – Maks. pooling utføres på et egenskapskart fra konvolusjonslaget (CNN Visualized, 2020).

Det finnes flere forskjellige teknikker for reduksjonslagene, og de mest vanlige er maksimum pooling, gjennomsnittlig pooling og minimum pooling. I maksimum pooling velges den maksimale pikselverdien av alle pikslene i matrisen, som Figur 2-31 er et eksempel på. I gjennomsnittlig pooling velges gjennomsnittsverdien av alle pikslene i matrisen, og i minimum pooling velges den minste verdien.

Det neste steget i et CNN går ut på å gjenta konvolusjon- og reduksjonslagene, slik at modellen gradvis lærer å gjenkjenne høyere nivåer av mønstre og egenskaper i inndataene.

2.4.8.3 Fullt tilkoblet lag

Etter flere konvolusjons- og reduksjonslag, flates resultatene til et fullt tilkoblet lag (*fully connected layer*), som illustrert på enden av Figur 2-26 og Figur 2-32. De tilkoblede lagene fungerer som et tradisjonelt nevralt nettverk der hver node er koblet til alle nodene i det forrige og det neste laget. I det siste av de tilkoblede lagene, blir inndataene klassifisert i forskjellige kategorier, slik Figur 2-32 viser.



Figur 2-32 – Klassifisering av et 5-tall ved hjelp av to konvolusjonslag, to reduksjonslag og to fullt tilkoblede lag noder (CNN Visualized, 2020).

3 PROSJEKTBESKRIVELSE

Dette kapittelet gir en mer inngående beskrivelse av problemet, inkludert problemstillingen og hovedmålet for prosjektet med tilhørende delmål og forskningsspørsmål. Videre gir kapittelet en innføring i den praktiske bakgrunnen for prosjektet. Dette innbefatter pågående arbeid, initielle krav og en initiell løsnings-idé. Kapittelet forklarer også de begrensninger og avgrensninger som gjelder for prosjektet, og identifiserer nødvendige og nyttige ressurser. Til slutt gis det en oversikt over relevant litteratur om problemstillingen.

3.1 Problembeskrivelse og mål

Som beskrevet i punkt 1.4 av den overordnede problembeskrivelsen, skal det bli utviklet en ny analysestrategi der maskinlæring brukes for å klassifisere simulerte kollisjonsbilder fra ATLAS.

En av de mest effektive teknikkene for å klassifisere bilder er dyp læring (Shawahna, Sait, og El-Maleh, 2018). Blant de dype nevrale nettverkene, har konvolusjonelle nevrale nettverk (CNN) vist seg å være spesielt godt egnet for slike oppgaver. Dette skyldes at CNN-modeller har evnen til å oppdage mer komplekse mønstre og sammenhenger i data, som går utover det som er mulig med tradisjonell maskinlæring. For å ytterligere forbedre en CNN-modell, kan mengden treningsdata økes gjennom data augmentering (Shorten og Khoshgoftaar, 2019). Basert på dette ble følgende problemstilling valgt:

> *Hvordan kan data augmentering implementert i en CNN-modell bidra til å øke sensitiviteten i klassifiseringen av detektorbilder fra ATLAS (HVL)?*

Prosjektets overordnede mål er å forbedre læringen og treningen av CNN-modellen ved hjelp av data augmentering, med sikte på å oppnå bedre sensitivitet i analysen av simulerte partikkelkollisjoner. For å nå det overordnede målet er det satt tre delmål for prosjektet:

1. Implementere den nye analysestrategien ved å konvertere simulerte kollisjonsdata fra ATLAS-detektoren til bilder som kan brukes i en CNN-modell for videre klassifisering.
2. Øke mengden treningsdata, og inkludere denne i treningsprosessen for å forbedre modellens sensitivitet i klassifiseringen. Dette vil gjøres i samsvar med oppdragsgivers mål listet i punkt 3.2.2.

3. Evaluere ytelsen til den modifiserte CNN-modellen ved å sammenligne resultatene fra denne med en enkel modell.

For å forbedre klassifiseringen av mikroskopiske sorte hull og sfaleroner, er det et overordnet forskningsspørsmål som er sentralt for prosjektet: Hvordan kan treningssettet og CNN-modellen tilpasses for å nå prosjektets overordnede mål? Dette forskningsspørsmålet kan deles inn i flere mindre forskningsspørsmål som er relevante:

- Hvordan kan egenskaper ved detektorbildene brukes til å tilpasse arkitekturen i CNN-modellen?
- Hvilke teknikker for data augmentering kan bli implementert i CNN-modellen for å forbedre sensitiviteten av klassifiseringen?

Flere av forskningsspørsmålene blir nærmere utredet i punkt 3.2.2.

3.2 Praktisk bakgrunn

3.2.1 Pågående arbeid

Dette bachelorprosjektet er tilknyttet det pågående arbeidet til doktorgradsstudenten, Aurora Singstad Grefsrud, som er en del av HVL sin ATLAS gruppe. Dette prosjektet skiller seg imidlertid fra hennes prosjekt, ettersom det har selvstendige mål og vil utforske andre teknikker og metoder innen maskinlæring.

Det pågår også et annet bachelorprosjekt parallelt med dette, med samme oppdragsgiver og datasett. På tross av at begge prosjektene benytter seg av konvolusjonelle nevrale nettverk og simulerte kollisjonsdata fra ATLAS (HVL), har prosjektene ulike fokusområder og mål som å teste diverse arkitekturer på nettverket og tapsfunksjoner. Det kan derimot være interessant å sammenligne eventuelle resultater.

3.2.2 Initielle krav

Oppdragsgiveren har følgende forventninger og krav:

- «Tilpasse første pooling-lag av et CNN til å ta hensyn til sylindersymmetrien, og høyre-venstresymmetrien til detektorbildene.»
- «Undersøke om, og hvordan, symmetriegenskapene til detektorbildene kan utnyttes til «data augmentation» uten å innføre bias i klassifikasjonen.» (Buanes og Mæland, 2023)

Basert på de ovennevnte målene fra oppdragsgiveren og forskningsspørsmålene gitt i punkt 3.1, er det visse mål som prosjektet må oppfylle.

3.2.2.1 Symmetriegenskaper

Som beskrevet i punkt 2.3, oppstår detektorbildene ved å "brette ut" den sylindriske ATLAS-detektoren. Dette medfører at bildene har bestemte symmetriegenskaper som vil bli utforsket nærmere, og som kan være av betydning for behandlingen og analysen av bildene i maskinlæringsarbeidet.

På grunn av ATLAS-detektorens sylindriske form oppstår det en sylindersymmetri i de resulterende bildene. Dette innebærer at bildene har en symmetrisk struktur langs den vertikale aksen når de blir brettet ut. Når sylinderen brettes ut og flates til et bilde, vil toppen og bunnen av sylinderen utgjøre hver sin side av kvadratet, noe som resulterer i en symmetrisk fordeling av partikler og deres interaksjoner langs den vertikale aksen (y-aksen) i bildet.

I proton-proton-kollisjonene er det sannsynlig at det oppstår speilvendte versjoner av detektorbildene. Dette skyldes at protonene kolliderer med samme energi, men i motsatte retninger. En høyre-venstresymmetri langs den horisontale kollisjonsaksen (x-aksen) kan derfor være en naturlig følge av kollisjonsprosessen, hvor partiklene spres i henholdsvis høyre og venstre retning. Kollisjonsaksen er en linje som passerer gjennom kollisjonens midtpunkt og er parallell med protonenes bevegelsesretning. Speilbildet av partikkelbanene og energideponeringen oppstår på begge sider av kollisjonsaksen, noe som resulterer i detektorbildenes høyre-venstresymmetri. Når en partikkel oppstår på den ene siden av kollisjonsaksen, er det sannsynlig at en annen partikkel med lignende egenskaper og energi vil oppstå på motsatt side.

3.2.2.2 Skjevhet og data augmentering basert på symmetri

Skjevhet (*bias*) oppstår når en modell systematisk favoriserer eller diskriminerer visse grupper eller kategorier på en urimelig eller urettferdig måte. Dette kan skyldes manglende mangfold i treningsdataene, feilaktige antagelser i modellens design eller feil behandling av symmetriegenskaper i detektorbildene.

Når man bruker symmetriegenskapene til å utføre data augmentering, er det viktig å være oppmerksom på at disse teknikkene ikke introduserer uønsket skjevhet i modellen. Derfor er det nødvendig å forsikre seg om at teknikker som eksempelvis rotasjon eller speiling av bildene langs symmetriaksene, ikke påvirker modellens evne til å skille mellom mikroskopiske sorte hull og sfaleroner.

3.2.2.3 Utdypning av forskningsspørsmål

Det første målet fra oppdragsgiveren krever at maskinlæringsmodellen tar hensyn til symmetriegenskapene til detektorbildene. Dette er tilknyttet forskningsspørsmålet «Hvordan kan egenskaper ved detektorbildene brukes til å tilpasse arkitekturen i CNN-modellen?»
I denne sammenhengen er symmetriegenskapene de kjente egenskapene ved detektorbildene. For å nå dette målet er det nødvendig å foreta justeringer i det første reduksjonslaget, slik at modellen kan tolke bildene med hensyn på symmetriegenskapene. Detaljer om overgangen fra ATLAS-sylinderstruktur til bilder ble forklart i punkt 2.3.

Det andre målet fra oppdragsgiveren er å utnytte symmetriegenskapene til detektorbildene for å øke mengden treningsdata ved bruk av data augmentering. Dette er knyttet til forskningsspørsmålet «Hvilke teknikker for data augmentering kan bli implementert i CNN-modellen for å forbedre sensitiviteten av klassifiseringen?» For å dekke målet og besvare forskningsspørsmålet er det nødvendig å utforske en rekke teknikker innen data augmentering for å finne ut hvilke(n) teknikk(er) som kan forbedre treningen til CNN-modellen.

Følgende kravliste kan oppsummeres:

1. Klassifisering av bilder: CNN-modellen skal kunne klassifisere bilder av mikroskopiske sorte hull og sfaleroner, uavhengig av symmetriegenskapene.
2. Treningsmengde: Det skal undersøkes hvordan teknikker innen data augmentering påvirker CNN-modellens ytelse, uten at det blir innført skjevhet i klassifiseringen.

3. Evaluering av modellen: CNN-modellen skal evalueres ved å måle modellens sensitivitet.
4. Sammenligning av resultater: Resultatene fra ulike teknikker innen data augmentering skal sammenlignes for å evaluere hvilken teknikk som gir økt sensitivitet.

Gjennom å kombinere forståelsen av symmetriene i detektorbildene og deres innvirkning på CNN-modellen, vil prosjektet kunne oppnå målene satt av oppdragsgiveren og besvare de forskningsspørsmålene som er stilt. Resultatene fra dette eksperimentet vil forhåpentligvis bidra til å forbedre klassifisering av kalorimeterbilder fra ATLAS og gi en bedre forståelse av hvordan data augmentering kan brukes effektivt i denne sammenhengen.

3.2.3 Initiell løsnings-idé

Prosjektets initielle løsnings-idé er å implementere to CNN-modeller: en enkel modell uten data augmentering og en modifisert modell med data augmentering. Den enkle modellen vil klassifisere mikroskopiske sorte hull og sfaleroner uten noen endringer i verken treningssettet eller modellens arkitektur. Siden datasettet ikke har blitt brukt før, vil den enkle modellen fungere som et utgangspunkt for å evaluere resultatene til den modifiserte modellen. Den modifiserte modellen vil klassifisere detektorbildene etter at det er utført data augmentering på treningssettet og det første reduksjonslaget er tilpasset symmetriegenskapene til detektor-bildene. Følgelig skal den samme evalueringsmetoden brukes for å evaluere begge maskinlæringsmodellene, slik at resultatene mellom de to modellene kan sammenlignes.

3.3 Begrensninger og avgrensninger

I dette bachelorprosjektet har det vært flere begrensninger utenfor prosjektgruppens kontroll som kan ha påvirket prosjektløpet. Arbeidsplassen og maskinvaren ble delt med en annen bachelorgruppe, og dette medførte at begge gruppene måtte tilpasse seg hverandre for å arbeide med prosjektene sine. Dette kan ha redusert effektiviteten i arbeidet, da gruppene måtte følge et tidskjema og ikke alltid kunne arbeide når det var ønskelig. Det var også noe feil med det første datasettet av simulerte kollisjonsdata som ble utlevert, hvilket gjorde at arbeidet ble satt til dels på pause frem til feilen ble rettet opp i.

Mangel på spesifikk domenekunnskap innen fysikk og maskinlæring spilte også en rolle i fremgangen av prosjektløpet. Tilgjengelig litteratur som kunne besvare de spesifikke problemene som ble forsøkt løst, var også begrenset. Dette førte til at mye av prosjektet har vært basert på eksperimentering, testing og justering av feil.

For å gjøre prosjektet mer håndterlig, ble det gjort noen avgrensninger i omfanget. Tidlig i prosjektfasen ble det utprøvd ulike arkitekturer for CNN-modellen, men dette ble holdt til et basistilfelle som beskrives nærmere i alternative løsninger i punkt 4.1. Basert på funnene, ble det bestemt å begrense antall typer CNN-modeller til kun én, som hadde fem konvolusjonslag fremfor tre som er standard for CNN-modeller, for å oppnå økt kompleksitet. Valget med å utforske kun en enkel CNN-modell ble tatt på bakgrunn av tidsbegrensninger for trening og testing av modeller.

Prosjektets fokus ble lagt på bruken av data augmentering og tilpasning av det første reduksjonslaget, mens bredden av hyperparametre inkludert tapsfunksjoner, og teknikker for regularisering og optimalisering ble nedprioritert. Dette ble gjort for å begrense prosjektets omfang og sikre en dypere forståelse av de valgte områdene. Følgelig ble det ikke investert tid eller ressurser i forbedring av andre deler av nettverket.

3.4 Ressurser

For å fullføre prosjektet, var det nødvendig å ha tilgang til ulike ressurser, både teknisk og faglig. Denne seksjonen vil beskrive de tekniske ressursene som var nødvendige for å utvikle modellen, samt hvordan effektiv kommunikasjon og veiledning ble sikret gjennom prosjektet.

3.4.1 Tekniske ressurser

For å utvikle en modell innen dyp læring, kreves betydelig datamaskinkraft. Derfor var det nødvendig å ha tilgang til maskinvare med høy ytelse, slik som en datamaskin med et kraftig grafikkort (GPU). Dette gjorde det mulig å utføre flere parallelle beregninger samtidig, som igjen var avgjørende for å prosessere store mengder data på en effektiv måte i utviklingsprosessen (Asano, Maruyama og Yamaguchi, 2009).

For å utvikle CNN-modellen, ble det valgt et utviklingsmiljø og et programmeringsspråk som er optimalt for maskinlæring. Python er kjent for å være det mest populære språket fordi det forenkler maskinlæringsdelen av prosjektet (Nagpal og Gabrani, 2019). Som en del av dette valget var det nødvendig å installere ulike pakker og biblioteker for å utføre ulike oppgaver. Fullstendig liste over brukte pakker og biblioteker kan bli funnet i tekstfilen kalt *req.txt* på GitHub oppbevaringssted[11].

GitHub ble valgt som versjonskontrollverktøy for å sikre en effektiv og sikker måte å lagre og dele koden. Dette gjorde det enklere å administrere prosjektet og opprettholde en god oversikt over endringer og versjoner av koden. Samtidig gjorde det også at koden kunne deles med oppdragsgiveren og andre som måtte ønske å følge med på prosjektets utvikling, uavhengig av deres foretrukne plattform. Med GitHub som grensesnitt, kunne alle prosjektmedlemmer bidra til prosjektet på en sømløs måte, samtidig som dataintegriteten ble opprettholdt.

3.4.2 Datasett

En avgjørende ressurs for bachelorprosjektet er datasettet som ble utlevert av oppdragsgiveren. Datasettet fungerer som grunnlaget for å trene og evaluere maskinlæringsmodeller, og gir en representasjon av virkeligheten som modellen skal generalisere fra. Innholdet i datasettet er beskrevet i punkt 2.3.

3.4.3 Veiledning og kommunikasjon i prosjektet

Kommunikasjonen mellom prosjektmedlemmene skjedde gjennom ulike sosiale medier, og møtene med oppdragsgiver fant sted fysisk ved HVL. E-post ble også benyttet ved behov. Valget av disse verktøyene for kommunikasjon var avgjørende for å sikre et effektivt prosjektløp og en vellykket utviklingsprosess.

Prosjektveilederen bidro som faglig hjelp innenfor prosjektskrivingen, og kontaktpersonene fra HVL sin ATLAS gruppe fungerte som ressurspersoner for det teoretiske og praktiske innenfor fysikk og maskinlæring. Doktorgradsstudenten som genererte datasettet, hjalp også med den tidlige forståelsen av datasettet og startkoden.

[11] Lenke til GitHub oppbevaringssted: https://github.com/591308/Optimazing-klassification-of-ATLAS-detector-data--CERN-/blob/master/req.txt

3.5 Litteratur om problemstillingen

I forarbeidet ble det ikke identifisert noen prosjekter som var identiske med dette. Dette skyldtes antageligvis den unike bruken av et datasett som representerer mikroskopiske sorte hull og sfaleroner. Det fantes imidlertid flere prosjekter med metoder og resultater som kunne tjene som inspirasjon for dette prosjektet. I den forbindelse, ble to studier analysert for å undersøke deres relevans og resultater.

Dieleman, Fauw, and Kavukcuoglu (2016) presenterte en studie der de undersøkte hvordan de kunne *Utnytte syklisk symmetri i konvolusjonelle nevrale nettverk*. Bakgrunnen for studien var at filtrene i lagene i CNN-modeller lærte å identifisere bestemte mønstre i inndataen, som ofte kunne være vilkårlige orienteringer av hverandre. Følgelig lærte CNN-modellen å gjenkjenne flere kopier av det samme mønsteret, som bare hadde forskjellige orienteringer. Dette gjaldt spesielt rotasjonssymmetri. Det ville derfor vært nyttig å inkorporere en form for rotasjonssymmetri i nettverksstrukturen, slik at modellen ikke trengte å lære ulike rotasjoner av det samme mønsteret. Målet med dette var å redusere redundans og frigjøre modellkapasitet, eller redusere risikoen for overtilpasning. Studien presenterte derfor et rammeverk bestående av fire nye lag/operasjoner som kunne implementeres i den eksisterende nettverksstrukturen for å gjøre den ekvivariant[12] til sykliske rotasjoner[13]. Et av lagene var et syklisk reduksjonslag (*cyclic pooling layer*). På samme måte ville dette prosjektet tilpasse deler av nettverksstrukturen – her det første reduksjonslaget – til egenskaper i inndataen. Forskjellen var at dette prosjektet skal tilpasse reduksjonslaget til symmetriegenskapene og ikke bruke syklisk rotasjon. Begge prosjektene søker med dette å forbedre ytelsen til en CNN-modell ved å tilpasse nettverksstrukturen, samt å redusere risikoen for overtilpasning.

Ramprasad, Raina og Mondal (2020) presenterte en studie der de undersøkte *Effekten av data augmentering på ytelsen til en finjustert CNN-modell*. I studien ble det lagt frem tre teknikker som ble brukt for å øke mengden treningsdata i datasettet: horisontal flipping, rotasjon og skalering. På samme måte som i dette prosjektet, var formålet med bruken av teknikker innen

[12] I denne sammenhengen refererer ekvivarians til en egenskap til et nevralt nettverk som bevarer forholdet mellom inndata og utdata når inndata har rotasjonssymmetri. Med andre ord betyr det at nettverket vil gi forutsigbare utdata når inndataen blir transformert.

[13] Syklisk rotasjon refererer til rotasjon av et objekt rundt en sirkulær bane, slik at objektet ender opp i sin opprinnelige posisjon. For eksempel vil en figur som roteres 90 grader, 180 grader eller 270 grader rundt en sirkulær bane, ende opp i sin opprinnelige posisjon.

data augmentering å forbedre nøyaktigheten for klassifiseringen og takle problemer med overtilpasning for å forbedre modellgeneralisering. Ved å sammenligne ytelsen til forhåndstrente modeller med og uten data augmentering, viste forfatterne at bruken av data augmentering betydelig forbedret nøyaktigheten til klassifiseringen. Studien understreket viktigheten av å bruke passende teknikker for data augmentering, spesielt i tilfeller der det tilgjengelige datasettet er lite eller begrenset. Begge prosjektene søker dermed å forbedre en CNN-modell gjennom å øke mengden treningsdata, i tillegg til å unngå overtilpasning av modellen til treningsdataen.

Studiene viser at det er mulig å forbedre klassifiseringsmodellen for bildegjenkjenning ved å bruke teknikker som inkorporerer egenskaper ved inndataene og ved å bruke data augmentering for å øke mengden treningsdata. Begge studiene har relevans for dette prosjektet og kan gi nyttige innspill for å forbedre klassifiseringen av mikroskopiske sorte hull og sfaleroner basert på deres mønstre.

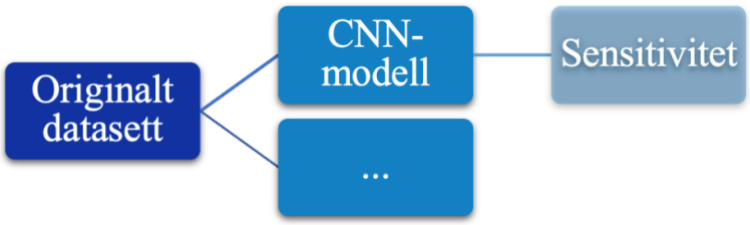
4 DESIGN AV PROSJEKTET

Dette kapittelet gir en oversikt over ulike alternativer for å løse prosjektet, deres fordeler og ulemper, og velger til slutt den mest passende løsningen. Videre diskuteres verktøyvalg og valg av utviklingsmetodikk, i tillegg til at det legges frem en plan for prosjektet og vurdering av risiko. Til slutt presenteres en evalueringsplan som vil vurdere prosjektresultatene i samsvar med forskningsspørsmålene og de initielle kravene beskrevet i henholdsvis punkt 1.4 og 3.2.2.

4.1 Forslag til løsning

4.1.1 Referansegrunnlag

For å bedømme om en alternativ løsning er god eller dårlig for klassifiseringen av mikroskopiske sorte hull og sfaleroner, er det nødvendig å etablere et referansegrunnlag. Dette er viktig for å kunne evaluere om endringene som gjøres fører til ønsket forbedring. Referansegrunnlaget vil bli utviklet ved å bruke det opprinnelige datasettet slik Figur 4-1 viser, og det vil heller ikke bli foretatt noen tilpasninger til symmetriegenskapene (beskrevet i punkt 3.2.2.1). Ved å strukturere referansegrunnlaget på denne måten oppnår man en mer nøytral sammenligningsbasis, som gir et objektivt grunnlag for evalueringen av den modifiserte maskinlæringsmodellen med data augmentering. I samsvar med målene og forsknings-spørsmålene beskrevet i punkt 3.1, vil modellenes ytelse bli vurdert ut fra sensitiviteten til klassifiseringen.



Figur 4-1 – Grunnlaget for den videre klassifiseringen skal etableres ved å trene en CNN-modell med original treningsdata, som evalueres med hensyn på sensitivitet. Med andre ord utgjør referansegrunnlaget basistilfellet for sammenligningen.

4.1.2 Alternativ løsning 1

I alternativ løsning 1 vil kun én type CNN-modell bli brukt for å klassifisere mikroskopiske sorte hull og sfaleroner. I dette prosjektet er det to mulige justeringer som kan gjøres med CNN-modellen, i henhold til det overordnede målet for prosjektet. Den første justeringen er å bruke data augmentering på treningssettet til modellen, og den andre justeringen er å tilpasse det første reduksjonslaget til symmetriegenskapene i detektorbildene. Dermed finnes det fire ulike variasjoner av CNN-modellen, som representert i Tabell 4-1. *Modifisert modell* refererer til en CNN-modell der det første reduksjonslaget er tilpasset til symmetriegenskapene i detektorbildene, mens *Enkel modell* er uten modifiseringer i arkitekturen.

Tabell 4-1 – De ulike variasjonene av CNN-modellen for alternativ løsning 1.

1. **Basistilfelle**: Enkel modell uten data augmentering
2. Enkel modell med data augmentering
3. Modifisert modell uten data augmentering
4. Modifisert modell med data augmentering

For alle tilfellene med data augmentering, vil det bli undersøkt om bruken av data augmentering vil innføre bias i klassifiseringen. Selve evalueringen av modellen baseres på sensitiviteten til analysen. Ved å sammenligne evalueringen av sensitiviteten til modellen før og etter justeringene, vil det være mulig å se om sensitiviteten har økt.

Fordeler med denne løsningen er at den er tidsbesparende, i tillegg til at den vil være tilstrekkelig for å teste alle målene som er gitt i problembeskrivelsen. Ulempen med denne tilnærmingen er at det ikke er mulig å garantere at den beste mulige løsningen blir funnet, siden det kun er én type CNN-modell som undersøkes.

4.1.3 Alternativ løsning 2

I alternativ løsning 2 vil to av de mest anerkjente CNN-modellene for klassifiseringsoppgaver bli valgt ut. VGGNet og ResNet er valgt på grunn av deres imponerende ytelse i bildeklassifisering og deres evne til å ekstrahere ulike nivåer av abstrakte egenskaper fra bilder (Too mfl., 2019). Nettverkene er kjente for sin dybde og evne til å oppdage og representere ulike detaljnivåer og kompleksiteter i bildedata, som en enkel CNN-arkitektur ikke kan oppnå.

VGGNet er kjent for sin enkelhet og uniforme arkitektur, noe som gjør den lett å forstå og implementere samtidig som den er svært effektiv i bildegjenkjenning (Too mfl., 2019). På den andre siden introduserte ResNet en banebrytende arkitektur med «hoppforbindelser», som muliggjør trening av mye dypere nettverk ved å redusere problemet med «forsvinnende gradienter», nevnt i punkt 2.4.5.2. Dette har ofte ført til bedre resultater innen bildegjenkjenning.

På samme måte som i alternativ 1, vil begge typene CNN-modeller bli evaluert etter de to mulige justeringene. For å representere arbeidet denne løsningen krever, er det nødvendig å bruke to tabeller for variasjonene av modellene. Tabell 4-2 viser de ulike variasjonene for VGGNet og Tabell 4-3 viser variasjonene for ResNet.

Tabell 4-2 – De ulike variasjonene av VGGNet for alternativ løsning 2.

1. **Basistilfelle:** Enkel VGGNet-modell uten data augmentering
2. Enkel VGGNet-modell med data augmentering
3. Modifisert VGGNet-modell uten data augmentering
4. Modifisert VGGNet-modell med data augmentering

Tabell 4-3 – De ulike variasjonene av ResNet for alternativ løsning 2.

1. **Basistilfelle:** Enkel ResNet-modell uten data augmentering
2. Enkel ResNet-modell med data augmentering
3. Modifisert ResNet-modell uten data augmentering
4. Modifisert ResNet-modell med data augmentering

Det vil også bli undersøkt om bruken av data augmentering vil innføre bias i disse CNN-modellene. Sensitiviteten til modellene vil også bli evaluert før og etter justeringene, slik at det er mulig å undersøke om sensitiviteten har økt.

En ulempe med denne tilnærmingen er at treningen av VGGNet og ResNet kan være tidkrevende og kreve betydelige beregningsressurser. Deres komplekse arkitektur og dybde øker tidsbehovet for treningen, og det kan være nødvendig med kraftig maskinvare for å håndtere beregningskostnadene. Dette kan være en begrensning for prosjektets tidsplan og ressurser.

4.1.4 Alternativ løsning 3

I denne løsningen vil det bli valgt ut et bredt utvalg CNN-modeller som har vist seg å prestere godt i klassifiseringsoppgaver. På samme måte som i de andre alternativene, blir alle modellene evaluert ved hjelp av de samme variasjonene (1 til 4) som ble brukt i alternativ løsning 1 og 2. Det blir også undersøkt om justeringene kan føre til skjevhet i klassifiseringen, og sensitiviteten før og etter justeringene vil bli sammenlignet for å vurdere om den blir forbedret. Fordelen med denne løsningen er at den gir et solid sammenligningsgrunnlag og øker sjansene for å finne den mest optimale modellen. Tilnærmingen vil også oppfylle alle mål og krav i prosjektet, men den kan også bevege seg utenfor prosjektets rammer. Ulempene med denne løsningen er at implementeringen av et stort antall CNN-modeller vil være svært tidkrevende. Hver modell må vurderes med tanke på enkel arkitektur, første pooling-tilpasning og forskjellige kombinasjoner av data augmentering. Dette kan gjøre det vanskelig å beholde oversikt over alle vurderingene, i tillegg til at det mest sannsynlig vil kreve betydelig databehandlingskraft og tid. Oppsett av modellparametere vil også legge til arbeidsmengden og gjøre den større enn det som er nødvendig for å besvare prosjektets mål og problemstilling.

4.2 Diskusjon av alternativene og valgt løsning

Det kan være utfordrende å ta en beslutning angående valg av løsning for prosjektet, da gjennomføringen av hver løsning avhenger av ytelsen til hver enkelt CNN-modell. Derfor ble basistilfellet for hvert alternativ testet for å få en indikasjon på utførelsen av de ulike løsningene. Dette gjør det mulig å ta en beslutning basert på en deduktiv tilnærming som tar hensyn til relevante faktorer som kostnad, tid og resultatene fra testingen av basistilfellet.

I prosessen med å teste basistilfellet for alternativ løsning 1 ble det oppnådd en liten forbedring i sensitiviteten, og treningen av nettverket tok heller ikke lang tid. Spesifikke resultater blir presentert i kapittel 6.

I prosessen med å teste basistilfellet for alternativ løsning 2, ble det gjort flere interessante observasjoner. VGGNet viste seg å gi bedre resultater enn ResNet og hadde også mindre tendens til overtilpasning til treningsdataene. Imidlertid tok treningen av VGGNet lengre tid sammenlignet med ResNet. I tillegg er VGGNet en kompleks modell som har en tendens til å overtilpasser til treningsdataene, noe som påvirker dens evne til å generalisere. Det mest interessante funnet i denne prosessen var at det var lite som skilte ytelsen mellom alternativ

løsning 1 og 2, men at det derimot er stor forskjell på hvor lang tid det tok å trene modellene. Modellene i alternativ løsning 1 viste seg å være langt mer tidseffektive sammenlignet med ResNet og VGGNet i alternativ løsning 2. Selv om alternativ løsning 2 teoretisk sett kunne ha gitt marginale forbedringer i resultatene, ble det ansett som en uforholdsmessig stor investering i tid. Derfor ble alternativ løsning 1 vurdert som det mest hensiktsmessige valget, da gevinsten ved å velge alternativ løsning 2 var begrenset.

Basert på resultatene fra gjennomgangen av basistilfellet for både løsning 1 og 2, ble alternativ løsning 3 også forkastet. Det ble antatt at å bruke enda flere modeller ville gi omtrent de samme resultatene, samtidig som det ville kreve betydelig mer tid. Dette gjaldt både implementeringen av modellene og treningen av et stort antall modeller. Selv om det ville kunne gi et bedre sammenligningsgrunnlag for å avgjøre den beste modellen for klassifiseringen, kunne det ikke garantere at den absolutt beste løsningen ville bli funnet. Det er også mulig at det ville være en liten forskjell mellom de beste modellene, og dermed kan resultatet fra alternativ løsning 1 være potensielt like bra som resultatet fra denne løsningen. Med andre ord vil det ikke være hensiktsmessig å teste et stort antall modeller, når resultatene har potensial til å være ganske like.

4.3 Valg av verktøy

I denne seksjonen blir valget av verktøy beskrevet. Dette inkluderer maskinvare, utviklingsmiljø og programmeringsspråk, biblioteker og rammer, samt arkitekturvalg. Riktig verktøyvalg spiller en avgjørende rolle for å oppnå vellykkede resultater.

4.3.1 Maskinvare

For å sikre en effektiv og vellykket gjennomføring av prosjektet, ble det nøye vurdert hvilke verktøy som skulle brukes. Et sentralt aspekt var valget av maskinvare, spesielt en kraftig GPU som kunne håndtere de store mengdene data og komplekse beregninger som er nødvendige i dyp læring (Buber og Diri, 2018). Etter grundig evaluering falt valget på *NVIDIA GTX 1080 Ti*, en GPU som er kjent for sin høye ytelse og kompatibilitet med rammeverk innen dyp læring, som eksempelvis PyTorch.

I tillegg til den kraftige GPU-en ble det også lagt vekt på andre viktige maskinvarekomponenter for å sikre en sømløs arbeidsflyt. Dette inkluderte 16 GB RAM (*Random Access Memory*) for

effektiv håndtering av store datasett og raskere beregninger, samt en høyhastighetslagringsløsning for å minimere ventetiden ved lesing og skriving av data. Den nøye utvalgte maskinvaren dannet en solid plattform for prosjektgruppen, hvilket var avgjørende for å oppnå høy ytelse og nøyaktighet i prosjektet.

4.3.2 Utviklingsmiljø og programmeringsspråk

Valget av utviklingsmiljø falt på Visual Studio Code (*VSCode*) på grunn av dets brukervennlighet, fleksibilitet og omfattende utvalg av utvidelser som bidrar til å forenkle og forbedre arbeidsflyten. Python tilbyr også kraftige verktøy for visualisering og matematiske funksjoner, gjennom biblioteker som eksempelvis matplotlib, NumPy og pandas.

Python er valgt som programmeringsspråk på grunn av dets popularitet, enkelhet og et bredt utvalg av biblioteker og rammeverk, egnet for maskinlæring og bildegjenkjenning (Nagpal og Gabrani, 2019). I tillegg tilbyr Python kraftige verktøy for visualisering og matematiske funksjoner gjennom diverse biblioteker som matplotlib, NumPy og pandas, noe som gjør det egnet for maskinlæringsoppgaver. I tillegg er det et bredt spekter av åpen kildekode-biblioteker tilgjengelig for Python, som for eksempel TensorFlow og PyTorch, som gir nødvendige funksjoner for å implementere og utvikle komplekse maskinlæringsmodeller. Dette gjør Python til et foretrukket valg for utviklere og forskere innen feltet maskinlæring.

4.3.3 Biblioteker og rammer

For å bygge modellene ble PyTorch valgt som rammeverk for prosjektoppbygning i utviklingsmiljøet. PyTorch er et populært og fleksibelt rammeverk som tilbyr dynamisk grafbygging (PyTorch Contributors, 2023). Dette gjør det mulig å lage komplekse og tilpassede modeller. I tillegg har PyTorch et stort brukersamfunn og en aktiv utviklingsprosess, noe som gjorde det enkelt å finne ressurser og få støtte. Blant de viktige bibliotekene som ble brukt, var:

- NumPy: For numeriske beregninger og manipulasjon av data (NumPy Developers, 2022).
- Torch: PyTorch-biblioteket for å bygge og trene nevrale nettverk
- Scikit-learn: For ulike maskinlæringsoppgaver, inkludert evaluering og validering av modeller (Pedregosa mfl., 2011).
- Matplotlib: For visualisering og plotting av data (Hunter mfl., 2012)

Full oversikt over biblioteker og pakker brukt er gitt på GitHub oppbevaringssted[14].

4.3.4 Arkitekturvalg

Flere populære arkitekturer innen dyp læring ble vurdert for prosjektet, slik som VGGNet og ResNet, for å bestemme hvilken som passet best for prosjekt. Både VGGNet og ResNet er en type CNN-modell. Etter å ha tatt hensyn til faktorer som ytelse, kompleksitet og tilpasningsevne, valgte gruppen å utvikle en egendefinert CNN-arkitektur for prosjektet ved å tilpasse eksisterende arkitekturer.

Valg av arkitektur inkluderte to varianter av et konvolusjonelt nevralt nettverk som er spesialtilpasset for prosjektet. Begge modellene inneholder fem konvolusjonelle lag etterfulgt av batch-normalisering *(batch-normalization)* og ReLU-aktiveringsfunksjonen. Mellom de konvolusjonelle lagene, er det også et gjennomsnittlig reduksjonslag for å redusere størrelsen på egenskapskartene.

1. Enkel CNN-modell (ConvModel):
   Den første modellen har en standard CNN-arkitektur uten spesielle tilpasninger. Den inneholder fem konvolusjonelle lag der det benyttes batch-normalisering og ReLU-aktiveringsfunksjon, etterfulgt av gjennomsnittlig reduksjonslag.

2. Modifisert CNN-modell, tilpasset til symmetriegenskapene (ConvModelFPLMod):
   Den andre modellen har en tilpasset CNN-arkitektur som inneholder en spesiell symmetri pooling-funksjon, kalt *circular_shift* og *symmetry_pooling*. Denne funksjonen kombinerer symmetrier langs x-aksen (venstre-høyre) og y-aksen (topp-bunn) med en sirkulær forskyvning i y-retning for å generere et mer robust funksjonssett. Denne tilpasningen kan potensielt forbedre modellens ytelse, ved at modellen kan utnytte informasjonen om symmetriene i datasettet.

Valget falt på disse arkitekturene fordi de er enkle og tilpasningsdyktig, samtidig som de utnytter kraften til CNN for å lære robuste funksjoner fra bildedataene. Spesifikke tilpasninger, som symmetri-pooling i den modifiserte modellen, åpner opp for muligheten til å

[14] Lenke til GitHub: https://github.com/591308/Optimazing-klassification-of-ATLAS-detector-data--CERN-

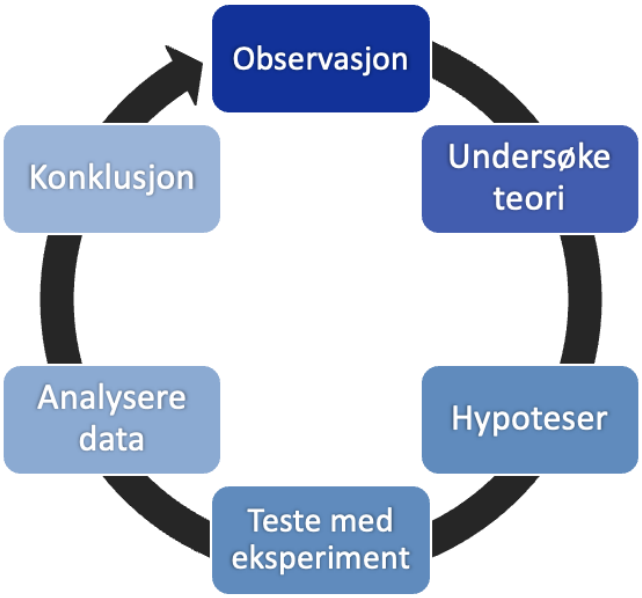
eksperimentere med ulike teknikker for å forbedre ytelsen til modellen og tilpasse modellene til prosjektets spesifikke formål. Den detaljerte løsningen til prosjektet og oppbyggingen av modellene blir nærmere beskrevet i Kapittel 5.

4.4 Prosjektmetodikk

4.4.1 Utviklingsmetodikk

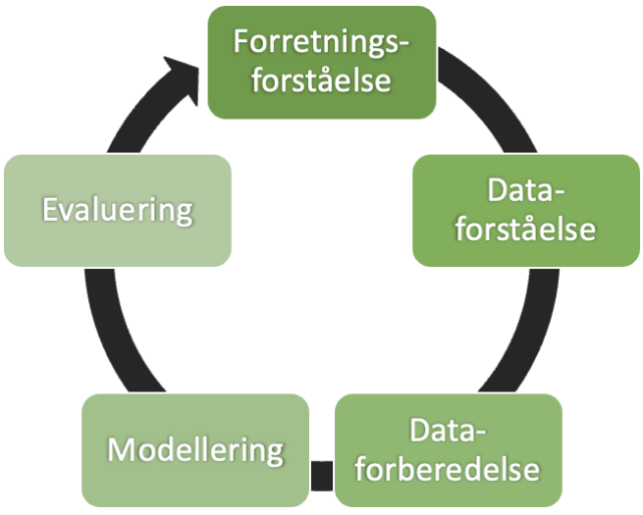
For å sikre effektiv og hensiktsmessig prosjektutvikling, var det viktig å benytte hensiktsmessige metodologier som fremmet systematikk, fleksibilitet og grundig evaluering. Denne seksjonen diskuterer bruken av vitenskapelig metode, CRISP-DM og smidig (*agile*) iterativ utvikling for å etablere en helhetlig og velbalansert tilnærming til modellutviklingen i prosjektet.

Vitenskapelig metode gir en systematisk og strukturert tilnærming for å forstå komplekse problemer og utvikle løsninger (Dodig-Crnkovic, 2002). Som nevnt i punkt 1.2 og illustrert i Figur 4-2, involverer den å observere og stille spørsmål og/eller identifisere et problem, formulere hypoteser, eksperimentere og teste hypotesene, analysere data fra eksperimentene, og deretter komme frem til en konklusjon basert på funnene. I dette prosjektet ble først oppgavebeskrivelsen undersøkt, etterfulgt av en periode med utforskning av relevant litteratur og tidligere forskning innenfor de aktuelle fagområdene. Dette bygger en solid forståelse av eksisterende kunnskap og teorier som er relevante for prosjektet. Litteraturgrunnlaget og oppgavebeskrivelsen gitt av oppdragsgiver, dannet grunnlaget for problemformuleringen, forskningsspørsmålene og metodene som skulle bli brukt i prosjektet. De ulike fremgangsmåtene for å løse problemstillingen og nå det overordnede målet representerer hypoteseformuleringene, og selve gjennomføringen av de alternative løsningene utgjør eksperimenteringsdelen. Resultatene fra utviklingen av en klassifiseringsmodell ble analysert, og forsøkene på å få den beste klassifiseringsmodellen gjennom tilpasning av det første reduksjonslaget og data augmentering ble gjentatt iterativt inntil en tilfredsstillende grad av resultatoppnåelse var nådd. Ved å introdusere vitenskapelig metode for å veilede forskningsprosessen, sikret det en grundigere forståelse av problemet og sørget for vitenskapelig grundighet i prosjektet.



Figur 4-2 – Vitenskapelig metode.

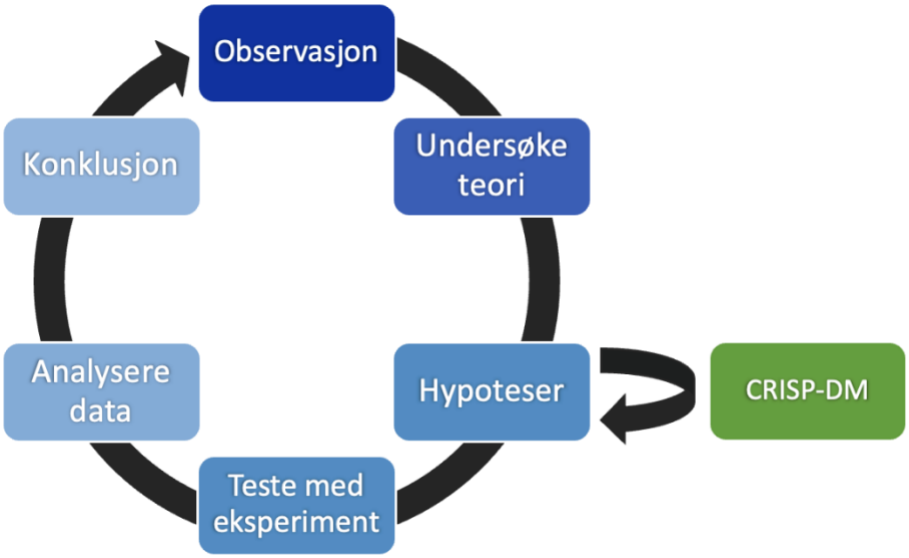
CRISP-DM (Cross-industry standard process for data mining) er en anerkjent metode og prosess innen datautvinning[15] eller «datagruvedrift» (*data mining*), og gir en systematisk tilnærming til modellutvikling (IBM, 2021). Denne metodologien omfatter flere viktige faser, inkludert forståelse av data, forberedelser av data, modellering og evaluering (Chapman mfl., 2000, s. 13). Fasene i CRISP-DM er illustrert i Figur 4-3. CRISP-DM vil dermed brukes som en utvidelse av eksperimenteringsdelen i vitenskapelige metode. Den strukturerte rammen til CRISP-DM passer sammen med de systematiske aspektene av vitenskapelig metode, for å sikre en grundig og omfattende prosess for modellutviklingen.



Figur 4-3 – CRISP-DM.

[15] Datautvinning er en prosess med å oppdage mønstre, sammenhenger og nyttig informasjon fra store datasett.

Metoden for smidig (*agile*) iterativ utvikling tilbyr en dynamisk og fleksibel tilnærming til modellutvikling, som gjør det mulig å tilpasse og forbedre modeller iterativt (Szalvay, 2004). I bachelorprosjektet er denne tilnærmingen kombinert med vitenskapelig metode og fasene fra CRISP-DM for å utdype eksperimenteringsdelen. Figur 4-4 viser hvordan den samlede utviklingsmetodikken vil se ut for prosjektet.



Figur 4-4 – Utviklingsmetodikk der vitenskapelig metode er kombinert med CRISP-DM.

Den smidige og iterative tilnærmingen gjør det mulig å raskt teste og validere hypoteser gjennom gjentatte iterasjoner av utviklingsprosessen (Szalvay, 2004). Dette er i tråd med vitenskapelig metode, som vektlegger systematisk og empirisk undersøkelse[16] for å sikre kontinuerlig kvalitetsforbedring. Ved å ta i bruk vitenskapelig metode og CRISP-DM for å oppnå smidig iterativ utvikling, gjør det mulig å evaluere og justere maskinlæringsmodellen kontinuerlig, basert på resultatene og funnene som oppnås underveis. Den systematiske og iterative fremgangsmåten legger dermed til rette for læring, justering og forbedring i prosjektet.

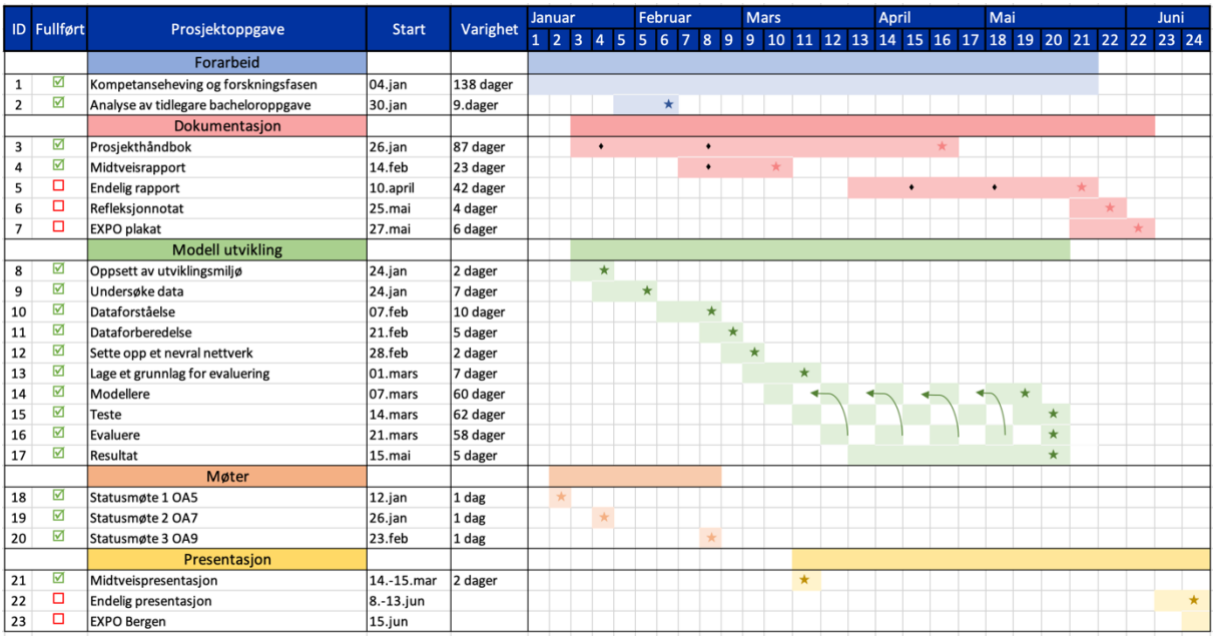
4.4.2 Prosjektplan

Prosjektplanen for bachelorprosjektet er presentert som et GANTT-diagram, som illustrert i Figur 4-5. GANTT-diagrammet viser prosjektets planlagte tidsplan og gir en oversikt over alle oppgavene som skal utføres i prosjektet, inkludert status og varighet for hver oppgave.

[16] Empirisk undersøkelse refererer til en tilnærming eller metode som er basert på direkte observasjon, erfaring eller datainnsamling fra virkelige hendelser eller fenomener.

Prosjektoppgavene er delt inn i fem kategorier: forarbeid, dokumentasjon, modellutvikling, møter og presentasjon.

Prosjektet har 21 milepæler, hver representert av en stjerne i Figur 4-5, som viser til viktige hendelser i prosjektløpet. Milepælene inkluderer blant annet oppsett av utviklingsmiljø, ferdigstilling av det nevrale nettverket, trening av modellen, evaluering av resultater og presentasjon av funn, gitt i kapittel 6. Figuren viser også iterasjoner av ulike dokumentasjonsoppgaver som hver representeres av en «diamant».



ID	Fullført	Prosjektoppgave	Start	Varighet
		Forarbeid		
1	☑	Kompetanseheving og forskningsfasen	04.jan	138 dager
2	☑	Analyse av tidlegare bacheloroppgave	30.jan	9.dager
		Dokumentasjon		
3	☑	Prosjekthåndbok	26.jan	87 dager
4	☑	Midtveisrapport	14.feb	23 dager
5	☐	Endelig rapport	10.april	42 dager
6	☐	Refleksjonnotat	25.mai	4 dager
7	☐	EXPO plakat	27.mai	6 dager
		Modell utvikling		
8	☑	Oppsett av utviklingsmiljø	24.jan	2 dager
9	☑	Undersøke data	24.jan	7 dager
10	☑	Dataforståelse	07.feb	10 dager
11	☑	Dataforberedelse	21.feb	5 dager
12	☑	Sette opp et nevral nettverk	28.feb	2 dager
13	☑	Lage et grunnlag for evaluering	01.mars	7 dager
14	☑	Modellere	07.mars	60 dager
15	☑	Teste	14.mars	62 dager
16	☑	Evaluere	21.mars	58 dager
17	☑	Resultat	15.mai	5 dager
		Møter		
18	☑	Statusmøte 1 OA5	12.jan	1 dag
19	☑	Statusmøte 2 OA7	26.jan	1 dag
20	☑	Statusmøte 3 OA9	23.feb	1 dag
		Presentasjon		
21	☑	Midtveispresentasjon	14.-15.mar	2 dager
22	☐	Endelig presentasjon	8.-13.jun	
23	☐	EXPO Bergen	15.jun	

Figur 4-5– GANTT-diagram som viser fremdriften i prosjektet. Stjernene representerer milepæler og «diamantene» representerer iterasjoner.

4.4.3 Risikovurdering

Enhver prosjektaktivitet er forbundet med en viss grad av risiko. Det ble derfor gjennomført en risikoanalyse for å identifisere mulige risikofaktorer, basert på en mal gitt i emnet «DAT191 Bacheloroppgave». For hver risiko i analysen ble det identifisert underliggende årsak(er) og angitt mulige proaktive og reaktive tiltak. Risikoene ble vurdert i alvorlighetsgrad basert på sannsynligheten (S) for at noe uønsket inntreffer og konsekvensen (K) av disse. Alvorlighetsgraden er dermed angitt av risikoproduktet (RP), altså produktet av S og K, og organisert i en risikomatrise gitt i Tabell 4-4. S og K strekker seg fra 1 til 5, hvilket gir en nedre grense på 2 og en øvre grense på 25 i risikomatrisen.

Tabell 4-4 – Risikomatrise som viser spennvidden av alvorlighetsgrad i risikoanalysen.

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					

Den fullstendige risikoanalysen er gitt i Tabell 10-1 i kapittel 10. De to risikoene med høyest alvorlighetsgrad er angitt i Tabell 4-5. Risikoer relatert til problemstillingen og manglende kompetanse kan forebygges ved å ha jevn og god dialog med interne og eksterne veiledere for å sørge for at det ikke oppstår noen misforståelser. Risikoer knyttet til konflikter mellom prosjektdeltakerne kan unngås ved å ha jevn arbeidsfordeling og åpenhet for hverandre dersom det oppstår uenigheter. Dersom det oppstår problemer i forbindelse med kontakt med veileder, kan man holde digitale møter og kommunikasjon over nett. Tekniske risikoer kan motvirkes ved å planlegge iterasjonene nøye og finne gode verktøy som kan minimere risikoen ved eventuelle problemer.

Tabell 4-5 – Utsnitt av risikoanalysen som viser de to risikoene med størst risikoprodukt (RP).

	Hendelse / risiko	Årsak	S	K	RP	Tiltak
1	Feiltolkning av oppgaven og tilhørende datasett	Grunnet prosjektets teoretiske og abstrakte natur kan man misforstå konsepter og/eller ha problemer med å forstå seg på oppgaven i sin helhet.	**4**	**5**	**20**	**Proaktive:** Jevn dialog med intern og eksterne veiledere. Se på relevant litteratur om problemstillingen og bli enige om en felles forståelse for oppgaven. **Reaktive:** Møte med veiledere for å finne ut av bakgrunnen for feiltolkningen, og sammen lage en plan for å rette opp i problemet.
2	Manglende kompetanse	Manglende kodeforståelse, kunnskaper om maskin-læring og/eller fysikk.	**4**	**5**	**20**	**Proaktive:** Lese seg opp på aktuelle teorier og begreper fra troverdige kilder. Be om hjelp fra eksterne veiledere og/eller HVL sin ATLAS gruppe ved eventuelle spørsmål / knutepunkter.

4.5 Evalueringsplan

Formålet med evalueringen er å vurdere om sensitiviteten til en CNN-modell kan forbedres gjennom spesifikke tiltak. Tiltakene innebærer å tilpasse det første reduksjonslaget til sylindersymmetri og høyre-venstresymmetri, samt å bruke data augmentering for å øke mengden treningsdata. For å foreta en slik evaluering, vil den modifiserte modellen sammenlignes med en enkel CNN-modell.

For å vurdere modellens ytelse og sensitivitet på en målrettet og effektiv måte, vil datasettet bli splittet i et treningssett og et testsett (Goot, 2021). Treningssettet vil bli brukt til å trene og tilpasse modellen, mens testsettet vil gi en uavhengig måling av modellens ytelse på data den ikke har sett før.

En forvirringsmatrise vil bli brukt for å oppnå en oversikt over antall sanne positive, falske positive, sanne negative og falske negative klassifiseringer, som beskrevet i punkt 2.4.6. Slik kan matrisen være med på å finne ut om modellen har vanskeligheter med å skille mellom de to klassene og/eller om den har høyere sensitivitet for en av klassene. Dette åpner for videre undersøkelser der man kan utforske hvorfor dette er tilfellet, og eventuelt tilpasse modellen for å forbedre sensitiviteten til begge klassene. Dersom en teknikk for data augmentering eksempelvis fører til at modellen blir mer sensitiv for en bestemt klasse, kan dette blant annet indikere at den har blitt introdusert for en form for skjevhet i datasettet.

Det er likevel en mulighet for at sensitivitet alene ikke gir en fullstendig indikasjon for modellens ytelse eller om det har blitt introdusert en form for skjevhet i datasettet. Derfor er det nyttig å bruke flere evalueringsteknikker for å få et mer komplett bilde. Dessuten vil dette gi en kvantitativ måte å evaluere modellenes prestasjon på, ettersom resultatene fra flere metrikker vil bli sammenlignet og diskutert for å identifisere styrker og svakheter ved modellen og evalueringen.

Nøyaktighet er en intuitiv og mye brukt metrikk i evalueringen av maskinlæringsmodeller. Ettersom datasettet som brukes er balansert – at det finnes like mange tilfeller av mikroskopiske sorte hull og sfaleroner – kan nøyaktigheten til klassifiseringen være en pålitelig indikator på modellens evne til å gjøre korrekte klassifiseringer. Nøyaktigheten vil også bli beregnet for mikroskopiske sorte hull og sfaleroner hver for seg, da sammenligningen av nøyaktigheten for

hvert fenomen kan gi verdifull innsikt i om modellen presterer å klassifisere det ene fenomenet bedre enn det andre.

Presisjon er også en nyttig metrikk å bruke for å evaluere ytelsen til maskinlæringsmodeller. Presisjonen er nyttig for å unngå falske positive klassifiseringer, hvilket gir verdifull innsikt i om modellen er i stand til å identifisere de to fenomenene uten å gjøre for mange feil. Som nevnt i punkt 2.4.6, kan presisjonen komme i konflikt med sensitiviteten til analysen. Derfor kan det være interessant å sammenligne disse metrikkene for å undersøke om det finnes sammenhenger mellom disse i klassifiseringen.

Tapet er en annen metrikk som er viktig å ta med i evalueringen av modellens ytelse. Tapet kan gi en indikasjon på hvor god modellen presterer på data som den ikke har sett før og uttrykkes med et tall fra 1.0 til 0 der lavere tall indikerer bedre resultat. Dette kan være spesielt nyttig for å undersøke om modellen overtilpasser eller undertilpasser til treningsdataene. Ved å beregne tapet for både mikroskopiske sorte hull og sfaleroner før og etter data augmentering, kan man undersøke om det blir innført skjevhet i modellen. Dersom tapet er høyt for en av kategoriene, kan dette indikere at modellen har problemer med å skille mellom bilder av dette fenomenet og bilder av det andre fenomenet.

5 Detaljert løsning

Dette kapittelet presenterer en detaljert løsning for maskinlæringsprosjektet, fra identifisering av problemet, til testing og evaluering. Alle trinnene i maskinlæringsprosessen vil bli gjennomgått, inkludert innsamling av data, forberedelse av data for trening, modellvalg, treningsprosessen, testing og evaluering. Bruken av teknikker for data augmentering for å forbedre modellens ytelse vil også bli beskrevet. Kapittelet gir dermed en helhetlig oversikt over hvordan den enkle og modifiserte maskinlæringsmodellen blir til.

5.1 Overordnet problem

Formålet med maskinlæringsprosjektet var å forbedre sensitiviteten i analysen av simulerte detektorbilder fra ATLAS ved hjelp av konvolusjonelle nevrale nettverk. Hovedutfordringen lå i å utforske hvordan symmetriegenskapene i bildene kunne utnyttes til data augmentering for å forbedre ytelsen til maskinlæringsmodellen. For å klargjøre problemet, ble følgende hovedpunkter tatt i betraktning:

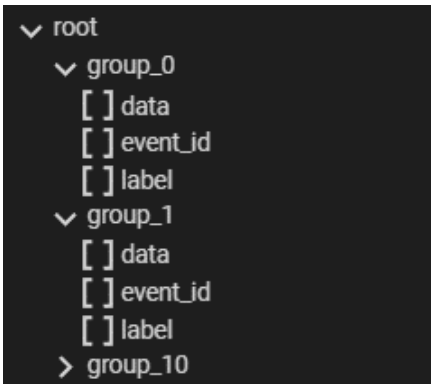
- Datasett: Bruke simulerte kollisjonsdata for å trene de konvolusjonelle nevrale nettverkene. Denne tilnærmingen gjør det mulig å bruke veiledet maskinlæring, ettersom modellen kan bli trent («lære») fra kjent data.
- Enkel modell: Lage en CNN-modell med minst mulig kompleksitet, men som fremdeles klarer å klassifisere mellom fenomenene. Den enkle modellen vil også fungere som et utgangspunkt for sammenligning med den modifiserte modellen.
- Bakgrunn for modifisering: ATLAS-detektoren har en sylindrisk form som gir en sylindersymmetri i detektorbildene. Partikkelkollisjonene i ATLAS gir også en høyre-venstresymmetri i bildene om kollisjonsaksen. Symmetriegenskapene kan dermed utnyttes til data augmentering.
- Ønsket utfall: Det første reduksjonslaget skal bli tilpasset slik at det tar hensyn til sylinder- og høyre-venstresymmetriene i detektorbildene. Deretter skal ulike teknikker for data augmentering benyttes, valgt med hensyn på å utnytte symmetriegenskapene. Målet er å gjøre dette uten å introdusere skjevhet i klassifiseringen, noe som kan føre til feilaktige resultater i analysen.
- Evaluering: Evalueringen av CNN-modellene vil foregå basert på evalueringsplanen gitt i punkt 4.5.

En tydelig problemdefinisjon var avgjørende for effektiv kommunikasjon og samarbeid med andre interessenter i prosjekter. Den skapte en felles forståelse og referanse for alle involverte, og sikret at alle jobbet mot samme mål. I andre maskinlæringsprosjekter der det ikke er forhåndsbestemte datasett og spesifisert hvilken type modell som skal brukes, fungerer denne fasen også for å identifisere hvilken type data som er nødvendig for å løse problemet. Det gir også et utgangspunkt for å kunne vurdere ulike tilnærminger og metoder som kan benyttes for å løse problemet.

5.2 Hente data

I denne delen av prosjektet ble datasettet som skulle benyttes til trening og testing av maskinlæringsmodellen, lest ut. Som nevnt i punkt 2.3 ble det utlevert et datasett bestående av to filer, der hver fil inneholder 15000 kollisjonsbilder. Filene er av typen HDF5-filer, som er et populært filformat for lagring av store mengder strukturert data. HDF5-filer er både kompakte og effektive, noe som gjør dem velegnet for lagring og gjenfinning av data i maskinlæringsprosjekter.

HDF5-filene er organisert i en hierarkisk struktur som ligner på et filsystem med mapper og filer. I dette prosjektet består hver HDF5-fil av en rot-mappe, som igjen inneholder undergrupper. Hver undergruppe representerer et unikt detektorbilde og består av tre elementer: *data*, *event_id* og *label*. Data inneholder selve 2D-histogrammene, event_id gir en unik identifikator for hvert bilde, og label angir hvilken klasse (mikroskopisk sort hull eller sfaleroner) hvert bilde tilhører. Nærmere forklaringer av hvert detektorbilde er gitt i punkt 2.3. Figur 5-1 viser et utklipp av hvordan undergruppene er strukturert.

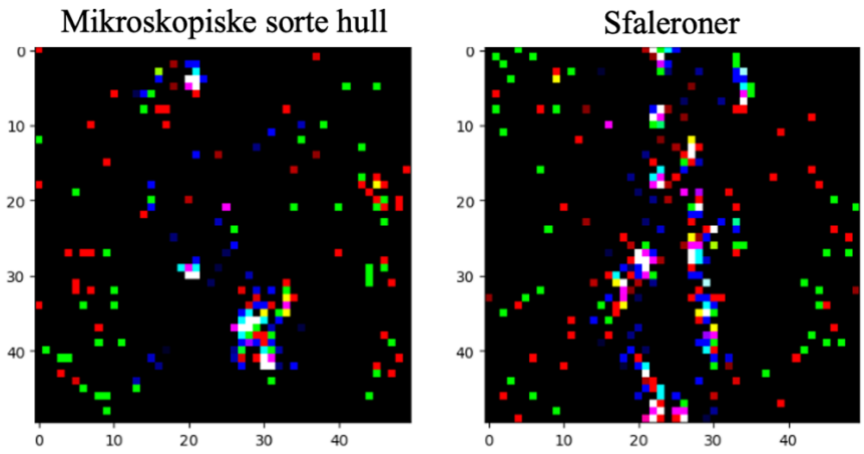


Figur 5-1 – Visualisering av hvordan hdf5 fil er strukturert.

For å lese ut data fra filene, ble hver HDF5-fil åpnet slik at det var mulig å hente ut en liste over alle nøklene, som hver representerer ulike undergrupper i filen. Deretter ekstraheres 2D-histogrammene knyttet til hver nøkkel, ved å få tilgang til data-elementet i hver undergruppe. Etter histogrammene er hentet, blir de konvertert til NumPy-tabeller (*Numpy-arrays*), som gjør dem enklere å jobbe med i løpet av maskinlæringsprosessen. NumPy-tabeller er en sentral datastruktur i Python for numeriske beregninger og er godt støttet av mange maskinlæringsbiblioteker.

5.3 Undersøke datasettet

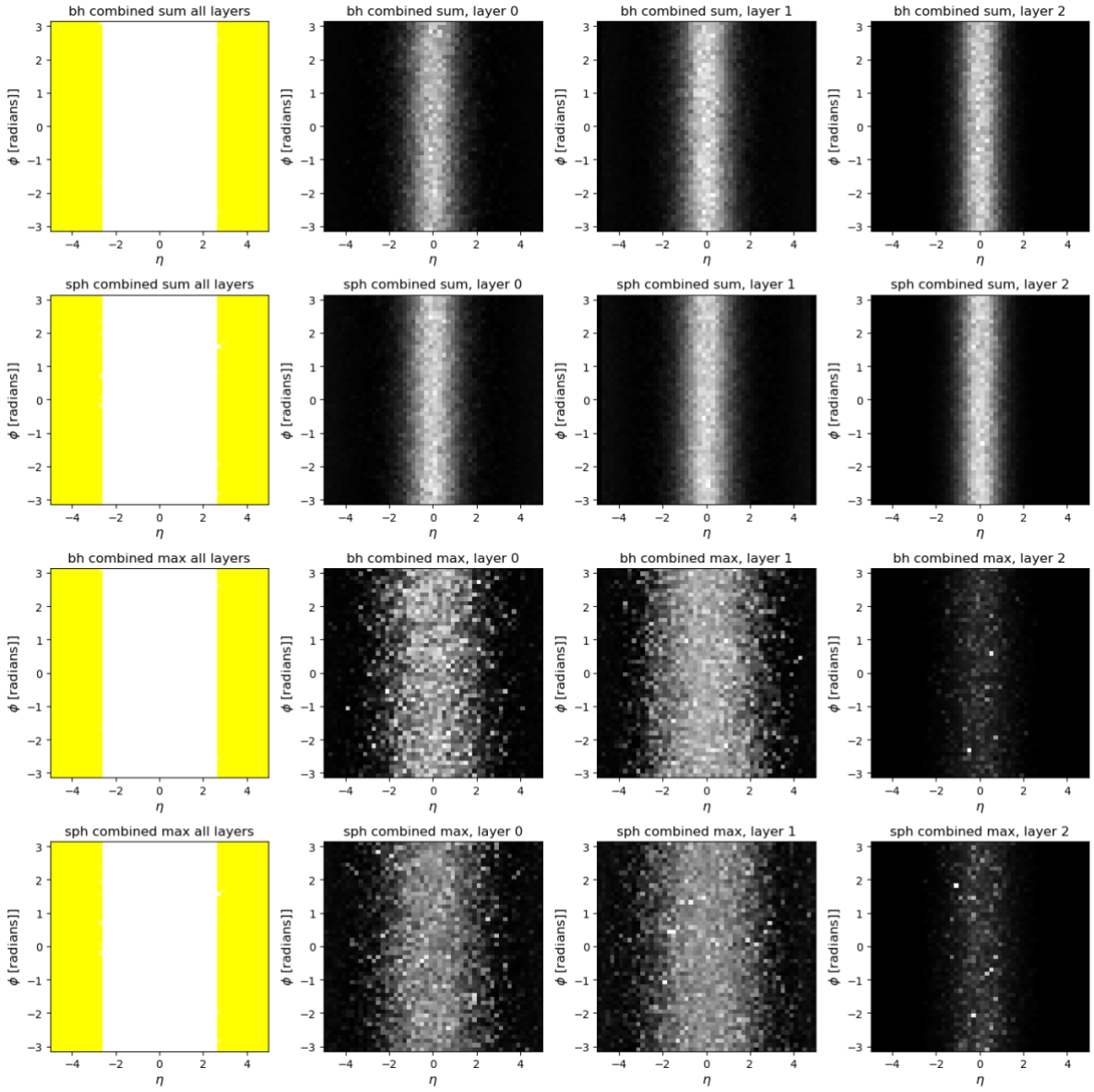
I den neste delen av prosjektet ble datasettet undersøkt for å få innsikt om hvordan bildene så ut. Bildene som ble behandlet er av størrelse 50x50 piksler og viser ulike deler av detektoren. Bildene var fargekodet med rød, grønn og blå for å indikere treff i forskjellige lag av detektoren. Hovedsakelig var bildene dominert av svarte områder, som signaliserte fravær av treff i detektoren i disse områdene. Figur 5-2 viser et eksempel på hvordan bildene av mikroskopisk sort hull og sfaleroner kunne se ut.



Figur 5-2 – Mikroskopisk sort hull (t.v.) og sfaleron (t.h.) etter de er transformert til NumPy-tabell.

For å analysere dataene nærmere, ble bildene for mikroskopiske sorte hull og bildene for sfaleron visualisert ved hjelp av ulike verktøy som 2D- og 3D-plott, slik Figur 5-3 er et eksempel på. Dette ga muligheten til å observere frekvensfordelingen og intensiteten av treff i detektoren, samt fordelingen av treff mellom de ulike lagene i detektoren. Disse visualiseringene bidro til å identifisere eventuelle mønstre og avvik i dataene, og kunne hjelpe med å øke forståelsen for datasettet.

I tillegg til visualiseringene ble også enkelte statistiske parametere beregnet for hvert datasett, som eksempelvis summen av treff[17] og maksimalverdien av treff[18], slik Figur 5-3 viser. Dette ga en indikasjon på forskjellene mellom de to datasettene, og bidro til å forstå hvilke egenskaper som kunne være viktige for maskinlæringsalgoritmen.



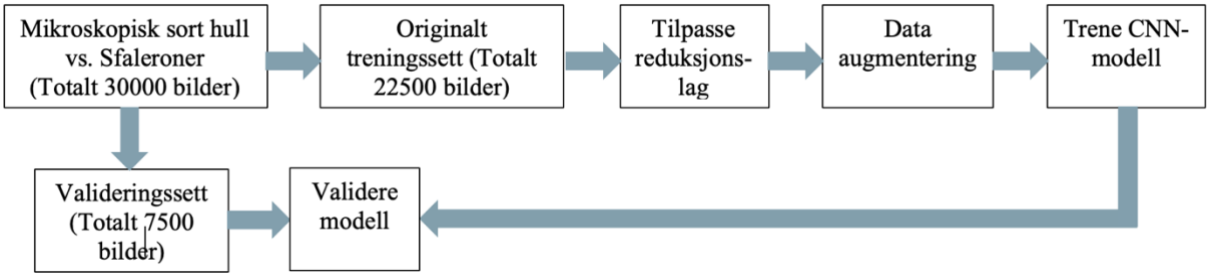
Figur 5-3 – Summen av treff for bildene mikroskopiske sorte hull (øverst) og sfaleroner (nest øverst), og maksimaltreff i bildene av mikroskopiske sorte hull (nest nederst) og sfaleroner (nederst).

[17] Summebildene, representert i den første og andre raden, viser den kumulative intensiteten av punkttreff på tvers av alle bildene i deres respektive klasser.
[18] Maksimalverdien av treff gir oss en indikasjon på de høyeste punktintensitetene som er observert i hver posisjon over hele datasettet, og for hvert lag.

5.4 Data forberedning

Før modellen kunne bli trent, måtte dataen først bli konvertert fra 2D-histogrammer til et format som er kompatibelt med PyTorch. Det ble gjort ved å transformere dataettet til DataLoader-objekter, som er en standard måte å representere data på i PyTorch. Dette gjør det mulig for modellen å behandle data effektivt og i mindre biter, kalt *batch*, som forenkler optimaliseringen av minnebruk og beregningshastighet.

Etter konverteringen, deles data inn i et treningssett og et testsett. Funksjonen *train_test_split()* ble brukt for å dele datasettet inn i et treningssett som bestod av 75% av datasettet og et testsett som bestod av 25% av datasettet, som vist i Figur 5-4. Treningssettet brukes til å trene modellen, mens testsettet brukes til å evaluere hvor godt modellen generelt presterer på nye, ukjente data. Ved å dele datasettet i et treningssett og et testsett, vil testsettet bli holdt uendret gjennom alle eksperimentene for en rettferdig sammenligning.



Figur 5-4 – Figuren viser splitting av data og videre treningsprosessen samt validering

Data augmentering er en annen viktig del av dataforberedelsen. Data augmentering er en teknikk for å kunstig øke mengden data, ved å lage nye, modifiserte kopier av eksisterende data med visse transformasjoner. I dette prosjektet benyttes symmetrien i dataene for å generere nye eksempler. Det ble brukt y-aksens forskyvning for å ta hensyn til den sylindriske symmetrien, speilvendt flipping av bilder på horisontal aksen for høyre-venstresymmetri, horisontal speiling og 180 graders rotasjon. Dette gjør at modellen blir mer robust og reduserer risikoen for at modelen blir overtilpasset treningsdataene.

Først flyttes aksen til bilder fra (-1) til (1) posisjon, og deretter for hvert bilde i treningssettet, utføres følgende data augmenterings transformasjoner:

- Horisontal speiling: Bildet speiles horisontalt
- Vertikal forskyvning: Bilder ruller langs y-aksen med en tilfeldig forskyvning
- 180 graders rotasjon: Bildet roteres 180 grader

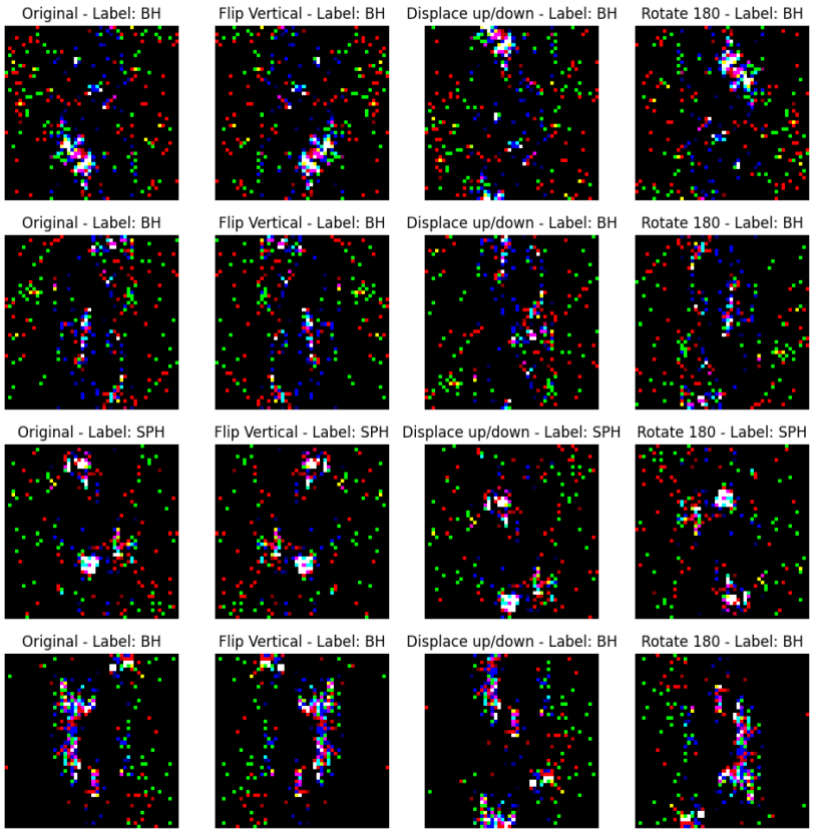
Etter transformeringen av bildene legges de til det augmenterte treningsdatasettet, noe som resulterer i en firedobling av størrelsen på treningsdatasettet. Dette gir et mer robust og sammenlignbart grunnlag for trening og evaluering gjennom hele prosjektet. Tilsvarende økes størrelsen på treningsdatasettet med fire ganger når modellen trenes med en enkelt data augmenteringsteknikk om gangen, og dette gir et bedre sammenligningsgrunnlag på tvers av teknikker for data augmentering. Etter at disse transformasjonene er utført, opprettes det en PyTorch DataLoader med en blanding av tilfeldig rekkefølge og en batch-størrelse på 150 for både det originale og augmenterte treningssettet. På samme måte blir også tekstdataene konvertert til et DataLoader-objekt.

For å illustrere prosessen med dataforberedelse og data augmentering kan referere til et bilde Figur 5-5 som viser hvordan dataene ser ut før og etter transformasjonene. Bildet gir en tydelig visualisering av effektene av y-aksens forskyvning, høyre-venstresymmetri, horisontal speiling og 180 graders rotasjon på dataene.

Ved å utføre disse trinnene for dataforberedelse, forventes det å utvikle en vellykket modell som kan lære mønstrene i dataene og gi nøyaktige prediksjoner på ukjente data.

Når det gjelder fordelingen av de augmenterte dataene og de originale dataene, gjelder forhold på 1:3 mellom originalt bilde og augmenterte bilder. For hvert originalt bilde genereres tre nye augmenterte bilder (horisontalt speilet, vertikalt forskjøvet og 180 graders rotert). Dermed blir det totale antallet bilder i det augmenterte treningssettet fire ganger større enn det opprinnelige treningssettet. Dette kan også observeres i koden der det ble brukt *np.repeat* for å gjenta treningssettets etiketter fire ganger for å matche det augmenterte datasettet.

Merk at data augmenteringen kun brukes på treningssettet og ikke på testsettet. Dette er fordi det er behov å evaluere modellen på ukjente data og se hvordan den generelt presterer på nye data som ikke er en del av treningsprosessen.



Figur 5-5 – Figur viser hvordan datasettet ser ut når data augmenterings teknikker utføres i forhold mot original

5.5 Valg av modell og arkitektur

CNN (Convolutional Neural Network) er en populær arkitektur for bildedataklassifisering. Her sammenlignes to forskjellige CNN-arkitekturer og deres forskjeller. Først presenterer en enkel CNN-modell med standard gjennomsnittlig-pooling operasjon, som er vanlig brukt i mange bildeklassifiseringsoppgaver. Deretter ser på en mer tilpasset CNN-modell med en egendefinert symmetrisk poolingfunksjon som tar hensyn til symmetrier i bildet. Ved å undersøke og sammenligne disse to modellene, dette vil gi oss en bedre forståelse av hvordan valg av arkitektur kan påvirke ytelsen til en CNN-modell.

5.5.1 Enkel CNN (ConvModel)

ConvModel er en mer avansert versjon av et egendefinert konvolusjonelt nevralt nettverk (CNN). Den består av fem konvolusjonslag, etterfulgt av batch-normalisering og ReLU-aktiveringsfunksjoner, og gjennomsnittlig-poolinglag. Utskriften av det femte konvolusjonslag flates ut og mates til tre fullt tilkoblede lag, også med ReLU-aktiveringsfunksjoner og

utkobling. Til slutt passerer utgangen gjennom det tredje fullt tilkoblede laget som representerer de to klassene som modellen er designet for å klassifisere.

Her er en detaljert beskrivelse av strukturen og terminologier for ConvModel:

1. **Konvolusjonslag:** Modellen inneholder fem konvolusjonslag (self.conv1, self.conv2, self.conv3, self.conv4, og self.conv5). Disse lagene lærer lokale mønstre i bildet ved å anvende et sett med filtre over hele bildet. For hvert konvolusjonslag er antall inngangskanaler, antall utgangskanaler (filtre) og kjerne (filter) størrelse spesifisert. Utfylling er også satt til 1 for alle lagene, noe som betyr at inngangs bilder blir "innrammet" med nuller rundt kantene for å holde funksjonskartets størrelse lik gjennom konvolusjonen.
2. **Batch-Normalisering:** Batch normalisering er en teknikk som brukes for å forbedre hastigheten, ytelsen og stabiliteten til CNN-modeller. Den normaliserer funksjonskartene i hvert lag ved å sentrere og skalere dem basert på minibatch-statistikker. I ConvModel er det fem batch normaliseringslag (self.bn1, self.bn2, self.bn3, self.bn4, og self.bn5) som ligger rett etter hvert konvolusjonslag.
3. **ReLU-aktiveringsfunksjon:** ReLU (Rectified Linear Unit) er en ikke-lineær aktiveringsfunksjon som brukes etter hvert batch normaliseringslag.
4. **Gjennomsnittlig- pooling:** Etter hver ReLU-aktiveringsfunksjon følger hvert reduksjonslag med gjennomsnittlig pooling (F.avg_pool2d). Dette laget reduserer den romlige størrelsen på funksjonskartene ved å utføre en gjennomsnittlig beregning innenfor et definert vindu. Hvert reduksjonslag har en nedskalert faktor spesifisert som argument, som bestemmer hvor mye størrelsen på funksjonskartene skal reduseres.
5. **Fullt tilkoblede lag:** Etter det siste gjennomsnittlig- reduksjonslaget blir funksjonskartene flatet og sendt gjennom tre fullt tilkoblede lag (self.fc1, self.fc2 og self.fc3). Hvert fullt tilkoblede lag har forskjellige dimensjoner:
    a. Det første fullt tilkoblede laget tar inn en inngangsvektor med størrelse 1x1x1024 og transformerer den til en større vektor med størrelse 1x2048.
    b. Det andre fullt tilkoblede laget tar inn en vektor med 1x2048 elementer og reduserer den til en vektor med størrelse 1x512.
    c. Det tredje fullt tilkoblede laget tar inn en vektor med 1x512 elementer og reduserer den til en vektor med størrelse 1x2, som tilsvarer de to klassene modellen er designet for å klassifisere.

6. **Utkobling**: For å forhindre overtilpasning benytter modellen utkobling-regulering (self.dropout) etter det første og andre fullt tilkoblede laget og før det tredje fullt tilkoblede laget. Utkobling fungerer ved å tilfeldig "slå av" en andel av nevronene i laget under trening med en spesifisert sannsynlighetsverdi også kalt (dropout rate). Dette hjelper modellen med å bli mer robust og forhindrer den i å bli for avhengig av enkelte nevroner.

Gjennom denne sekvensen av konvolusjonslag, batch normalisering, aktiveringsfunksjoner, reduksjonslag med gjennomsnittlig pooling og fullt tilkoblede lag lærer modellen å gjenkjenne og klassifisere de to klassene den er designet for.

Koden til ConvModel-klassen definerer alle disse lagene og prosessene i konstruktøren og forward-metoden. Konstruktøren initialiserer lagene, mens forward-metoden beskriver hvordan inndataene skal passere gjennom lagene for å produsere en utgang.

5.5.2 CNN med første reduksjon laget tilpasning (ConvModelFPLMod)

ConvModelFPLMod er en variant av en konvolusjonell nevral nettverksarkitektur (CNN) som er tilpasset med en egendefinert reduksjon funksjon som tar hensyn til symmetrier i bildet. Denne modellen består av fem konvolusjonslag, med en batch normalisering og ReLU-aktiveringsfunksjon etter hver konvolusjonsoperasjon, etterfulgt av reduksjonslag med gjennomsnittlig-pooling operasjon. Modellen består også av tre fullt tilkoblede lag med ReLU-aktiveringsfunksjon og dropout-regulering.

Forskjellen mellom ConvModel og ConvModelFPLMod er at ConvModel bruker reduksjonslaget med gjennomsnittlig pooling operasjonen i hvert konvolusjonslag som reduserer dimensjonaliteten i utgangen fra konvolusjonslag med en faktor på 2. I ConvModelFPLMod er det første reduksjons laget erstattet med en egendefinert symmetrisk

funksjon som reduserer med hensyn til symmetrier i bildet. Denne funksjonen kalles symmetri-pooling og er definert som følger:

```python
# Custom pooling layer
def circular_shift(tensor: Tensor, shift: int, dim: int):
    return torch.cat((tensor[:, :, -shift:], tensor[:, :, :-shift]), dim=dim)

def symmetry_pooling(x: Tensor, shift: int):
    x_lr = torch.flip(x, dims=[3])  # Left-right symmetry
    x_ud = circular_shift(x, shift, dim=2)  # CIrcular shift for cyllindrical symmetry
    x_lr_ud = torch.flip(x_ud, dims=[3])  # 180-degree rotation

    return (x + x_lr + x_ud + x_lr_ud) / 4
```

Figur 5-6 – Utsnitt for koden til tilpasning av første pooling laget

Funksjonen symmetry_pooling utfører en symmetrisk reduksjonsoperasjon på inngang x ved å kombinere inngang x med en venstre-høyre speiling av x (x_lr), en sirkulær forskyvning av x med en faktor på shift (x_ud), og en 180 graders rotasjon av x_ud (x_lr_ud). Resultatet er gjennomsnittet av disse fire inngangsverdier.

Resten av nettverksarkitekturen i ConvModelFPLMod er lik ConvModel, bortsett fra at det første konvolusjonslag bruker symmetry_pooling i stedet for reduksjonslaget med gjennomsnittlig – pooling .

Til slutt, utgangen fra det siste fullt tilkoblede laget passerer gjennom en kryssentropi-tap-funksjon. Denne funksjonen beregner tap ved å vurdere sannsynlighetsfordelingen av de forutsagte klassene, og straffer mer for sikre, men feilaktige, prediksjoner. Denne teknikken bidrar til å justere modellens vekter for å forbedre presisjonen under opplæringsprosessen. Dropout-regulering brukes etter det første og andre fullt tilkoblede laget for å redusere overtilpasning.

I sum har ConvModelFPLMod en egendefinert reduksjons funksjon som tar hensyn til symmetrier i bildet i stedet for å bruke standard gjennomsnittlig – pooling operasjon. Dette kan forbedre modellens evne til å lære distinkte funksjoner og øke ytelsen i bildedata-klassifiseringsoppgaver som krever håndtering av symmetrier.

Dette kan bidra til å redusere antall parametere i modellen og øke dens evne til å generalisere, siden gjennomsnittlig pooling gir en mer robust representasjon av bildeegenskaper enn

maksimal pooling. Sammenlignet med den enkle CNN-modellen fra forrige kapittel, har ConvModelFPLMod en tilpasset poolingfunksjon. Dette kan føre til bedre ytelse og evne til å generalisere, spesielt når det gjelder å håndtere symmetrier i bildene.

Mer detaljert forklaring kan man lese i koden til prosjektet på en offentlig GitHub oppbevaringssted[19].

5.6 Trening

Når modellen er bygget, defineres en treningsfunksjon. Treningsfunksjonen er den sentrale komponenten i treningsprosessen. Den tar imot flere argumenter, inkludert modellen, datalastere for trenings- og testdata, optimerer, tapkriterium og antall trenings-epoker. Funksjonen går deretter i gang med å trene modellen over det angitte antall epoker.

En optimerer er en algoritme som justerer modellens parametere, basert på gradientene som beregnes under treningsprosessen. Målet er å minimere tapet, som er et mål for hvor godt modellens prediksjoner stemmer overens med de faktiske etikettene. I dette tilfellet brukes Adam-optimereren med en vekt forfall på 1e-4. Vekt forfall *(weight decay)* bidrar til å redusere overtilpasning ved å legge til en strafferamme for store vekter i tapsfunksjonen.

Tapsfunksjonen, også kalt tapkriterium, måler forskjellen mellom modellens prediksjoner og de faktiske etikettene. Ved å minimere dette tapet, blir modellen bedre til å forutsi riktige etiketter for ukjente data. Her brukes kryss entropitap, som er en vanlig tapsfunksjon for klassifikasjonsproblemer.

For hver epoke settes modellen i treningsmodus og itererer over treningsdataene. En epoke er en enkelt gjennomgang av hele treningsdatasettet. Antall epoker er en hyperparametre som bestemmer hvor mange ganger modellen vil iterere over datasettet. Å øke antall epoker kan forbedre modellens ytelse, men det kan også føre til overtilpasning hvis modellen trenes for mange epoker.

[19] Lenken til GitHub oppbevaringssted: https://github.com/591308/Optimazing-klassification-of-ATLAS-detector-data--CERN-

I tillegg til Adam-optimereren og kryssentropitapet brukes også en planlegger *(scheduler)* som endrer læringshastigheten under treningen for å unngå overtilpasning. Her ble brukt ReduceLROnPlateau-scheduleren, som overvåker tapet på valideringssettet og reduserer læringshastigheten med en faktor på 0.5 når tapet ikke forbedres i løpet av et visst antall epoker.

Prosessen for hver batch med data kan beskrives i følgende trinn:

1. Inndataene og etikettene flyttes til enheten (GPU). Dette gjøres for å øke hastighet for trening av modellen ved bruk av GPU sine egenskaper for å gjøre tusenvis av parallelle beregninger samtidig.
2. Gradientene til modellparametrene nullstilles for å forhindre akkumulering av gradienter fra tidligere iterasjoner.
3. Modellen beregner utdataene for de gitte inndataene.
4. Tapet beregnes ved hjelp av det angitte tapkriteriet ved å sammenligne utdataene og de faktiske etikettene.
5. Tapet tilbakeføres gjennom modellen, som beregner gradientene for hver modellparameter.
6. Optimereren oppdaterer modellparametrene ved hjelp av de beregnede gradientene.

Etter at treningsprosessen er fullført, evalueres modellen på et testdatasett som ikke har blitt brukt under treningen. Dette gir en indikasjon på hvor godt modellen vil prestere på nye, ukjente data. Evalueringen inkluderer beregning av metrikker som nøyaktighet, tap, presisjon og forvirringsmatrise for å vurdere modellens ytelse og sensitivitet.

Det er viktig å balansere modellens kapasitet til å lære av treningsdataene og evnen til å generalisere til ukjente data. En modell som er for enkel kan ikke lære de underliggende mønstrene i dataene og vil ha dårlig ytelse både på trenings- og testdata, noe som kalles underjustering. På den annen side kan en modell som er for kompleks og trenes for lenge, bli for tilpasset treningsdataene og miste evnen til å generalisere, noe som fører til overtilpasning. Dette er grunnen til at det er viktig å velge riktig antall epoker og andre hyperparametre for å sikre at modellen ikke over- eller undertilpasse.

I løpet av trenings- og evalueringsprosessen kan ulike hyperparametre justeres også gir mulighet til å eksperimentere med ulike arkitekturer for modellen for å finne den beste

kombinasjonen som gir optimal ytelse på testdataene. Selv om hovedmålet i dette prosjektet er tilpasning av det første poolingslaget og data augmentering, er det fortsatt viktig å evaluere modellens ytelse og unngå overtilpasning.

5.7 Test og evaluering

Etter hver epoke med trening evalueres modellen på testdatasettet. For å gjøre dette, settes modellen i evalueringsmodus ved hjelp av model.eval(), som påvirker oppførselen til enkelte lag i nettverket, for eksempel dropout-lag, slik at de fungerer korrekt under evaluering.

Under evaluering er gradientberegning deaktivert ved hjelp av kontekstbehandleren torch.no_grad(). Kontekstbehandleren er en teknikk i Python som tillater midlertidige endringer i oppførselen til en funksjon eller en del av koden. I dette tilfellet deaktiverer kontekstbehandleren gradientberegning under evalueringen for å spare minne og beregningsressurser, ettersom det ikke er nødvendig å oppdatere modellvektene i evalueringsfasen.

Test-tapet beregnes ved hjelp av det angitte tapkriteriet kryssentropitap, og testnøyaktigheten beregnes ved å sammenligne utdataene med de faktiske etikettene. Evalueringsløkken beregner også separate nøyaktigheter for sorte hull og sfaleroner ved å telle opp korrekte forutsigelser for hver klasse og dele dem med antall eksempler i hver klasse. Videre beregnes presisjon og gjenskallingsmålinger for begge klasser ved hjelp av precision_recall_fscore_support metoden fra sklearn.metrics.

Gjennom trenings- og testing prosessen genereres en verdifull samling av data i form av en liste med ordbøker, betegnet som metrics_per_epoch. Ved slutten av hver epoke blir trenings- og test-tap, nøyaktighet, presisjon og sensitivitet lagt til i sine respektive lister. Disse listene katalogiserer målingene for hver epoke, noe som muliggjør enkel visualisering av modellens ytelse gjennom treningsprosessen. Målingene samles også i en variabel kalt metrics som inneholder alle metrikkene for hver epoke, samt alle forutsigelser og etiketter.

Listen metrics_per_epoch arkiverer disse ordbøkene for hver epoke, noe som gir muligheten til å lage grafer, forvirringsmatriser og tabellariske presentasjoner av metrikkene i etterkant. Disse visualiseringene kan deretter brukes for å presentere og diskutere resultater.

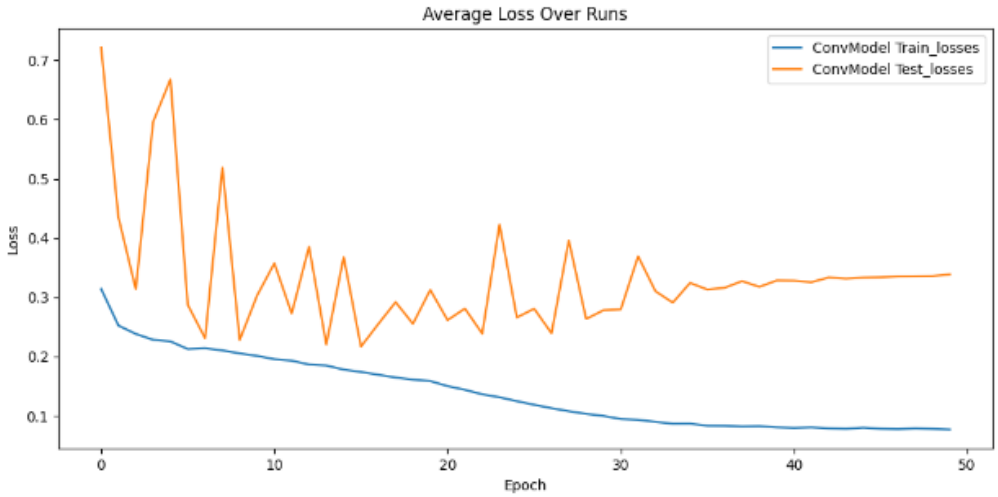
Denne prosessen gir ikke bare de beste modellvektene (best_model_state) som kontinuerlig oppdateres gjennom hver epoke med hensyn til tap, men muliggjør også henting av modellen med laveste tap på testsettet gjennom hele treningsprosessen. Dette omfattende datasettet gir mulighet for både plotting av resultater og dypere analyse av treningsprosessen og evalueringen. Det er viktig å merke seg at analysen og forvirringsmatrisen er basert på den beste modellen identifisert gjennom treningsprosessen, ikke nødvendigvis modellen fra den siste epoken.

6 Resultat

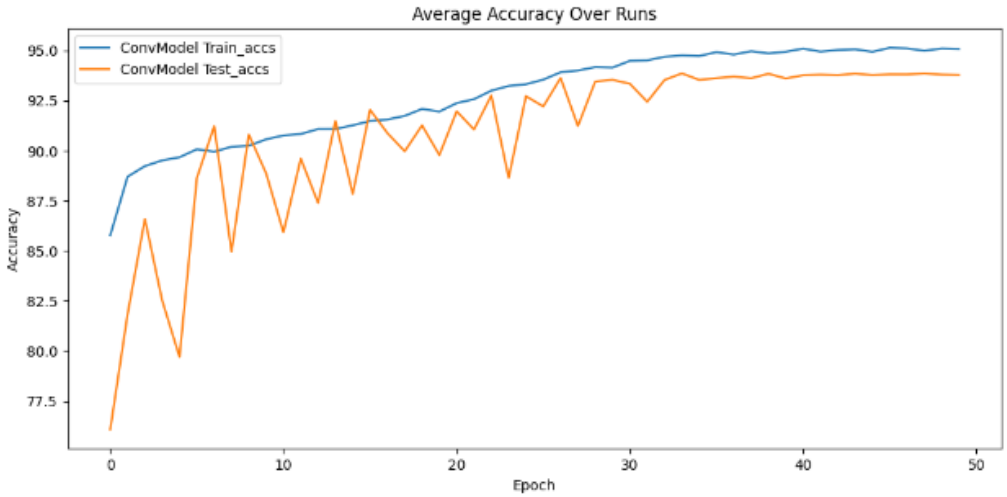
Dette kapittelet presenterer resultatene av maskinlæringsprosjektet. Resultatene gir innsikt i modellens evne til å klassifisere mikroskopiske sorte hull og sfaleroner.

6.1 Referansegrunnlag (ConvModel)

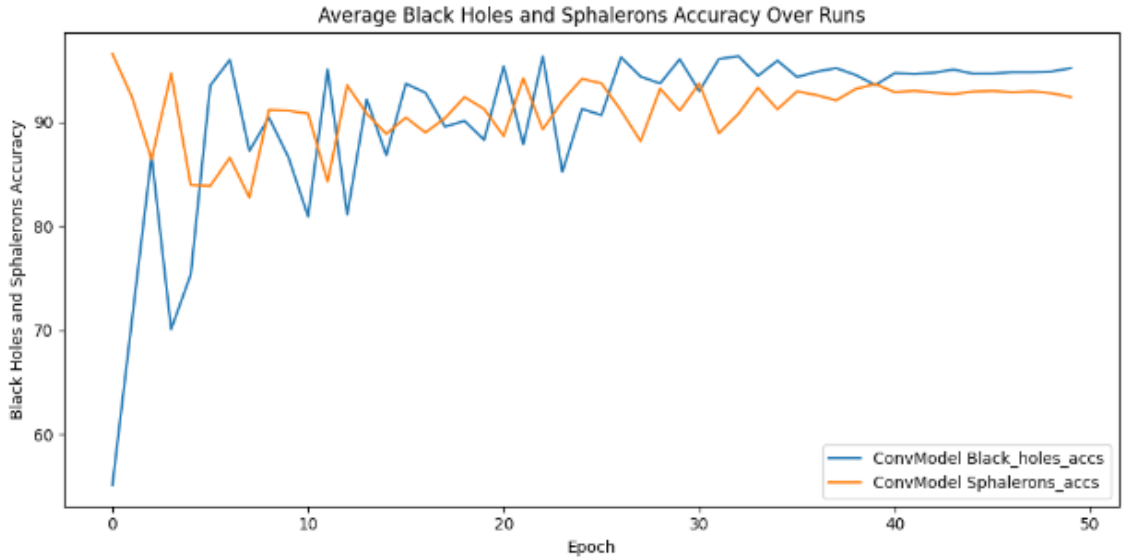
Nedenfor viser Figur 6-1 til Figur 6-4 resultatene fra kjøringen av den enkle CNN-modellen, kalt *ConvModel*, uten data augmentering. Det vil si at den ikke tar hensyn til symmetriegenskapene i detektorbildene i det første reduksjonslaget og modellen trenes på det opprinnelige treningssettet. Modellen ble kjørt fem ganger med 50 epoker i hver gjennomkjøring, og gjennomsnittet av resultatene ble beregnet for å oppnå en mer pålitelig tilnærming.



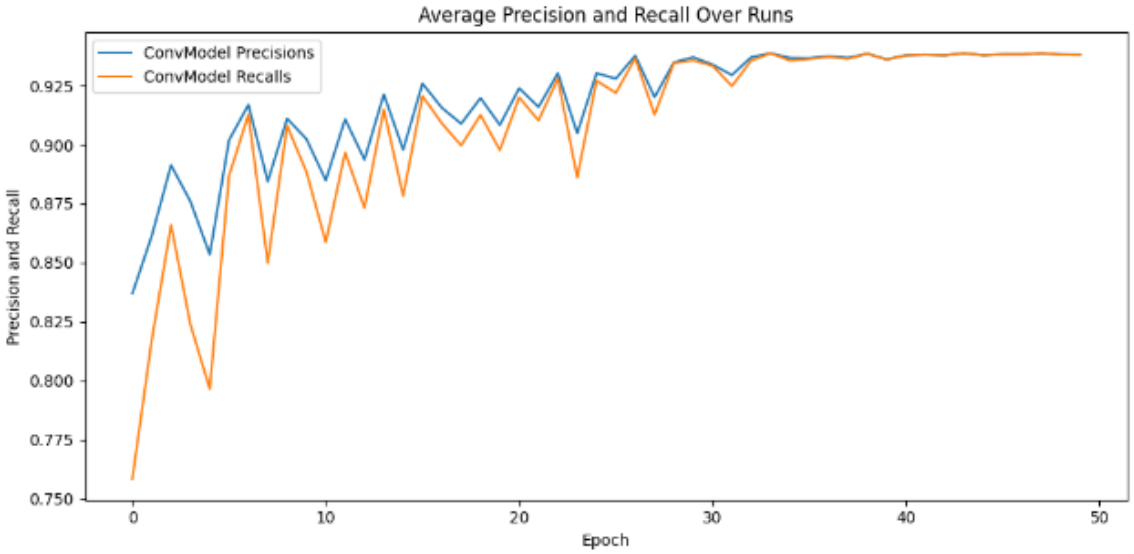
Figur 6-1 – Gjennomsnittlig tap av kjøringene til treningssettet (blå graf) og testsettet (oransje graf) for den enkle modellen.



Figur 6-2 – Gjennomsnittlig nøyaktighet av kjøringene til treningssettet (blå graf) og testsettet (oransje graf) for den enkle modellen.

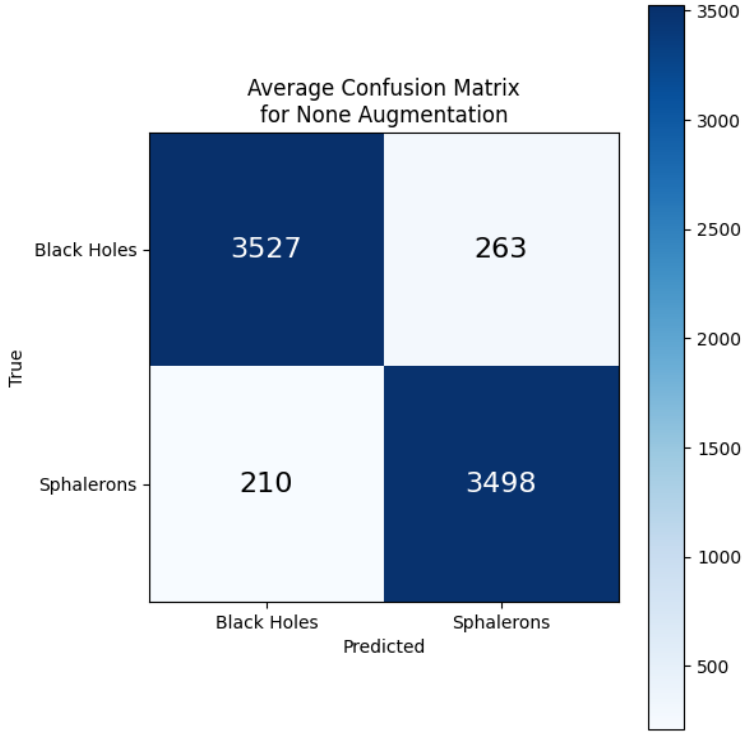


Figur 6-3 – Gjennomsnittlig nøyaktighet av mikroskopiske sorte hull (blå graf) og sfaleroner (oransje graf) for testsettet til den enkle modellen.



Figur 6-4 – Gjennomsnittlig presisjon (blå graf) og sensitivitet (recall) (oransje graf) for testsettet til den enkle modellen.

Forvirringsmatrisen gitt i Figur 6-5 gjelder for gjennomsnittet av kjøringene av ConvModel.



Figur 6-5 – Forvirringsmatrise for ConvModel som er uten data augmentering eller tilpasning av det første reduksjonslaget.

Tabell 6-1 og Tabell 6-2 viser gjennomsnittet av resultatene av fem kjøringer med 50 epoker av ConvModel uten data augmentering, evaluert med hensyn på henholdsvis nøyaktighet, sensitivitet og presisjon samt standardavviket.

Tabell 6-1 – Resultatet av basistilfellet for prosjektet med hensyn på total nøyaktighet av klassifiseringen.

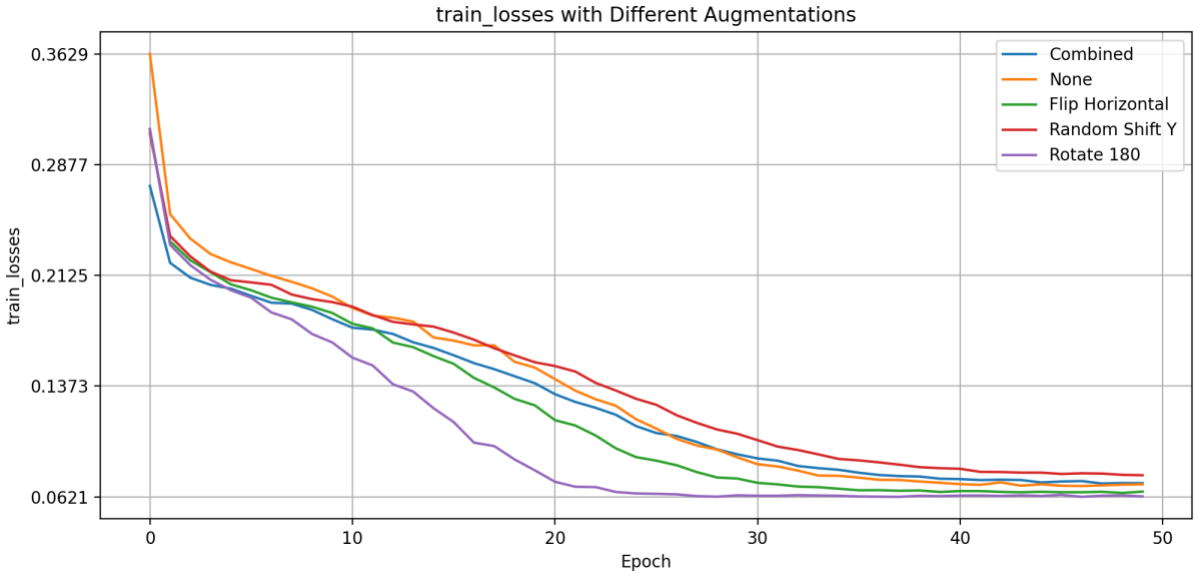
Enkel CNN	**Tap Treningssett**	**Nøyaktighet Treningssett**	**Tap Testsett**	**Nøyaktighet Testsett**
basistilfelle	0,13 ± 0,07	93,00 ± 2,56	0,43 ± 0,20	89,63 ± 4,90

Tabell 6-2 – Resultatet av basistilfellet for prosjektet med hensyn på sensitivitet, presisjonen og nøyaktighet for hvert av fenomenene, på testsettet.

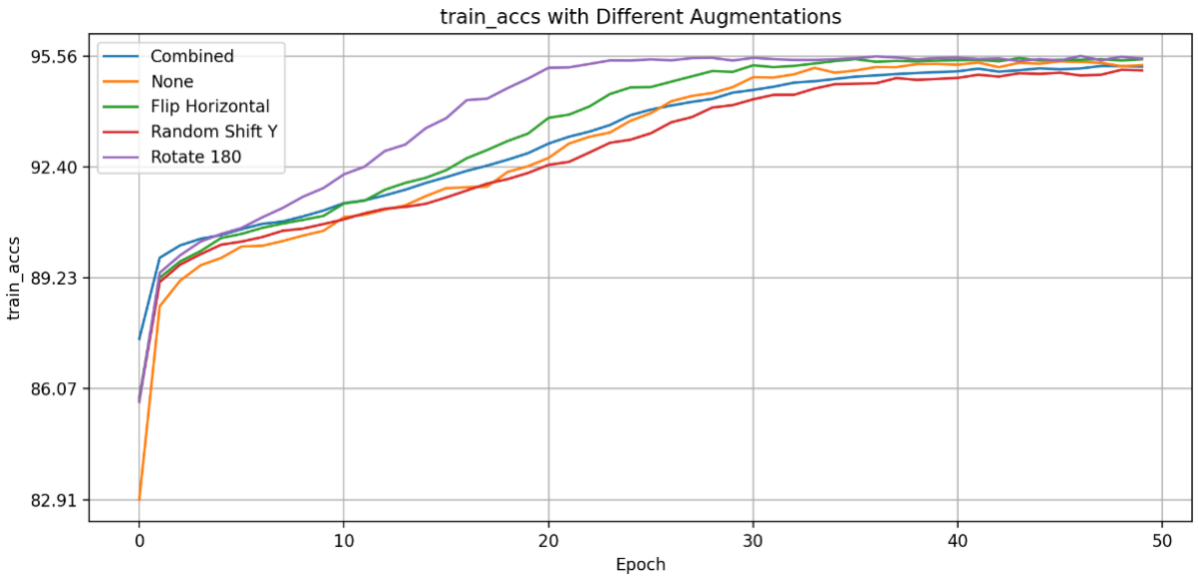
Enkel CNN	**Sensitivitet Testsett**	**Presisjon Testsett**	**Nøyaktighet Mikroskopisk sort hull**	**Nøyaktighet Sfaleron**
basistilfelle	0,90 ± 0,05	0,91± 0,03	87,45 ± 7.31	91,77 ± 3,08

6.2 Enkel CNN med data augmentering (ConvModel)

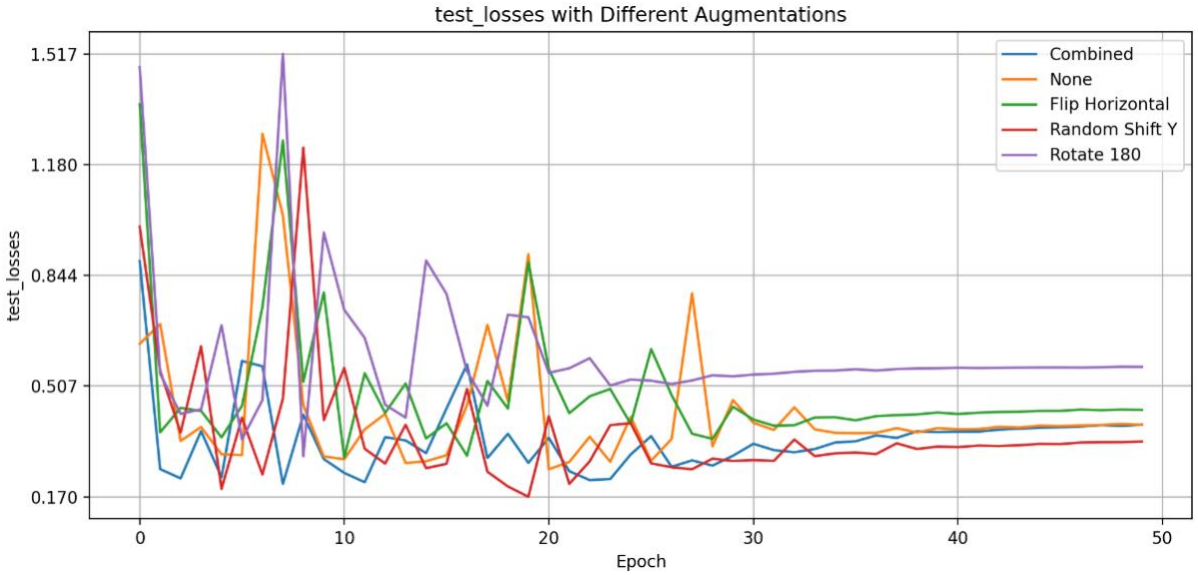
Nedenfor viser Figur 6-6 – Figur 6-9 resultatene av den enkle CNN-modellen, ConvModel, der det har blitt brukt ulike teknikker for data augmentering på treningssettet. De ulike teknikkene for data augmentering blir representert i figurene som grafer med egendefinerte farger. Modellen ble kjørt fem ganger med 50 epoker i hver gjennomkjøring, slik som i basistilfellet, og gjennomsnittet av resultatene ble beregnet.



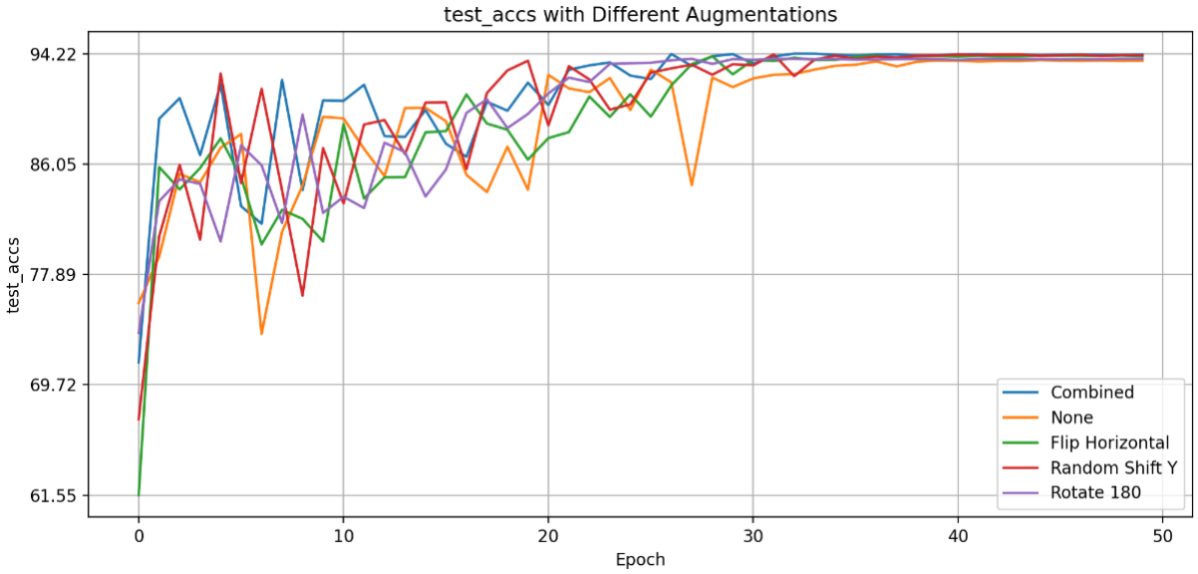
Figur 6-6 – Gjennomsnittlig tap av treningssettet til ConvModel, med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).



Figur 6-7 – Gjennomsnittlig nøyaktighet av treningssettet til ConvModel, med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).



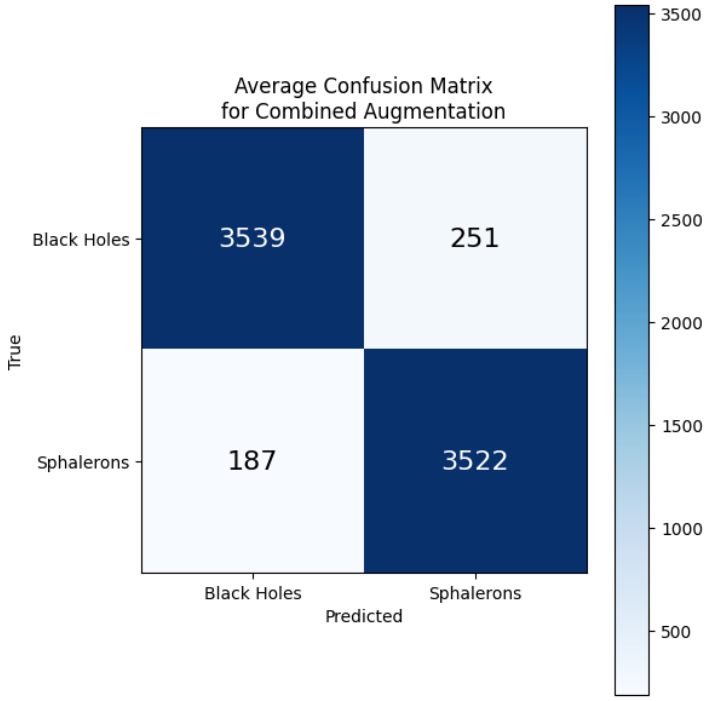
Figur 6-8 – Gjennomsnittlig tap av testsettet til ConvModel, med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).



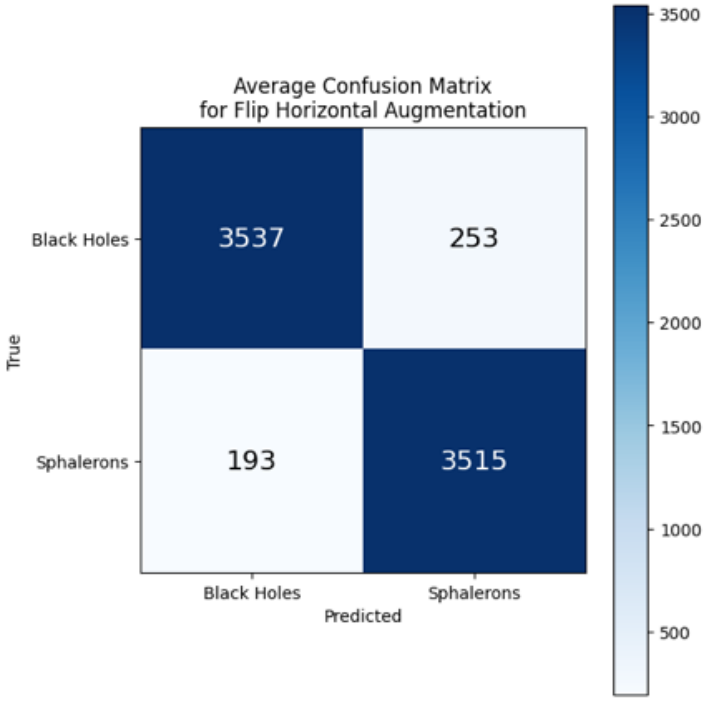
Figur 6-9 – Gjennomsnittlig nøyaktighet av testsettet til ConvModel, med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).

Forvirringsmatrisene i Figur 6-10 til Figur 6-13 gjelder for gjennomsnittet av kjøringene av ConvModel med de ulike teknikkene for data augmentering i treningssettet. Øverst til venstre er forvirringsmatrisen for modellen der alle teknikkene for data augmentering er kombinert. Øverst til høyre er forvirringsmatrisen for ConvModel der speilvending om den horisontale aksen (x-aksen) er benyttet for å utnytte høyre-venstresymmetrien i detektorbildene. Nederst til venstre er forvirringsmatrisen for modellen der tilfeldige forskyvninger langs den vertikale aksen (y-aksen) er brukt for å utnytte sylindersymmetrien i detektorbildene. Nederst til høyre

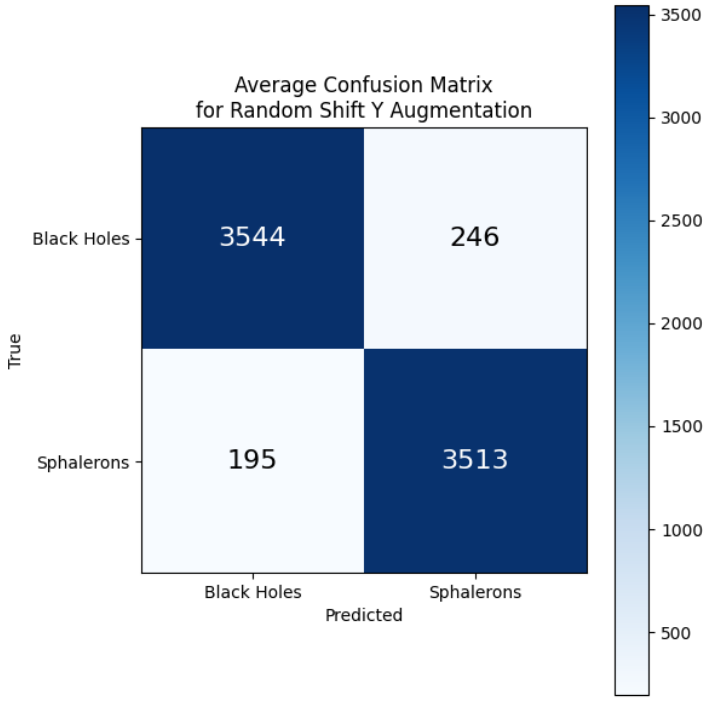
er forvirringsmatrisen for ConvModel der 180 graders rotasjon er brukt for å utnytte høyre-venstresymmetri.



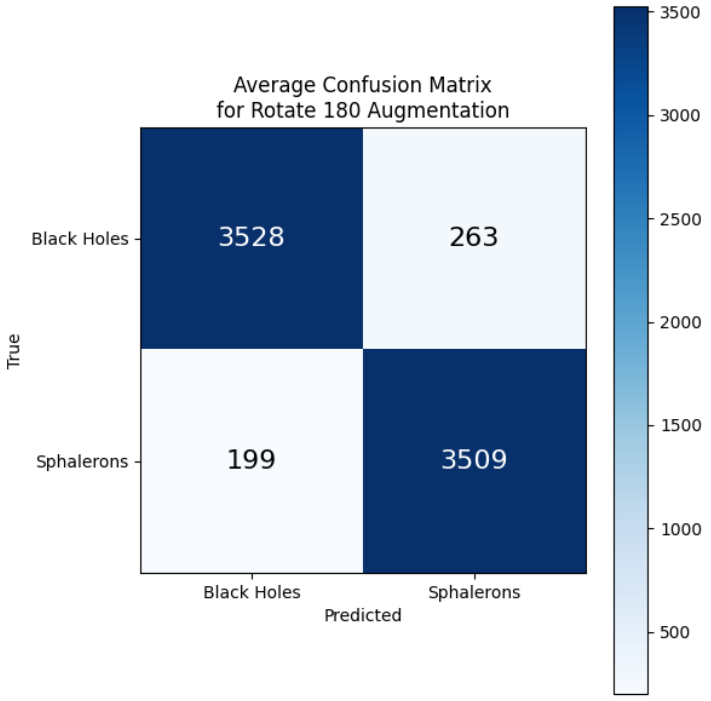
Figur 6-10 – Forvirringsmatrise for ConvModel med kombinasjon av alle teknikker for data augmentering.



Figur 6-11 – Forvirringsmatrise for ConvModel der horisontal flipp er brukt på treningssettet.



Figur 6-12 – Forvirringsmatrise for ConvModel der tilfeldig skift i y-aksen er brukt på treningssettet.



Figur 6-13 – Forvirringsmatrise for ConvModel der 180 graders rotasjon er brukt på treningssettet.

Tabell 6-3 og Tabell 6-4 viser gjennomsnittet av resultatene av de fem kjøringene med 50 epoker av ConvModel med data augmentering samt deres standard avvik, evaluert med hensyn på henholdsvis nøyaktighet, sensitivitet og presisjon. Evalueringen gjøres av alle teknikker kombinert, og med hver av teknikkene hver for seg.

Tabell 6-3 – Resultatene etter data augmentering er brukt, evaluert med hensyn på total nøyaktighet av klassifiseringen.

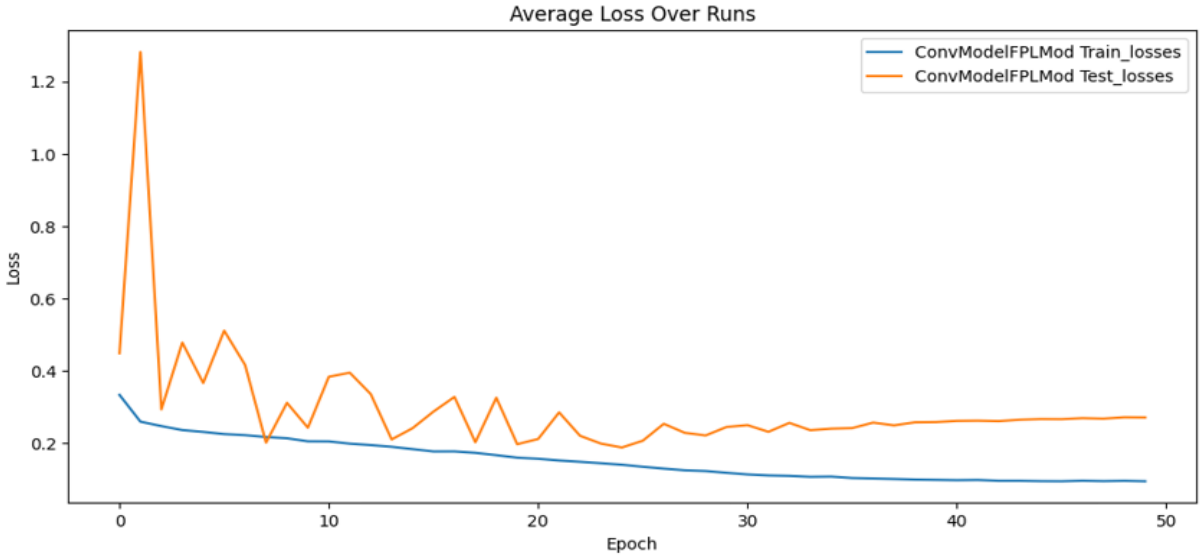
Enkel CNN	**Tap Treningssett**	**Nøyaktighet Treningssett**	**Tap Testsett**	**Nøyaktighet Testsett**
Kombinert	0,13 ± 0,05	93,24 ± 1,94	0,35 ± 0,11	91,42 ± 4,25
Forskyvning på y-aksen (sylindersymmetri)	0,14 ± 0,06	92,86 ± 2,12	0,36 ± 0,18	90,47 ± 5,40
Speilvendt på x-aksen (Høyre-venstresymmetri)	0,12 ± 0,06	93,51 ± 2,27	0,49 ± 0,20	89,59 ± 5,81
180 graders rotasjon (Høyre-venstresymmetri)	0,10 ± 0,06	94,03 ± 2,22	0,60 ± 0,22	90,21 ± 4,85

Tabell 6-4 – Resultatene etter data augmentering er brukt, evaluert med hensyn på sensitivitet, presisjon og nøyaktighet for hvert av fenomenene, på testsettet.

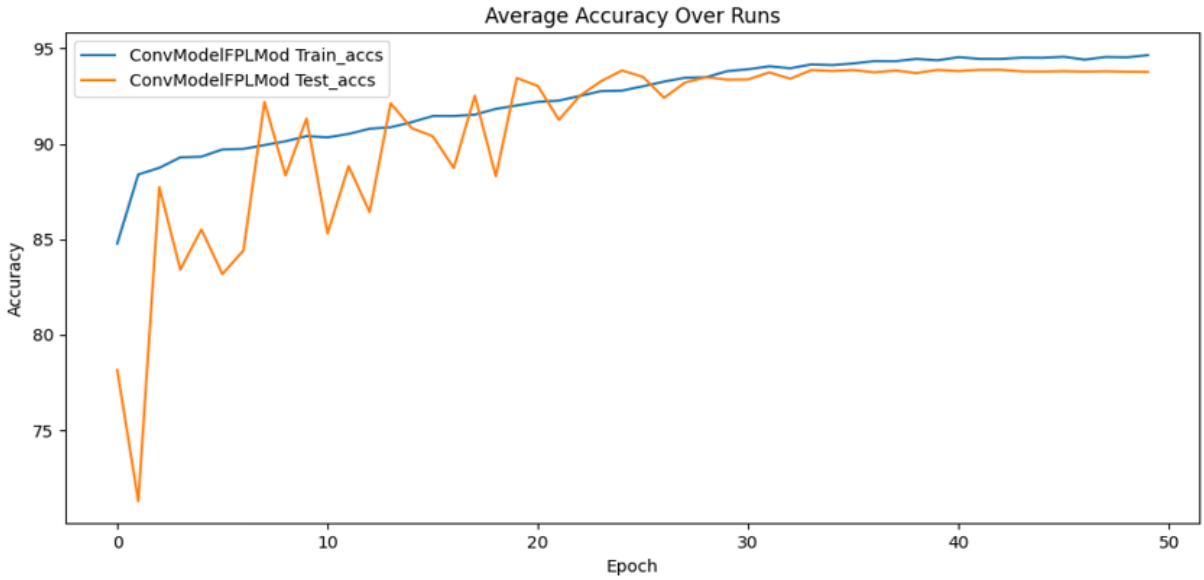
Enkel CNN	**Sensitivitet Testsett**	**Presisjon Testsett**	**Nøyaktighet Mikroskopisk sort hull**	**Nøyaktighet Sfaleron**
Kombinert	0,91 ± 0,04	0,92 ± 0,03	91,22 ± 6,68	91,61 ± 3,99
Forskyvning på y-aksen (sylindersymmetri)	0,90 ± 0,05	0,92 ± 0,03	88,30 ± 11,71	92,59 ± 3,13
Speilvendt på x-aksen (Høyre-venstresymmetri)	0,90± 0,06	0,91 ± 0,03	87,81 ± 7,13	91,32 ± 4,35
180 graders rotasjon (Høyre-venstresymmetri)	0,90 ± 0,05	0,92 ± 0,03	89,62 ± 8,30	90,79 ± 3,98

6.3 Modifisert CNN uten data augmentering (ConvModelFPLMod)

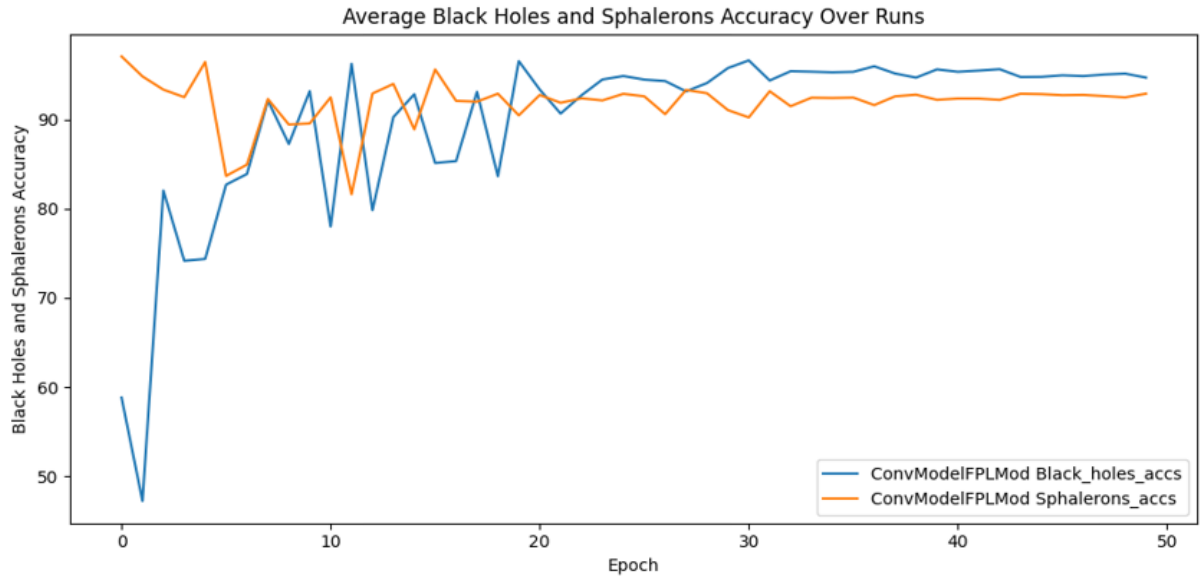
Nedenfor viser Figur 6-14 til Figur 6-17 resultatene fra kjøringen av den modifiserte CNN-modellen, kalt *ConvModelFPLMod*, uten data augmentering. Det vil si at den tar hensyn til symmetriegenskapene i detektorbildene i det første reduksjonslaget og modellen trenes på det opprinnelige treningssettet. Modellen ble kjørt fem ganger med 50 epoker i hver gjennomkjøring, og gjennomsnittet av resultatene ble beregnet for å oppnå en mer pålitelig tilnærming.



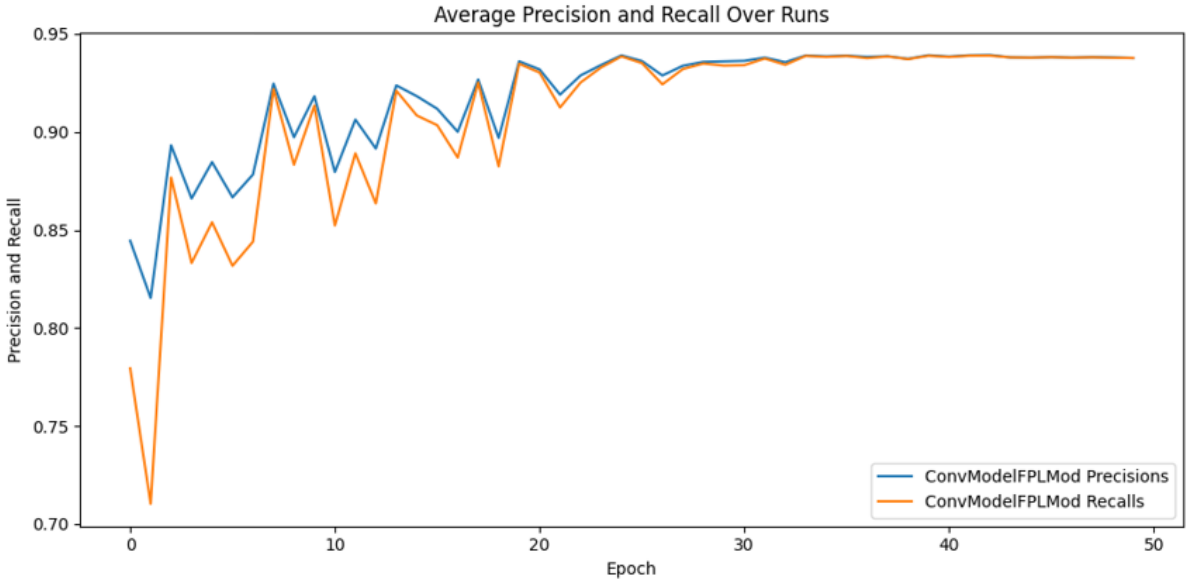
Figur 6-14 – Gjennomsnittlig tap av kjøringene til treningssettet (blå graf) og testsettet (oransje graf) for den modifiserte modellen.



Figur 6-15 – Gjennomsnittlig nøyaktighet av kjøringene til treningssettet (blå graf) og testsettet (oransje graf) for den modifiserte modellen.

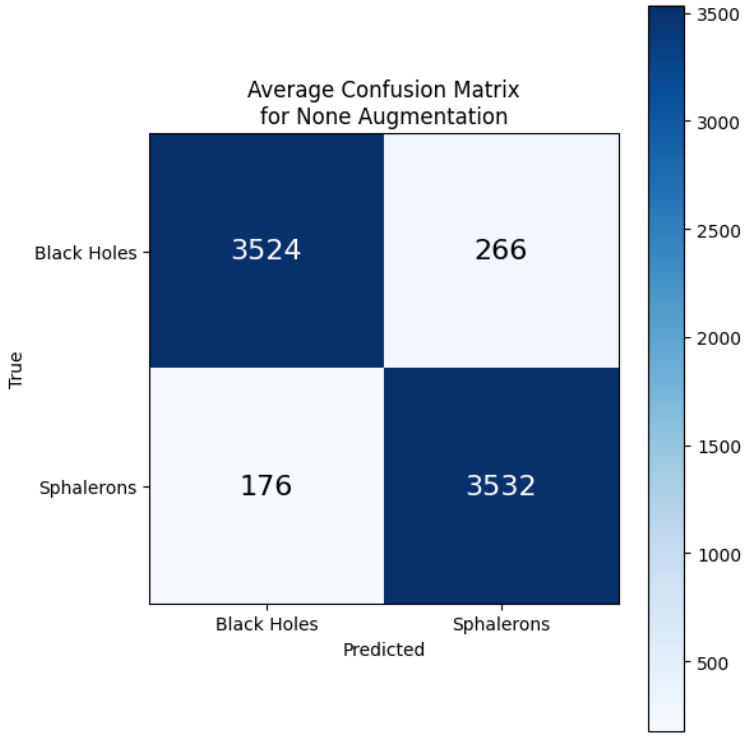


Figur 6-16 – Gjennomsnittlig nøyaktighet av mikroskopiske sorte hull (blå graf) og sfaleroner (oransje graf) for den modifiserte modellen.



Figur 6-17 – Gjennomsnittlig presisjon (blå graf) og sensitivitet «recall» (oransje graf) for den modifiserte modellen.

Forvirringsmatrisen gitt i Figur 6-18 gjelder for gjennomsnittet av kjøringene av ConvModelFPLMod.



Figur 6-18 – Forvirringsmatrise for ConvModelFPLMod som er tilpasset til symmetriegenskaper i det første reduksjonslaget, men som er uten data augmentering.

Tabell 6-5 og Tabell 6-6 viser gjennomsnittet av resultatene av fem kjøringer med 50 epoker av ConvModelFPLMod uten data augmentering samt deres standard avvik, evaluert med hensyn på henholdsvis nøyaktighet, sensitivitet og presisjon.

Tabell 6-5 – Resultatet etter tilpasninger av det første reduksjonslaget, evaluert med hensyn på total nøyaktighet av klassifiseringen.

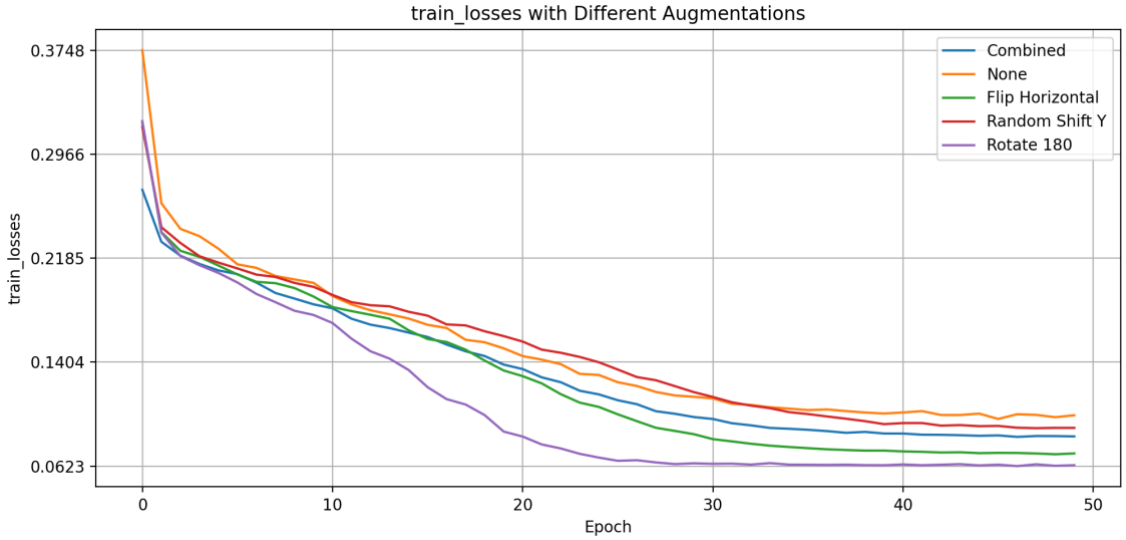
Modifisert CNN	**Tap Treningssett**	**Nøyaktighet Treningssett**	**Tap Testsett**	**Nøyaktighet Testsett**
uten augmentering	0,15 ± 0,06	92,59 ± 2,30	0,31 ± 0,16	90,74 ± 5,05

Tabell 6-6 – Resultatet etter tilpasninger av det første reduksjonslaget, evaluert med hensyn på sensitivitet, presisjon og nøyaktighet for hvert av fenomenene, på testsettet.

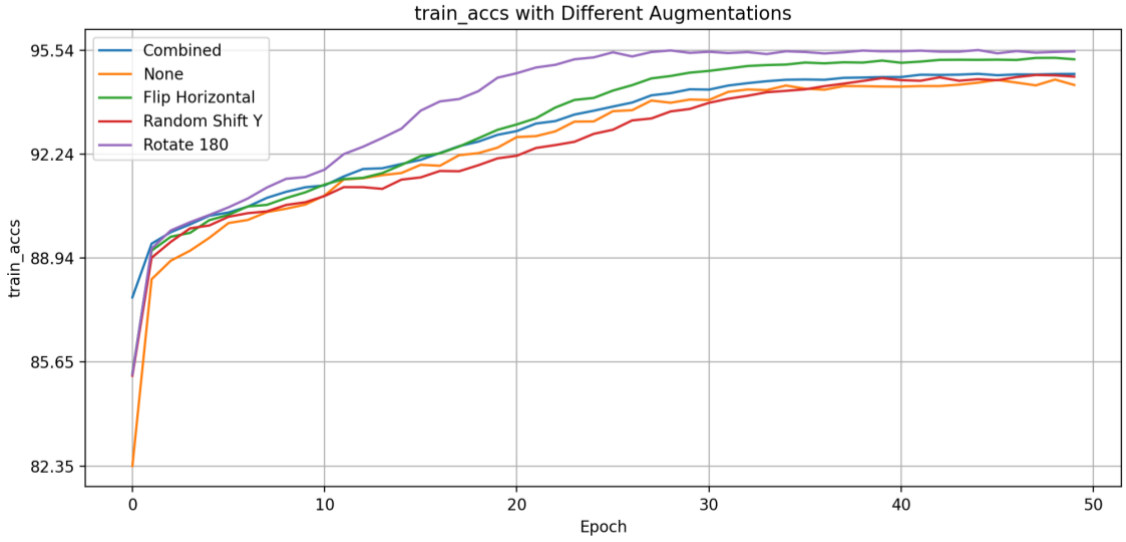
Modifisert CNN	**Sensitivitet Testsett**	**Presisjon Testsett**	**Nøyaktighet Mikroskopisk sort hull**	**Nøyaktighet Sfaleron**
uten augmentering	0,91 ± 0,05	0,92 ± 0,03	89,92 ± 9,73	91,54 ± 2,81

6.4 Modifisert CNN med data augmentering (ConvModelFPLMod)

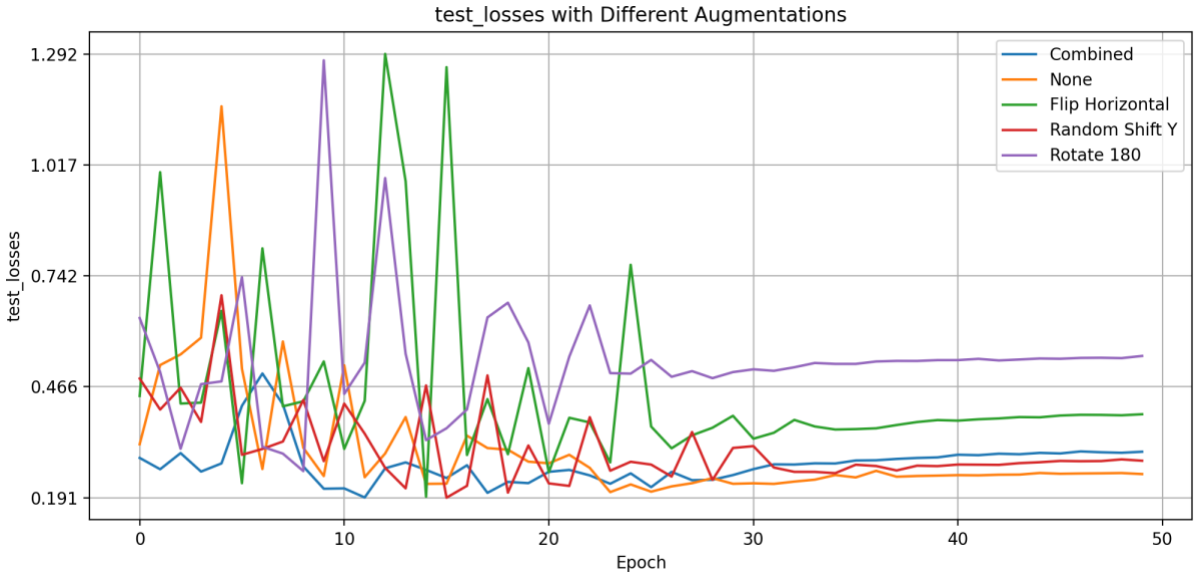
Nedenfor viser Figur 6-19 til Figur 6-22 resultatene av den modifiserte CNN-modellen, ConvModelFPLMod, der det har blitt brukt ulike teknikker for data augmentering på treningssettet. Som i de tidligere versjonene av CNN-modellen, blir de ulike teknikkene for data augmentering representert i figurene som grafer med egendefinerte farger. Modellen ble kjørt fem ganger med 50 epoker i hver gjennomkjøring, slik som i de tre foregående versjonene, og gjennomsnittet av resultatene ble beregnet.



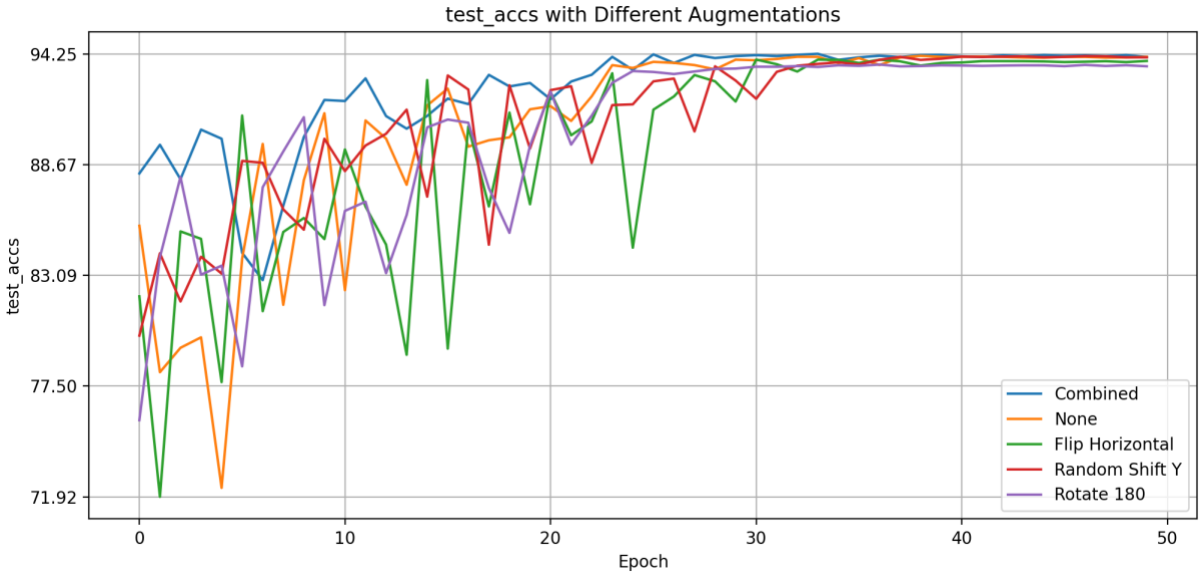
Figur 6-19 – Gjennomsnittlig tap av treningssettet til ConvModelFPLMod med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).



Figur 6-20 – Gjennomsnittlig nøyaktighet av treningssettet til ConvModelFPLMod med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).



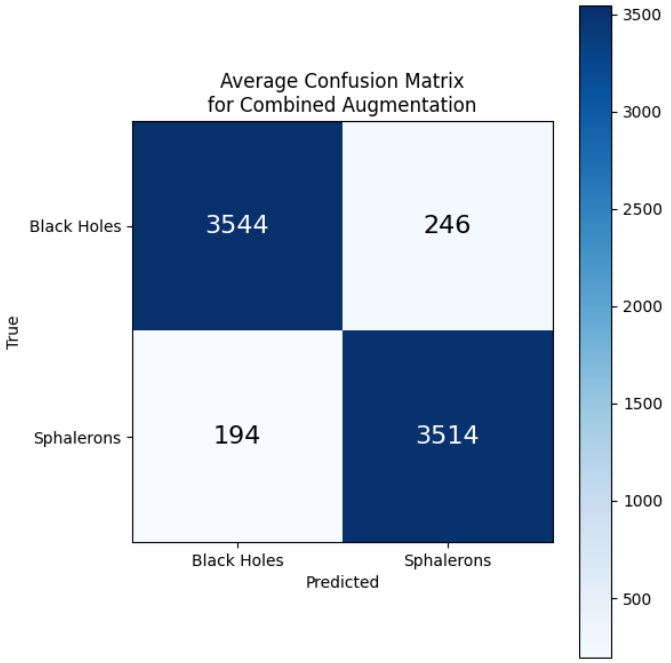
Figur 6-21 – Gjennomsnittlig tap av testsettet til ConvModelFPLMod med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).



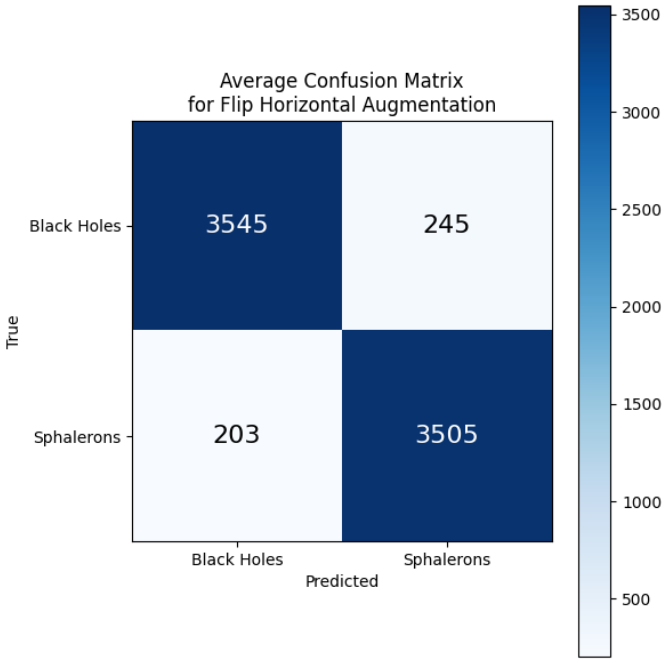
Figur 6-22 – Gjennomsnittlig nøyaktighet av testsettet til ConvModelFPLMod med ingen data augmentering (oransje graf), med ulike teknikker for data augmentering (grønn, rød og lilla graf) og alle teknikkene kombinert (blå graf).

Forvirringsmatrisene i Figur 6-23 til Figur 6-26 gjelder for gjennomsnittet av kjøringene av ConvModelFPLMod med de ulike teknikkene for data augmentering i treningssettet. Øverst til venstre er forvirringsmatrisen for modellen der alle teknikkene for data augmentering er kombinert. Øverst til høyre er forvirringsmatrisen for ConvModelFPLMod der speilvending om den horisontale aksen (x-aksen) er benyttet for å utnytte høyre-venstresymmetrien i

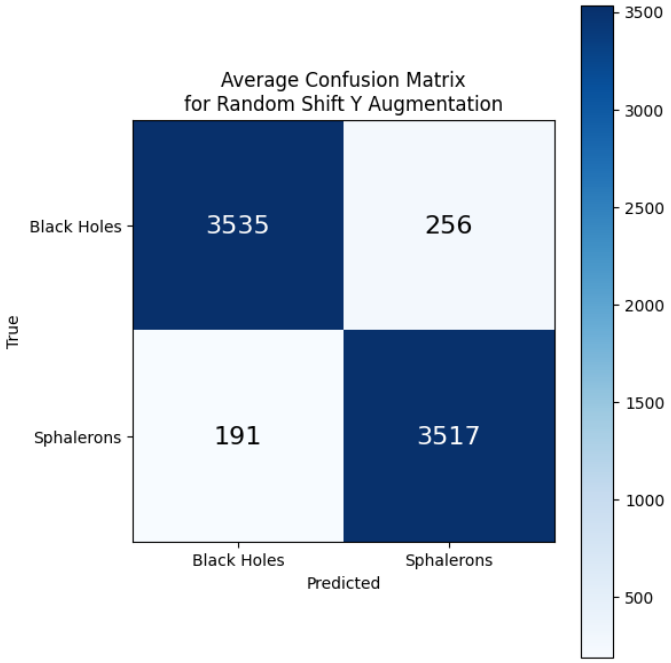
detektorbildene. Nederst til venstre er forvirringsmatrisen for modellen der tilfeldige forskyvninger langs den vertikale aksen (y-aksen) er brukt for å utnytte sylindersymmetrien i detektorbildene. Nederst til høyre er forvirringsmatrisen for ConvModelFPLMod der 180 graders rotasjon er brukt for å utnytte høyre-venstresymmetri.



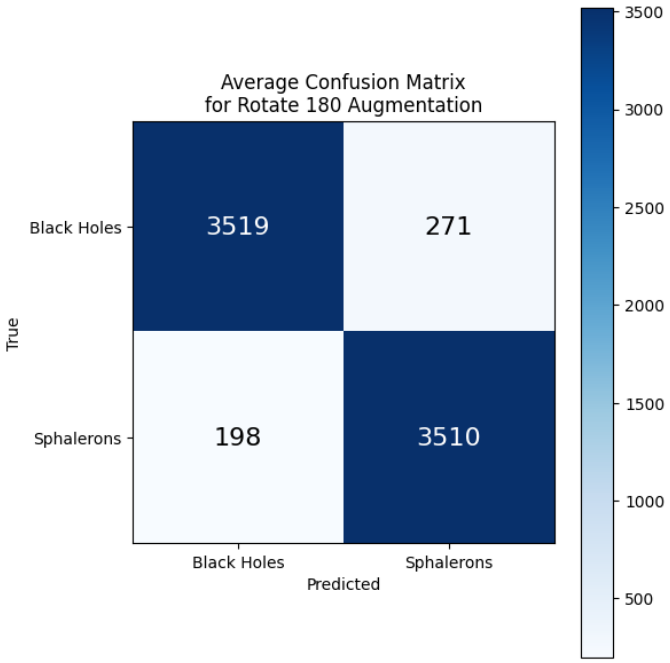
Figur 6-23 – Forvirringsmatrise for ConvModel med kombinasjon av alle teknikker for data augmentering.



Figur 6-24 – Forvirringsmatrise for ConvModel der horisontal flipp er brukt på treningssettet.



Figur 6-25 – Forvirringsmatrise for ConvModel der tilfeldig skift i y-aksen er brukt på treningssettet.



Figur 6-26 – Forvirringsmatrise for ConvModel der 180 graders rotasjon er brukt på treningssettet.

Tabell 6-7 til Tabell 6-8 viser gjennomsnittet av resultatene av de fem kjøringene med 50 epoker der data augmentering blir brukt i treningsdatasettet samt deres standard avvik. Modellen evalueres med alle teknikker for data augmentering kombinert, og med hver av teknikkene hver for seg.

Tabell 6-7 – Resultatene etter data augmentering er brukt, evaluert med hensyn på total nøyaktighet av klassifiseringen.

Modifisert CNN	**Tap Treningssett**	**Nøyaktighet Treningssett**	**Tap Testsett**	**Nøyaktighet Testsett**
Kombinert	0,13 ± 0,05	93,03 ± 1,82	0,28 ± 0,05	92,41 ± 2,61
Forskyvning på y-aksen (sylindersymmetri)	0,15 ± 0,05	92,59 ± 2,01	0,31 ± 0,09	90,97 ± 3,74
Speilvendt på x-aksen (Høyre-venstresymmetri)	0,13 ± 0,06	93,24 ± 2,23	0,46 ± 0,23	89,73 ± 5,30
180 graders rotasjon (Høyre-venstresymmetri)	0,11 ± 0,06	93,90 ± 2,29	0,53 ± 0,16	90,37 ± 4,47

Tabell 6-8 – Resultatene etter data augmentering er brukt, evaluert med hensyn på sensitivitet, presisjon og nøyaktighet for hvert av fenomenene, på testsettet.

Modifisert CNN	**Sensitivitet Testsett**	**Presisjon Testsett**	**Nøyaktighet Mikroskopisk sort hull**	**Nøyaktighet Sfaleron**
Kombinert	0,92 ± 0,03	0,93 ± 0,02	93,49 ± 4,36	91,35 ± 2,95
Forskyvning på y-aksen (sylindersymmetri)	0,91 ± 0,04	0,92 ± 0,02	90,02 ± 6,61	91,91 ± 2,21
Speilvendt på x-aksen (Høyre-venstresymmetri)	0,90± 0,05	0,91 ± 0,03	87,42 ± 10,51	91,99 ± 2,88
180 graders rotasjon (Høyre-venstresymmetri)	0,90 ± 0,05	0.92 ± 0,03	89,25 ± 8,51	91,47 ± 2,94

7 Diskusjon

Dette kapittelet tar for seg analysen av resultatene fra maskinlæringsmodellen i sammenheng med prosjektets teorifundament og tidligere forskning. Hensikten er å undersøke om det er en overensstemmelse mellom de avdekkede svarene og den tidligere presenterte teorien. Strukturen for diskusjonen baserer seg på de fire variasjonene av CNN-modellen beskrevet i alternativ løsning 1. I denne sammenhengen refererer enkel modell til en CNN-modell uten modifiseringer i arkitekturen, mens modifisert modell refererer til en CNN-modell der det første reduksjonslaget er tilpasset til symmetriegenskapene i detektorbildene. Sammenligningen vil foregå mellom enkel og modifisert modell, som er uten og med data augmentering. Videre vil diskusjonen omfatte en vurdering av eventuell skjevhet i som kan oppstå i klassifiseringen som følge av bruk av data augmentering. Diskusjonskapittelet spiller en avgjørende rolle i å konkludere på prosjektets problemstilling og overordnede mål.

7.1 Enkel CNN med data augmentering

I denne seksjonen vil resultatene fra den enkle modellen med data augmentering bli diskutert. Sylindersymmetrien og høyre-venstresymmetriene i detektorbildene er utnyttet for data augmentering.

Hva gjorde at de spesifikke teknikkene for data augmentering ble ansett som passende for å øke mengden treningsdata?

Flere eksperimenter ble utført for å evaluere effekten av ulike teknikker for data augmentering, både hver for seg og kombinert. Basert på at sylindersymmetrien gir en symmetrisk struktur langs den vertikale aksen (y-aksen) i bildene, ble det antatt at valget av utsnitt langs denne aksen ikke hadde noen betydning for modellens ytelse. Derfor kunne vertikal forskyvning som teknikk benyttes for å utnytte sylindersymmetrien i bildene. Kollisjonsretningen og partiklenes energi ved partikkelsammenstøtene gjorde det også like sannsynlig at den speilvendte versjonen av bildene kunne være til stede i treningssettet. Derfor var det mulig å utnytte høyre-venstresymmetrien ved å inkludere den speilvendte versjonen av bildene i treningssettet. Speilvendingen ble utført ved å bruke både horisontal speiling og 180 graders rotasjon.

Forventet resultat:

Ved å implementere teknikker for data augmentering basert på symmetriegenskapene i bildene, forventes det at modellen blir mer robust, får økt generaliseringsevne, og at eventuell overtilpasning blir redusert. Ved å øke mengden treningsdata gjennom teknikker som vertikal forskyvning, horisontal speiling og rotasjon, vil modellen bli eksponert for variasjon i treningsdataene. Dermed antas det at modellen vil bli mer robust mot variasjoner i inndataene og få økt generaliseringsevne som et resultat av eksponering for et bredere spekter av treningsdata. Det er også en forventning om at overtilpasning i modellen reduseres gjennom data augmentering, ved å oppfordre modellen til å lære generelle trekk i treningsdataene fremfor å memorere spesifikke detaljer.

Faktiske resultater:

Bruken av ulike teknikker for data augmentering resulterte i varierte resultater med hensyn til nøyaktighet, sensitivitet og presisjon i CNN-modellen. Ved å kombinere alle teknikkene for data augmentering økte gjennomsnittlig nøyaktighet på testsettet fra 89,63 ± 0,04 i basistilfellet til 91,42 ± 4,25. Sensitiviteten til testsettet økte også, fra 0,90 ± 0,05 i basistilfellet til 0,91 ± 0,04 i den enkle modellen med data augmentering, for alle kombinasjoner av data augmentering. Forbedringen i sensitiviteten kan indikere at modellen hadde bedre prediksjonskapasitet sammenlignet med basistilfellet. Årsaken for dette kan være at modellen fikk mer data å lære fra, hvilket bidro til å øke mengden mønstre den kunne dra nytte av for å forbedre klassifiseringen. Gjennomsnittlig presisjon i modellen, målt til 0,91 ± 0,03 i basistilfellet, ble også forbedret da alle teknikker for data augmentering ble kombinert. Presisjonen økte til 0,92 ± 0,03. Forbedringene kan skyldes at modellen dro nytte av variasjonene og utvidelsen av treningsdataene gjennom data augmentering. Forbedringene kan også skyldes at alle teknikkene for data augmentering ble brukt samtidig, hvilket introduserer enda flere typer variasjoner og forvrengninger i treningsdataen enn det en enkelt teknikk kunne oppnådd.

Bruken av teknikken med forskyvning langs den vertikale aksen (y-aksen) resulterte i en marginal forbedring i gjennomsnittlig nøyaktighet sammenlignet med basistilfellet. Nøyaktigheten økte fra 89,63 ± 0,04 i basistilfellet til 90,47 ± 5,40 med denne teknikken for data augmentering. Forbedringen indikerer at utnyttelsen av denne spesifikke teknikken kan være gunstig for klassifiseringen av mikroskopiske sorte hull og sfaleroner. Den marginale forbedringen i nøyaktighet kan skyldes utnyttelse av sylindersymmetrien i bildene.

Sylindersymmetrien kan tillate en mer effektiv generalisering av modellen, da den kan dra nytte av likhetene mellom ulike regioner i bildene som skyldes den symmetriske strukturen. Dette kan bidra til å forbedre modellens evne til å gjenkjenne og klassifisere objekter i bildene. Det er imidlertid viktig å merke seg at nøyaktigheten for mikroskopiske sorte hull viste en nedgang i forhold til basistilfellet og den totale nøyaktigheten for hele testsettet. Den gjennomsnittlige nøyaktigheten for mikroskopiske sorte hull ble målt til 88,30 ± 11,71. Nedgangen i nøyaktighet kan skyldes kompleksiteten og variasjonen i bildene til de mikroskopiske sorte hullene, som kan gjøre dem vanskeligere å gjenkjenne og klassifisere nøyaktig. Selv om den generelle nøyaktigheten ble forbedret med denne teknikken, kan det være behov for ytterligere tilpasninger eller alternative tilnærminger for å øke nøyaktigheten spesifikt for mikroskopiske sorte hull. Standardavviket for denne nøyaktigheten var også vesentlig større enn for de andre målingene. Hvis standardavviket for en måling er stort, vitner det om en betydelig variasjon eller usikkerhet i modellens ytelse når den predikerer klassen for mikroskopisk sort hull. Sensitiviteten, derimot, forble uendret fra basistilfellet noe som indikerer at modellen fortsatt er like effektiv til å identifisere positive tilfeller i begge klassene. På den annen side kan det ha vært en økning i antall feil i identifiseringen av negative tilfeller i den klassen hvor vi ser en reduksjon i spesifikk nøyaktighet. Dette kan være et resultat av at modellen prioriterer sensitivitet fremfor spesifisitet i sine klassifiseringsbeslutninger.

Med utgangspunkt i et basistilfelle, der den gjennomsnittlige presisjonen var målt til 0,91 med et standardavvik på 0,03, ble det sett en forbedring til 0,92 med et tilsvarende standardavvik ved bruk av kombinerte data augmneteringsteknikker. Det bemerkes et unntak med bruk av en spesifikk teknikk, nemlig speilvending langs x-aksen. Når speilvending langs x-aksen ble benyttet, ble det sett en nedgang i modellens gjennomsnittlige nøyaktighet til 89,59, med et økt standardavvik på 5,81, i forhold til basistilfellet. Dette kan indikere enten lavere presisjon, større usikkerhet, eller begge deler. Det tyder på en potensiell negativ innvirkning på modellens ytelse ved bruk av denne formen for data augmentering. Til tross for nedgangen i nøyaktighet, forble modellens sensitivitet, evnen til korrekt å identifisere positive tilfeller, omtrent uendret på 0,90, men med et litt høyere standardavvik på 0,06. Ytterligere analyse av nøyaktighet for spesifikke klasse av bilder, nemlig bilder av mikroskopiske sorte hull, viste også en reduksjon til 87,81 ± 7,13, fra 87,45 ± 7,31 i basistilfellet. Dette tyder på at bruk av speilvending langs x-aksen kan forverre modellens evne til korrekt å klassifisere slike bilder, selv om den generelle sensitiviteten er bevart. Årsaken til denne reduksjonen i ytelse kan være at høyre-venstre-symmetrien i partikkelkollisjonsbilder kanskje ikke er så fremtredende som antatt. Selv om det

generelt er like sannsynlig at partiklene vil forekomme på hver side av kollisjonsaksen, kan innføring av speilvending langs x-aksen skape en form for 'forvirring' for modellen, noe som fører til lavere nøyaktighet.

Data augmenteringsteknikken med 180 graders rotasjon, som utnytter høyre-venstre-symmetrien, resulterte i en betydelig forbedring i gjennomsnittlig nøyaktighet til 90,21 ± 4,85, sammenlignet med basistilfellet. Dette indikerer en positiv innvirkning på modellens ytelse ved bruk av denne formen for augmentering. Samtidig ble det observert en konstant sensitivitet på 0,90 ± 0,05, likt basistilfellet, noe som indikerer en vedvarende evne til korrekt identifikasjon av positive tilfeller. En merkbar forbedring ble også notert i nøyaktigheten for mikroskopiske sorte hull ved bruk av 180 graders rotasjonsteknikken. Her økte den gjennomsnittlige nøyaktigheten til 89,62, med ± 8,30, en økning fra basistilfellets 87,45 ± 7,31. Dette antyder en potensiell effektivitet av 180 graders rotasjonsteknikken for å forbedre modellens evne til å klassifisere bilder av mikroskopiske sorte hull, muligens grunnet utnyttelse av den iboende høyre-venstre-symmetrien i slike bilder. Resultatene i sin helhet indikerer en variabel effekt av data augmneteringsteknikker på nøyaktighet og sensitivitet innen en CNN-modell. Mens noen teknikker kan forbedre disse parameterne, kan andre potensielt ha en mindre innvirkning, eller i noen tilfeller til og med redusere ytelsen. Dermed understrekes viktigheten av en nøye vurdering av valgte teknikker i forhold til det spesifikke bruksområdet.

Tolkning av grafiske funn:

Det var også forventet at bruken av data augmentering ville redusere risikoen for overtilpasning til treningsdataen. Dette kan undersøkes ved å se på både nøyaktighet og tapet gjennom treningen (se Figur 6-6 til Figur 6-8). Observasjonene viser at både trening og test nådde et punkt der modellen ikke ble bedre etter epoke 30, og dette nivået holdt seg stabilt etter epoke 40. Dette kan skyldes flere faktorer, inkludert bruken av vekt fall for optimalisering og en planlegger som justerer læringshastigheten når tapet ikke forbedrer seg over et fast antall epoker.

Ved å se på utsnittene av de ulike grafene, kan man observere at resultatene flater ut på et relativt stabilt nivå med lite variasjon. Dette indikerer at modellen ikke er overtilpasset til treningsdataen, da det hadde vært forventet at resultatene ville blitt verre over tid hvis overtilpasning oppstod. En mulig årsak til dette kan være bruken av data augmentering. En

annen faktor er bruken av en planlegger i treningsprosessen, der modellen sammenligner gjeldende tap med forrige tap og justerer læringshastigheten hvis tapet forverres. Dette bidrar til å unngå overtilpasning og opprettholder stabile resultater etter epoke 40.

Det er imidlertid viktig å merke seg at dersom modellen hadde blitt kjørt over en enda lenger periode eller med flere epoker per gjennomkjøring, kunne det ha oppstått overtilpasning og bedre læring av treningssettet. Dermed er det en mulighet for ytterligere forbedringer hvis modellen hadde blitt trent lenger.

Eventuelle begrensninger og/eller utfordringer ved å bruke data augmentering:
Selv om data augmentering er nyttig for å forbedre ytelsen til maskinlæringsmodeller, er det viktig å være oppmerksom på noen begrensninger og utfordringer. For det første kan det føre til økt beregningskostnad på grunn av den økte datamengden som må behandles. Videre er ikke alle teknikker for augmentering relevante eller effektive for alle datasett, og det er nødvendig å tilpasse teknikkene til den aktuelle oppgaven. Data augmentering kan også introdusere utfordringer knyttet til generalisering av modellen og potensiell overtilpasning til treningssettet. Noen teknikker for data augmentering kan ha begrenset eller ingen effekt på ytelsen, og det er nødvendig å nøye evaluere og validere resultatene. Til slutt, selv om data augmentering kan øke variasjonen, er det fortsatt behov for et stort og mangfoldig datasett for å trene modellen på en pålitelig måte. Ved å nøye håndtere disse begrensningene og utfordringene kan data augmentering bidra til bedre resultater i maskinlæring og bildeanalyse.

Tidligere forskning og sammenligning:
Resultatene i studien *«Effect of data-augmentation on fine-tuned CNN model performance»* av Ramprasad, Raina og Mondal (2020) i punkt 3.5, som undersøkte effekten av data augmentering på ytelsen til CNN-modeller, kan sammenlignes med funnene i denne analysen. Studien demonstrerte at bruk av teknikker for data augmentering, som eksempelvis fast rotasjon, horisontal flipp og skalering hadde en betydelig positiv innvirkning på ytelsen til både VGG16- og ResNet50-modellene.

Forskerne oppnådde betydelige forbedringer i ytelsen til de finjusterte modellene ved å bruke data augmentering. Et eksempel på en forbedring er at den finjusterte VGG16-modellen nådde treningsnøyaktighet på 93,5% og en valideringsnøyaktighet på 91,5% med data augmentering.

Tilsvarende oppnådde den finjusterte ResNet50-modellen enda bedre nøyaktighet, med 95% på trening og 93% på validering. Videre ble det observert betydelige forbedringer i testnøyaktighet, med 88% for VGG16 og 90% for ResNet50 når data augmentering ble brukt. I motsetning til dette oppnådde modellene bare testnøyaktigheter på henholdsvis 80% og 82% uten data augmentering. Dette indikerer at data augmentering reduserte overtilpasning og forbedret modellens nøyaktighet, noe som resulterte i en mer vellykket bildeklassifiseringsoppgave.

I resultatene observert i dette prosjektet ble det også observert forbedringer i ytelsen til CNN-modellen. For eksempel økte nøyaktigheten på treningssettet fra basistilfellet til 91,42% når alle teknikkene ble kombinert. Disse funnene stemmer overens med resultatene i den nevnte studien, der VGG16-modellen oppnådde en treningsnøyaktighet på 93,5% med data augmentering.

Det ble også observert en marginal forbedring i nøyaktigheten ved bruk av forskyvning på y-aksen, en teknikk som utnyttet sylindersymmetrien i bildene. Dette samsvarer med funnene i studien, der både VGG16- og ResNet50-modellene oppnådde høyere nøyaktigheter når data augmentering ble brukt.

Det er viktig å merke seg at resultatene i den aktuelle undersøkelsen kan variere fra studien utført av Ramprasad, Raina og Mondal (2020) på grunn av forskjeller i datasett, modellarkitektur og implementeringsdetaljer. Testnøyaktighetene for mikroskopiske sorte hull og sfaleroner var sammenlignbare eller til og med litt høyere enn de rapporterte resultatene i studien.

7.2 Modifisert CNN uten data augmentering

I denne seksjonen vil resultatene fra den modifiserte modellen uten data augmentering bli diskutert. Tilpasningen av reduksjonslaget har blitt utført for å ta hensyn til sylindersymmetrien, og høyre-venstresymmetrien i detektorbildene.

Hvorfor ble det valgt å tilpasse reduksjonslaget på denne måten?

Arkitekturen til den modifiserte CNN-modellen er designet for å inkorporere kunnskap om symmetriegenskapene i detektorbildene ved å tilpasse det første reduksjonslaget som tar

gjennomsnittet av høyre-venstre, rotasjons- og sylindrisk symmetrier. Denne tilnærmingen ble valgt fordi reduksjonslaget kan bidra til å redusere den romlige størrelsen, noe som dermed senker bygningsmessig kompleksitet og forhindrer overjustering. 180 graders rotasjon ble valgt fremfor 90 og 270 grader for å opprettholde vertikal symmetri. Andre metoder for tilpasning av reduksjonslaget ble unngått for å hindre tap av data eller forvrengning av de originale bildene.

Forventet resultat:

Hvis symmetriene er til stede i bildene og den modifiserte CNN tar hensyn til dem, burde modellens evne til å predikere mellom mikroskopiske sorte hull og sfaleroner øke. Ikke bare forventes at modellen skal være bedre til å klassifisere, men også at den flater ut raskere siden den ikke trenger å lære symmetriene fra bunnen av.

Faktiske resultater:

Ved å sammenligne en enkel og en modifisert konvolusjonell nevrale nettverksmodell, kommer frem det interessante resultater. Den enkle modellen hadde et gjennomsnittlig treningstap på 0,13 ± 0,07, og en gjennomsnittlig treningsnøyaktighet på 93,00 ± 2,56. Når det gjelder testsettet, var gjennomsnittlig tap 0,43 ± 0,20, og gjennomsnittlig nøyaktighet var 89,63 ± 4,90. Modellens presisjon og sensitivitet på testsettet var henholdsvis 0,91 og 0,90.

Den modifiserte modellen, på den annen side, hadde et noe høyere gjennomsnittlig treningstap på 0,15, men et lavere standardavvik på 0,06. Treningsnøyaktigheten var litt lavere enn den enkle modellen på 92,59, men igjen hadde den et lavere standardavvik på 2,30. Når det gjelder testsettet, så ble det en forbedring på tapet som falt til 0,31, og nøyaktigheten økte til 90,74. Denne modellen oppnådde også en liten forbedring i presisjon og sensitivitet på testsettet, med henholdsvis 0,92 og 0,91.

Til tross for litt høyere treningstap, viste den modifiserte modellen bedre resultater på testsettet både i form av tap og nøyaktighet, så vel som presisjon og sensitivitet. Dette tyder på at tilpasning av første reduksjonslaget i den modifiserte modellen, designet for å anerkjenne symmetrier som høyre-venstre, sylinder-symmetri og rotasjonssymmetri, kan bidra til å forbedre ytelsen til modellen, selv om økningen ikke er stor.

Tolkning av grafiske funn:

Basert på grafen til enkel CNN-modell Figur 6-1, vises det at modellen har en tendens til overtilpasning rundt epoke 15-20, noe som kan observeres gjennom tapet. Men etter omtrent 30 epoker ser det ut til at tapet for både trenings- og testsettet stabiliserer seg og endrer seg lite. Dette skjer fordi det ble benyttet en planlegger under treningen som justerer lærehastigheten. Denne reduserer steglengden til gradientoppdateringene, slik at modellen ikke presterer dårligere over tid.

Når en ser på grafen for den modifiserte CNN-modellen Figur 6-14, ble det oppdaget lignende resultater som for den enkle modellen. Det er imidlertid en markant forskjell: kurvene for trening og testing ligger nærmere hverandre, og det er mindre forskjell i tap mellom trenings- og testsett. Dette indikerer at modellen kan være bedre til å lære spesifikke mønstre, noe som tyder på at modifiseringen av modellen kan ha forbedret dens evne til å generalisere fra treningsdataene til nye, ukjente data.

Tolkning av forvirringsmatriser:

Resultatene av forvirringsmatrisen fra basistilfellet sammenlignet med forvirringsmatrisen for de ulike variasjonene av data augmentering, viser en forbedring i antall riktige prediksjoner for både sanne positive tilfeller (TP) og for sanne negative tilfeller (TN). Siden sensitivitet måler evnen til modellen til å identifisere positive tilfeller korrekt, vil den også bli påvirket positivt, da økningen i TP fører til en høyere andel av de faktiske positive tilfellene som blir riktig identifisert (ref. formel for sensitivitet i 2.4.6). Det er derimot viktig å merke seg at sensitiviteten ikke bare avhenger av TP, men også av FN. Hvis antall feil negative prediksjoner (FN) reduseres, vil sensitiviteten øke. Så selv om det er en forbedring i både TP og TN, er det viktig å vurdere om sensitiviteten har økt, og om modellen fortsatt er i stand til å gjenkjenne og fange opp de positive tilfellene på en pålitelig måte.

Eventuelle begrensninger og/eller utfordringer ved å tilpasse reduksjonslaget:

Integrering av et reduksjonslag i en konvolusjonell nevrale nettverksmodell (CNN), tilpasset til symmetri-egenskapene i bildene, representerer en nyskapende tilnærming som kan løfte modellens generelle ytelse. Likevel er det viktig å være bevisst på en rekke begrensninger og utfordringer knyttet til dette.

Økt kompleksitet er en vesentlig bekymring, da tilpasningen kan gjøre modellstrukturen mer kompleks og øke de nødvendige ressursene for trening. Dette kan potensielt forlenge treningsprosessen, noe som kan skape problemer i tilfeller der rask modellutvikling er ønskelig, eller når det jobbes med store datasett.

På samme tid er det utfordrende å nøyaktig identifisere og implementere symmetri-egenskapene i et bilde. Dette kan være spesielt vanskelig med komplekse eller støyende bilder, hvor eventuelle feil i deteksjonen av symmetri kan resultere i klassifiseringsfeil.

Selv om det er intuitivt å dra nytte av symmetri-egenskapene i bildene, gir det ingen absolutt garanti for forbedring i modellens ytelse. Resultatene er sterkt avhengig av det spesifikke datasettet og den aktuelle oppgaven som skal løses.

Tilpasning av et reduksjonslaget til symmetri-egenskapene er heller ikke en universell løsning for alle bildeklassifiseringsoppgaver, ettersom ikke alle bilder eller datasett vil nødvendigvis inneholde nyttige symmetri-egenskaper.

Selv om det finnes mange andre utfordringer som kan tas i betraktning, understreker disse omstendighetene viktigheten av å nøye evaluere nytteverdien og gjennomførbarheten ved å tilpasse et reduksjonslag til symmetriegenskapene for hver spesifikk oppgave. Til tross for de tidligere nevnte utfordringene, indikerer resultatene fra sammenligningen mellom den grunnleggende og den modifiserte konvolusjonelle nevrale nettverksmodellen at det er muligheter for å forbedre modellens ytelse ved å justere det første reduksjonslaget for å gjenkjenne symmetrier i bildedataene.

Tidligere forskning og sammenligning:
Sammenligningen av dette prosjektet med arbeidet utført av Sander Dieleman i studien *«Exploiting Cyclic Symmetry in Convolutional Neural Networks»* i punkt 3.5, avdekker interessante sammenhenger og kontraster.

I denne studien ble symmetri-egenskapene i bildene utnyttet for å forbedre ytelsen til den konvolusjonelle nevrale nettverksmodellen (CNN). Et tilpasset reduksjonslag ble innlemmet i CNN-modellen, som tok hensyn til høyre-venstre, rotasjons- og sylindrisk symmetrier i bildene. Dette bidro til å redusere den romlige størrelsen på inndata dataene, noe som senket

beregningsmessig kompleksitet og forhindret overjustering. Det ble observert forbedringer i resultater på testsettet både i form av tap og nøyaktighet, så vel som presisjon og sensitivitet, selv om økningen ikke var stor.

Dieleman og kollegaene hans brukte en lignende tilnærming ved å innføre et lag med rotasjonssymmetri i nettverksarkitekturene sine, og oppnådde betydelige forbedringer. For eksempel reduserte de antall parametere fra 2.84M til 0.95M i plankton datasettet, mens nøyaktigheten forble relativt stabil. Dette demonstrerer effektiviteten av deres metode og støtter funnene fra denne studien om at utnyttelse av symmetri-egenskapene i bildene kan forbedre ytelsen til CNN-modeller.

Imidlertid, mens Dieleman og hans gruppe oppnådde en betydelig reduksjon i antall parametere samtidig som de opprettholdt en relativt stabil nøyaktighet, ble det observert en liten forbedring i presisjon og sensitivitet i denne studien, men ikke en like betydelig reduksjon i antall parametere. Dette kan skyldes forskjeller i datasettene, oppgavene som skal løses, eller de spesifikke tilpasningene som er gjort i reduksjonslagene i de to studiene.

Samlet sett gir disse funnene en indikasjon på at det er verdt å utforske videre bruk av symmetri-egenskapene i bilder for å forbedre ytelsen til konvolusjonelle nevrale nettverksmodeller.

7.3 Modifisert CNN med data augmentering

I denne seksjonen blir resultatene fra en modifisert CNN-modell diskutert, der symmetriegenskapene ble utnyttet for data augmentering og det første reduksjonslaget ble tilpasset for å dra nytte av disse symmetriegenskapene.

Hvorfor kombinere de to justeringene som gjøres i og for maskinlæringsmodellen?
Det kan være interessant å undersøke om en kombinasjon av forskjellige justeringer i og for en maskinlæringsmodell kan føre til økt modellytelse. Ved å kombinere den modifiserte modellen med bruken av data augmentering i treningssettet, kan man overvinne begrensninger som enkeltstående justeringer kan ha. En enkelt justering kan ha begrensede effekter eller ikke være tilstrekkelig for å løse bestemte problemer. Ved å kombinere de ulike justeringene kan man dra hver enkelt teknikk sine styrker og overvinne deres begrensninger. Dette kan føre til bedre

resultater og mer effektive løsninger. Til slutt kan kombinering av teknikker føre til uventede samspill og forbedringer. Noen ganger oppdager man at kombinasjonen av forskjellige teknikker resulterer i bedre resultater enn hver teknikk individuelt. Dette kan åpne for nye tilnærminger eller metoder som ikke ville vært mulig å oppnå med enkeltjusteringer eller modifikasjoner alene.

Forventet resultat:

En potensiell forventning er en økning i modellens nøyaktighet. Kombinasjonen av data augmentering, og tilpasning av reduksjonslaget kan resultere i en forbedring av modellens nøyaktighet på testsettet. Ved å forbedre modellens evne til å generalisere og redusere overtilpasning, samtidig som man tilpasser reduksjonslaget for bedre håndtering av bildeskala, kan man oppnå bedre resultater i form av høyere nøyaktighet i klassifiseringen. Dette skyldes at modellen lærer mer robuste og generaliserbare trekk samtidig som den optimaliseres for å håndtere variasjon og viktige detaljer i bildene.

Faktiske resultater:

Basistilfellet hadde en målt nøyaktighet på 89,63 ± 4,90. Alle teknikkene for data augmentering som ble brukt i den modifiserte modellen viste en forbedring fra basistilfellet, der det beste resultatet kom fra den modifiserte modellen som kombinerte alle teknikkene. Da ble den gjennomsnittlige nøyaktighetet forbedret til 92,41 ± 2,61. Sensitiviteten ble også forbedret i den modifiserte modellen med data augmentering sammenlignet med basistilfellet. Basistilfellet målte sensitiviteten til 0,90 ± 0,05 og den modifiserte modellen med data augmentering målte sensitiviteten til 0,92 ± 0,03 for det beste resultatet.

En enkel CNN-modell uten tilpasning av det første reduksjonslaget, men med data augmentering, viste en nøyaktighet som varierte fra 89,59 ± 5,81 til 91,42 ± 4,25. Dette indikerer at data augmenteringen alene kunne bidra til en viss økning i modellens ytelse i forhold til basistilfellet. Imidlertid viser resultatene fra den modifiserte modellen med data augmentering at kombinasjonen av begge justeringene ga enda bedre gjennomsnittlig nøyaktighet, som varierte fra 89,73 ± 5,30 til 92,41 ± 2,61.

Den modifiserte modellen, som hadde tilpasning av det første reduksjonslaget, men uten data augmentering, oppnådde en gjennomsnittlig nøyaktighet på 90,74 ± 5,05. Dette viser at tilpasning av symmetriegenskaper kunne gi en viss forbedring i modellens ytelse sammenlignet

med basistilfellet. Det var derimot kombinasjonen av den modifiserte modellen med data augmentering som ga den beste gjennomsnittlige nøyaktigheten, på 92,41 ± 2,61.

Den beste gjennomsnittlige nøyaktigheten ble oppnådd av den modifiserte modellen med data augmentering, med en variasjon fra 89,73 ± 5,30 til den beste nøyaktigheten på 92,41 ± 2,61. Den nest beste målingen av gjennomsnittlig nøyaktighet var på 91,42 ± 4,25, og ble observert fra den enkle modellen med data augmentering. Ved å studere standardavviket mellom de to beste resultatene, kan man observere at enkel modell med data augmentering har et vesentlig større standardavvik enn den beste modellen som var den modifiserte modellen med data augmentering. Dette tyder på at den enkle modellen var mer variabel i ytelsen, og resultatene varierte mer fra ett forsøk til et annet. På den annen side var den modifiserte modellen med data augmentering mer stabil og leverte mer konsistente resultater med mindre variasjon mellom forsøkene.

Tolkning av grafiske funn:
Visuelt sett fremstår det nesten ingen forskjell mellom den enkle modellen og den modifiserte modellen som anvender diverse teknikker for data augmentering. Den subtile visuelle forskjellen, uansett hvor liten den måtte være, underbygger viktigheten av detaljert kvantitativ analyse for å avsløre de faktiske forbedringene oppnådd gjennom disse modifiseringene.

Med det i betraktning, blir sammenligningen av resultatene mest meningsfull ved å utforske de tabulære dataene. Det er gjennom dette mer dyptgående undersøkelseslinsen at de faktiske fordelene og forbedringene som er oppnådd gjennom modifiseringer og data augmentering kan evalueres nøyaktig. Denne type analyse bidrar til å identifisere forbedringer, selv de som er nesten umerkelige visuelt, men som kan ha betydelig innvirkning på den totale ytelsen til modellene.

Tolkning av forvirringsmatriser:
Ved å analysere forvirringsmatrisene fra det grunnleggende CNN-tilfellet og det modifiserte CNN-tilfellet ved bruk av teknikker for data augmentering, observeres en forbedring i antall korrekte prediksjoner både for sanne positive (TP) og sanne negative (TN) tilfeller. Dette sammenlignes med forvirringsmatrisene for de ulike variantene av dataaugmentering.

Ettersom sensitivitet er en måling av modellens evne til å identifisere positive tilfeller korrekt, vil en økning i TP resultere i en positiv påvirkning, da en større andel av de faktiske positive tilfellene blir riktig identifisert, som beskrevet i formelen for sensitivitet i avsnitt 2.4.6.

Det er imidlertid viktig å understreke at sensitivitet ikke bare er avhengig av TP, men også av antall feil negative prediksjoner (FN). En reduksjon i FN vil føre til en økning i sensitiviteten. Selv om det er en forbedring i både TP og TN, er det derfor viktig å evaluere om sensitiviteten faktisk har økt og om modellen fortsatt effektivt kan identifisere og registrere positive tilfeller pålitelig.

Eventuelle begrensninger og/eller utfordringer ved bruk av data augmentering og tilpasning av første pooling laget samtidig:
Selv om tilpasning av det første reduksjonslaget i et CNN og samtidig bruk av data augmentering kan forbedre modellens ytelse betydelig, er det også viktig å bemerke potensielle utfordringer og begrensninger ved denne tilnærmingen, spesielt i forhold til symmetrier.

En utfordring kan være den økte treningskompleksiteten. Modifikasjoner til det første reduksjonslaget i et CNN kan kreve betydelig mer prosesseringskraft og minne, noe som kan forsinke treningsprosessen og kreve mer avansert maskinvare. Dette kan være en begrensning for noen organisasjoner eller forskningsinstitutter med begrensede ressurser.

En annen utfordring er knyttet til risikoen for overtilpasning til de spesifikke symmetriene i treningsdataene. Hvis treningsdatasettet ikke er representativt for det generelle problemområdet eller den underliggende populasjonen, kan modellen utvikle bias eller overfitting til spesifikke symmetrier i treningsdataene. Dette kan resultere i redusert generaliseringsytelse på nye data som ikke deler de samme symmetriene.

Data augmentering kan også komme med sine egne utfordringer. Selv om data augmentering kan bidra til å øke variasjonen i treningsdataene og forbedre modellens evne til å generalisere, kan det være utfordrende å velge riktig type og grad av data augmentering. For mye augmentering kan føre til at modellen lærer falske mønstre, mens for lite kan ikke gi tilstrekkelig regularisering for å forhindre overfitting. Dessuten kan enkelte augmneteringsteknikker forvrenge de naturlige symmetriene i dataene på måter som er

usannsynlige i virkelige situasjoner, noe som kan lede modellen til å lære irrelevante eller feilaktige mønstre.

Det er også viktig å merke seg at både tilpasning av det første reduksjonslaget og data augmentering krever en dypere forståelse av de underliggende dataene og problemområdet. Uten dette kan feilaktige antagelser gjøres, noe som kan redusere modellens ytelse eller til og med føre til feilaktige konklusjoner.

Her er oppsummert resultater og det store bildet:
I betraktning av modellen uten data augmentering og modifikasjoner, ble det observert et tap på treningssettet på 0,13 ± 0,07 og nøyaktighet på 93,00 ± 2,56. Disse verdiene tjener som en grunnleggende referanse. På testsettet var tapet på 0,43 ± 0,20 og nøyaktigheten på 89,63 ± 4,90. Disse resultatene indikerer en viss grad av overfitting, ettersom treningsnøyaktigheten er høyere enn testnøyaktigheten - et kjent problem i dyp læring, som kan håndteres effektivt med data augmentering.

Med introduksjon av data augmentering i en enkel CNN, ble det observert en generell forbedring i ytelsen sammenlignet med basistilfellet. Tapet på treningssettet varierer fra 0,10 ± 0,06 til 0,14 ± 0,06, mens nøyaktigheten varierer fra 92,86 ± 2,12 til 94,03 ± 2,22. På testsettet varierer tapet fra 0,35 ± 0,11 til 0,60 ± 0,22, mens nøyaktigheten varierer fra 89,59 ± 5,81 til 91,42 ± 4,25. Dette demonstrerer hvordan data augmentering reduserer overtilpasning ved å tilby et mer variert treningssett, noe som bidrar til bedre generalisering på testsettet.

Videre, ved introduksjon av en modifisert CNN med tilpasning av det første reduksjonslaget, men uten data augmentering, blir det observert en ytterligere forbedring i ytelsen. På treningssettet var tapet 0,15 ± 0,06 og nøyaktigheten 92,59 ± 2,30. På testsettet var tapet 0,31 ± 0,16 og nøyaktigheten 90,74 ± 5,05. Dette underbygger at tilpasning av symmetriegenskapene fører til bedre forståelse av de underliggende mønstrene i datasettet, og dermed til bedre ytelse.

Endelig, ved kombinasjon av en modifisert CNN med data augmentering, ble det observert den beste ytelsen. På treningssettet varierer tapet fra 0,11 ± 0,06 til 0,15 ± 0,05, mens nøyaktigheten varierer fra 92,59 ± 2,01 til 93,90 ± 2,29. På testsettet varierer tapet fra 0,28 ± 0,05 til 0,53 ± 0,16, mens nøyaktigheten varierer fra 89,73 ± 5,30 til 92,41 ± 2,61. Denne ytelsesforbedringen

viser at kombinasjonen av dataaugmentering og tilpasning av det første reduksjonslaget gir de beste resultatene.

Ved å analysere sensitiviteten, som er evnen til å korrekt identifisere positive tilfeller, ble det også observert en forbedring med bruk av data augmentering. Den beste ytelsen ble observert i den modifiserte CNN-modellen med data augmentering, som oppnådde en sensitivitet på 0,92 ± 0,03, sammenlignet med basistilfellets 0,90 ± 0,05.

Presisjon, evnen til å identifisere positive tilfeller som faktisk er positive, fulgte samme trend. Den modifiserte modellen med data augmentering oppnådde den høyeste presisjonen på 0,93 ± 0,02, en forbedring fra basistilfellets 0,91 ± 0,03.

Hvordan unngikk modellen trent med modifisert reduksjonslagtilpassning og bruk av data augmneteringsteknikker å introdusere bias?

Basert på de gitte resultatene kan det diskuteres hvorfor modellen trent med modifisert reduksjonslagtilpasning og bruk av data augmneteringsteknikker ikke introduserte bias for å svare på forskningsspørsmåla.

Resultatene viser at alle data augmenteringsteknikkene som ble brukt i den modifiserte modellen, resulterte i forbedring sammenlignet med basistilfellet. Den modifiserte modellen kombinerte alle teknikkene og oppnådde den høyeste gjennomsnittlige nøyaktigheten på 92,41 ± 2,61, sammenlignet med basistilfellet med en gjennomsnittlig nøyaktighet på 89,63 ± 4,90. Sensitiviteten i den modifiserte modellen med data augmentering var også høyere enn i basistilfellet.

En enkel CNN-modell uten tilpasning av det første reduksjonslaget, men med data augmentering, viste en viss økning i nøyaktighet sammenlignet med basistilfellet. Resultatene fra den modifiserte modellen med data augmentering viste imidlertid at kombinasjonen av både tilpasning av reduksjonslaget og data augmentering ga enda bedre ytelse. Den modifiserte modellen uten data augmentering, men med tilpasning av symmetriegenskaper, oppnådde også en viss forbedring i ytelse sammenlignet med basistilfellet. Den beste gjennomsnittlige nøyaktigheten ble imidlertid oppnådd av den modifiserte modellen med data augmentering.

Ved å sammenligne de to beste modellene, gir mulighet til å se at den enkle modellen med data augmentering hadde et større standardavvik i ytelsen, noe som indikerer større variasjon mellom forsøkene. Den modifiserte modellen med data augmentering var derimot mer stabil og leverte mer konsistente resultater med mindre variasjon mellom forsøkene.

8 Konklusjon

Bachelorprosjektet hadde som formål å utvikle en ny analysestrategi ved å benytte maskinlæring, spesielt dyp læring og konvolusjonelle nevrale nettverk (CNN), for å klassifisere simulerte kollisjonsbilder fra ATLAS-detektoren. Prosjektet tok sikte på å forbedre sensitiviteten og nøyaktigheten i klassifiseringen ved å utnytte symmetriegenskaper i detektorbildene til data augmentering og tilpasse det første reduksjonslaget til å ta hensyn til symmetriegenskapene. I den anledning var det ønskelig å besvare følgende problemstilling:

> *Hvordan kan data augmentering implementert i en CNN-modell bidra til å øke sensitiviteten i klassifiseringen av detektorbilder fra ATLAS (HV)?*

Gjennom analyse av resultatene ble det funnet at den modifiserte modellen med alle kombinasjoner av teknikker for data augmentering ga de beste resultatene når det gjaldt forbedring av sensitiviteten og nøyaktigheten. Selv om det ikke var noen drastisk økning eller forbedring, viste den modifiserte modellen seg å være litt mer effektiv i læringsprosessen sammenlignet med den enkle modellen med data augmentering alene.

Ved å tilpasse det første reduksjonslaget til å ta hensyn til symmetriegenskapene og utnytte symmetriegenskaper gjennom data augmentering, oppnådde prosjektet sitt overordnede mål. Den nye analysestrategien, som involverte transformasjon av simulerte kollisjonsdata til bilder for videre analyse, økning av treningsdata gjennom ulike symmetriske manipulasjoner og evaluering av ulike variasjoner av CNN-modellen, viste seg å ha en positiv effekt på modellens ytelse og dens evne til å generalisere til nye data.

Diskusjonen av resultatene indikerer at kombinasjonen av data augmentering og tilpasning av symmetriegenskaper bidro til å redusere bias og forbedre modellens evne til å håndtere ulike bildevariasjoner og generalisere til ukjente data. Den beste modellen oppnådde høyere nøyaktighet enn både basistilfellet og den enkle modellen med data augmentering alene, noe som tyder på at kombinasjonen av justeringene var mer effektiv for å forbedre modellens ytelse. Dermed kan det konkluderes med at ved å bruke teknikker som tok hensyn til symmetriegenskapene, kunne sensitiviteten i klassifiseringen av detektorbilder fra ATLAS økes, selv om forbedringen ikke var drastisk.

8.1 Videre arbeid

Etter en grundig diskusjon og konklusjon av maskinlæringsprosjektet, åpenbarer det seg et betydelige potensiale for videre utforskning og forbedringer i dette domenet.

8.1.1 Endringer i datasett

Prosjektets innsikt gir mange muligheter for videre forskning, spesielt innen data augmentering og tilpasning av det nevrale nettverkets første reduksjonslag. En spennende retning kan være å inkludere andre fenomener enn de mikroskopiske sorte hullene og sfaleronene i datasettet. Ved å utforske denne tilpassede CNN-modellens ytelse på et bredere datasett, kan man få bedre forståelse av generaliseringskapasiteten. Videre kan prosjektet utvides fra binær til flerklasse-klassifisering, noe som vil gi innsikt i modellens robusthet og fleksibilitet når det gjelder å skille mellom forskjellige typer partikkelkollisjoner. Til slutt, ville det være interessant å se modellens ytelse på datasett der det er store forskjeller mellom klassene. En slik undersøkelse vil gi verdifull innsikt i modellens evne til å håndtere mer markant variasjon, noe som kan være nyttig for videre forbedring av modellen.

8.1.2 Optimalisering av CNN-arkitekturer og hyperparametre

Videreutvikling av prosjektet kan innebære utforsking og eksperimentering med flere og alternative CNN-arkitekturer. Gjennom å teste forskjellige arkitekturer, kan modellens ytelse potensielt bli forbedret, hvilket kan gi innsikt i de arkitektoniske valgene som støtter oppgaver relatert til partikkelkollisjonsklassifisering.

Eksperimentering med ulike nettverkslag og filterkonfigurasjoner kan også være nyttig for å optimalisere modellens ytelse. En systematisk tilnærming til justering av lagstrukturer og filtreparametere kan avdekke optimal konfigurasjon for prosjektets oppgaver.

Videre er det rom for forbedring gjennom justering av hyperparametre, som læringshastighet, batch størrelse, regulariseringsteknikker, antall treningsiterasjoner osv. Systematisk testing og observasjon av effekten på modellens ytelse kan gi fininnstilling av modellen og ytterligere forbedring i klassifisering av partikkelkollisjoner. Samlet kan disse forskningsveiene bidra til en forbedret modell og en dypere forståelse av beste praksis for bruk av CNN i denne konteksten.

8.1.3 Symmetribasert data augmentering

I forlengelsen av prosjektets arbeid med data augmentering og symmetrier, ligger det et stort potensial i å utvide metoder for å dra nytte av symmetri-egenskapene i bildedata. Det kan utforskes flere måter å utnytte disse egenskapene på for å generere et enda mer mangfoldig sett av treningsdata. Dette kan blant annet omfatte inkorporering av flere eksisterende teknikker for bildebehandling og -manipulasjon, utvikling av nye, spesialtilpassede metoder for data augmentering, eller en kombinasjon av disse tilnærmingene. Ved å utvide og forbedre prosjektets teknikker for data augmentering, kan man potensielt forbedre både kvaliteten på treningsdataene og modellens generelle klassifiseringsytelse.

I tillegg til å utvide anvendelsen av de utviklede metodene innenfor rammen av prosjektets opprinnelige kontekst, kan det være fruktbart å se på mulige applikasjoner av metodene i andre partikkelfysikk-eksperimenter og med forskjellige typer detektorer. Dette kan bidra til å validere og generalisere prosjektets funn i et bredere spektrum av oppgaver og settinger, noe som igjen kan øke den overordnede relevansen og bidraget til feltet. Det vil være viktig å tilpasse og finjustere metodene i henhold til de spesifikke kravene og utfordringene knyttet til de nye oppgavene og dataene, men grunnleggende prinsipper og teknikker er sannsynligvis overførbare.

Sluttelig, det kan også være interessant å utforske mulighetene for å anvende de utviklede metodene og teknikkene på klassifiseringsoppgaver som ligger utenfor partikkelfysikk. Maskinlæring er et svært fleksibelt og allsidig verktøy som kan brukes i et bredt spekter av domener og oppgaver, og de metodene som er utviklet i dette prosjektet kan potensielt ha verdifull overføringsverdi til andre oppgaver og domener. Dette kunne omfatte alt fra bildegjenkjennelse og tekstklassifisering til prediktiv modellering og anbefalingssystemer, blant mange andre potensielle anvendelser. Igjen, det vil være nødvendig med nøye tilpasning og finjustering av metodene for de nye oppgavene, men de grunnleggende prinsippene og teknikkene er sannsynligvis overførbare. På denne måten kan prosjektet ikke bare bidra til partikkelfysikk, men også til maskinlæring og dataanalyse mer generelt.

9 Referanser

Litteraturliste

Amini, A. (2023) *Introduction to Deep Learning* [Forelesningsvideo]. Introduction to Deep Learning 6.S191. Sted: Massachusetts Institute of Technology: MIT. Tilgjengelig fra: https://www.youtube.com/watch?v=QDX-1M5Nj7s&list=PLtBw6njQRU-rwp5__7C0oIVt26ZgjG9NI (Hentet: 30. april 2023)

Asano, S., Maruyama, T. og Yamaguchi, Y. (2009) Performance comparison of FPGA, GPU and CPU in image processing. *IEEE Xplore*. doi: 10.1109/FPL.2009.5272532. Tilgjengelig fra: https://ieeexplore.ieee.org/document/5272532 (Hentet: 9. mars 2023)

ATLAS Collaboration (2023a) *Detector and technology*. Tilgjengelig fra: https://atlas.cern/Discover/Detector (Hentet 25. februar 2023)

ATLAS Collaboration (2023b) *The ATLAS Experiment.* Tilgjengelig fra: https://atlas.cern/about (Hentet: 25. februar 2023)

ATLAS Collaboration (2023c) *The Higgs boson.* Tilgjengelig fra: https://atlas.cern/Discover/Physics/Higgs (Hentet: 22. februar 2023)

ATLAS Collaboration (2023d) *The Physics*. Tilgjengelig fra: https://atlas.cern/Discover/Physics (Hentet: 20.februar 2023)

ATLAS Collaboration (2023e) *Trigger and Data Acquisition.* Tilgjengelig fra: https://atlas.cern/Discover/Detector/Trigger-DAQ (Hentet: 20.februar 2023)

Brown, S. (2021) *Machine learning, explained.* Tilgjengelig fra: https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained (Hentet: 14. februar)

Buanes, T. og Mæland, S. (2023) *Utnytting av symmetri til å kunstig øke mengden treningsdata for klassifisering av kalorimeterbilder fra ATLAS(HVL)*. Institutt for data og realfag, Høgskulen på Vestlandet. Upublisert.

Buber, E. og Diri, B. (2018) Performance Analysis and CPU vs GPU Comparison for Deep Learning, s. 1-6. doi: 10.1109/CEIT.2018.8751930. Tilgjengelig fra: https://www.researchgate.net/publication/334168063_Performance_Analysis_and_CPU_vs_GPU_Comparison_for_Deep_Learning (Hentet: 25. april 2023)

Brownlee, J. (2018) *What is the Difference Between a Batch and an Epoch in a Neural Network?* Tilgjengelig fra: https://deeplearning.lipingyang.org/wp-content/uploads/2018/07/What-is-the-Difference-Between-a-Batch-and-an-Epoch-in-a-Neural-Network_.pdf (Hentet: 19. mai 2023)

CERN (2011) *Does CERN create black holes?* Tilgjengelig fra: https://angelsanddemons.web.cern.ch/faq/black-hole.html (Hentet: 21. mars 2023)

CERN (2023a) *ATLAS*. Tilgjengelig fra: https://home.cern/science/experiments/atlas (Hentet: 20. februar 2023)

CERN (2023b) *Extra dimensions, gravitons, and tiny black holes*. Tilgjengelig fra: https://home.cern/science/physics/extra-dimensions-gravitons-and-tiny-black-holes (Hentet: 25. februar 2023)

CERN (2023c) *The Large Hadron Collider*. Tilgjengelig fra: https://home.cern/science/accelerators/large-hadron-collider (Hentet: 24. februar 2023)

CERN (2023d) *The Standard Model*. Tilgjengelig fra: https://home.cern/science/physics/standard-model (Hentet: 25. februar 2023)

Chapman, P. mfl. (2000) CRISP-DM 1.0: Step-by-step data mining guide. NCR Systems Engineering Copenhagen, DaimerChrysler AG, SPSS Inc., OHRA Verzekeringen en Bank Groep B.V. Tilgjengelig fra: https://www.kde.cs.uni-kassel.de/wp-

content/uploads/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf (Hentet: 16. mai 2023)
Chen, M. og Sbert, M. (2022) A Bounded Measure for Estimating the Benefit of Visualization (Part I): Theoretical Discourse and Conceptual Evaluation. *Entropy*, 24(2), s.228. doi: 10.3390/e24020228. Tilgjengelig fra: https://www.mdpi.com/1099-4300/24/2/228 (Hentet: 4. mai 2023)
Cirilli, M. (2021) *CERN's impact on medical technology*. Tilgjengelig fra: https://cerncourier.com/a/cerns-impact-on-medical-technology/ (Hentet: 9. mars 2023)
Daria, S (u.å.) *Search for Sphalerons in Proton-Proton Collisions.*CERN: CERN Summer Student. Tilgjengelig fra: https://indico.cern.ch/event/650465/contributions/2653473/attachments/1494771/2325303/Search_for_Sphalerons_in_Proton-Proton_Collisions.pdf (Hentet: 21. mars 2023)
Dieleman, S., Fauw, J.D. og Kavukcuoglu, K. (2016). Exploiting Cyclic Symmetry in Convolutional Neural Networks, *Proceedings of The 33rd International Conference on Machine Learning, in Proceedings of Machine Learning Research*, 48, s. 1889-1898 Tilgjengelig fra: https://proceedings.mlr.press/v48/dieleman16.html (Hentet: 18. april 2023)
Dodig-Crnkovic, G. (2002) Scientific Methods in Computer Science. Philosophy Of Science: Science Method. Tilgjengelig fra: https://www.researchgate.net/publication/2563629_Scientific_Methods_in_Computer_Science (Hentet: 19. mai 2023)
Ellis, J. og Sakurai, K. (2016) Search for sphalerons in proton-proton collisions. *Journal of High Energy Physics.* 86 (2016) doi: 10.1007/JHEP04(2016)086. Tilgjengelig fra: https://link.springer.com/article/10.1007/JHEP04%282016%29086 (Hentet: 29. mars 2023)
Faiz, K. W. (2023) hjernen, i *Store medisinske leksikon* på snl.no. Tilgjengelig fra: https://sml.snl.no/hjernen (Hentet: 1. mai 2023)
Ghosh, A., mfl. (2022) EMD-Based Binary Classification of Mammograms. *Artificial Intelligence on Medical Data*. Lecture Notes in Computational Vison and Biomechanics, vol 37, s. 47-59. doi: 10.1007/978-981-19-0151-5_5. Tilgjengelig fra: https://link.springer.com/chapter/10.1007/978-981-19-0151-5_5 (Hentet: 29. april)
Goodfellow, I., Bengio, Y., og Courville, A. (2016) *Deep learning.* Massachusetts: MIT Press. https://www.deeplearningbook.org (Hentet 2. april 2023)
Goot, R. van (2021) We Need to Talk About train-dev-test Splits. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.368. Tilgjengelig fra: https://blogs.bournemouth.ac.uk/research/2011/08/10/referencing-dutch-flemish-german-names-in-the-harvard-system/ (Hentet 8. mars 2023)
Han S. H., mfl. (2018) Artificial Neural Network: Understanding the Basic Concepts without Mathematics. *Dement Neurocogn Disord*, 17(3), s. 83-89. 2018 doi: 10.12779/dnd.2018.17.3.83. Tilgjengelig fra: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6428006/ (Hentet: 30. april 2023)
Howard, J. og Gugger, S. (2020) *Deep Learning for Coders with fastai and PyTorch – AI Applications Without a PhD*. California: O'Reilly Media. Tilgjengelig fra: https://dl.ebooksworld.ir/books/Deep.Learning.for.Coders.with.fastai.and.PyTorch.Howard.Gugger.OReilly.9781492045526.EBooksWorld.ir.pdf (Hentet: 4. april 2023)
Hunter, J., mfl. (2012) Quick start quide. Tilgjengelig fra: https://matplotlib.org/stable/tutorials/introductory/quick_start.html#quick-start-guide (Hentet: 3. februar 2023)
HVL ATLAS Group (2023a) *About us*. Tilgjengelig fra: https://learningdarkmatter.com/about/ (Hentet: 22. februar 2023)

HVL ATLAS Group (2023b) *Home*. Tilgjengelig fra: https://learningdarkmatter.com (Hentet: 18. februar 2023)
IBM (2021) CRISP-DM Help Overview. Tilgjengelig fra: https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview (Hentet: 23. februar 2023)
IBM (2023) *What is a neural network?* Tilgjengelig fra: https://www.ibm.com/topics/neural-networks (Hentet: 19. februar 2023)
Kline, D. og Berardi, V. (2005). Revisiting squared-error and cross-entropy functions for training neural network classifiers. Neural Computing and Applications. 14, s. 310-318. doi: 10.1007/s00521-005-0467-y. Tilgjengelig fra: https://www.researchgate.net/publication/220372744_Revisiting_squared-error_and_cross-entropy_functions_for_training_neural_network_classifiers (Hentet: 7. mai 2023)
Lopes, A. og Perrey, M. L. (2022) *LHC the guide FAQ*. CERN-Brochure-2021-004-Eng. CERN: Geneva. Tilgjengelig fra: https://cds.cern.ch/record/2809109/files/CERN-Brochure-2021-004-Eng.pdf (Hentet: 20. februar 2023)
Murugan, P. (2017) Feed Forward and Backward Run in Deep Convolution Neural Network. doi: 10.48550/arXiv.1711.03278. Tilgjengelig fra: https://arxiv.org/abs/1711.03278 (Hentet: 13. mai 2023)
Nagpal, A. og Gabrani, G. (2019) Python for Data Analytics, Scientific and Technical Applications. *IEEE Xplore*. doi: 10.1109/AICAI.2019.8701341. Tilgjengelig fra: https://ieeexplore.ieee.org/document/8701341 (Hentet: 9. mars 2023)
NumPy Developers (2022) *NumPy: the absolute basics for beginners*. Tilgjengelig fra: https://numpy.org/doc/stable/user/absolute_beginners.html (Hentet: 13. februar 2023)
Nwankpa, C., mfl. (2018) Activation Functions: Comparison of trends in Practice and Research for Deep Learning. doi: 10.48550/arXiv.1811.03378. Tilgjengelig fra: https://arxiv.org/abs/1811.03378 (Hentet: 19. mai 2023)
Pedregosa, F., mfl. (2011) Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, s. 2825-2830. Tilgjengelig fra: https://scikit-learn.org/stable/ (Hentet: 23. Februar 2023)
PyTorch Contributors (2023) PyTorch Documentation. Tilgjengelig fra: https://pytorch.org/docs/stable/index.html (Hentet: 5. mars 2023)
Ramprasad, P., Raina, R. og Mondal, A. K. (2020) Effect of data-augmentation on fine-tuned CNN model performance. *IAES International Journal of Artificial Intelligence (IJ-AI)*, 10(1), s. 84-92. doi: 10.11591/ijai.v10.i1.pp84-92. Tilgjengelig fra: https://www.researchgate.net/publication/347437393_Effect_of_data-augmentation_on_fine-tuned_CNN_model_performance (Hentet: 18. april 2023)
Raschka, S. (2020) Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. doi: 10.48550/arXiv.1811.12808. Tilgjengelig fra: https://arxiv.org/abs/1811.12808 (Hentet: 31. mars)
Ruder, S. (2017) An overview of gradient descent optimization algorithms. doi: 10.48550/arXiv.1609.04747. Tilgjengelig fra: https://arxiv.org/abs/1609.04747 (Hentet: 4. Mai 2023)
Shawahna, A., Sait, S. og El-Maleh, A. (2018) FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review. *IEEE Xplore*. doi: 10.1109/ACCESS.2018.2890150. Tilgjengelig fra: https://ieeexplore.ieee.org/document/8594633 (Hentet: 29. mars 2023)
Singh, P. (2019a) Supervised Machine Learning. *Learn PySpark*. doi: 10.1007/978-1-4842-4961-1_6. Tilgjengelig fra: https://link.springer.com/chapter/10.1007/978-1-4842-4961-1_6#citeas (Hentet: 22. mars 2023)

Singh, P. (2019b) Unsupervised Machine Learning. *Learn PySpark*. doi: 10.1007/978-1-4842-4961-1_7. Tilgjengelig fra: https://link.springer.com/chapter/10.1007/978-1-4842-4961-1_7#citeas (Hentet: 22. mars 2023)

Shorten, C og Khoshgoftaar, T. M. (2019) A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*. 6 (60). doi: 10.1186/s40537-019-0197-0. Tilgjengelig fra: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0 (Hentet: 29. mars 2023)

Szalvay, V. (2004) *An Introduction to Agile Software Development*. Tilgjengelig fra: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2efe4d840631ebf026fede741e85195e36f8b134 (Hentet: 19. mai 2023)

Teuwen, J og Moriakov, N. (2020) Handbook of Medical Image Computing and Computer Assisted Intervention. *Academic press: The Elsevier and MICCAI Society Book Series*, s. 481-501. doi: 10.1016/B978-0-12-816176-0.00025-9. Tilgjengelig fra: https://www.sciencedirect.com/science/article/pii/B9780128161760000259 (Hentet: 20. mai 2023)

Too, E. C., Yujian, L., Njuki, S. og Yingchun, L. (2019) A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture*, 161, s. 272-279. doi: 10.1016/j.compag.2018.03.032. Tilgjengelig fra: https://www.sciencedirect.com/science/article/pii/S0168169917313303 (Hentet: 26. april 2023)

Wu, J. (2017) *Introduction to Convolutional Neural Networks*. Tilgjengelig fra: https://cs.nju.edu.cn/wujx/paper/CNN.pdf (Hentet: 31. mars 2023)

Zeiler, M. D., mfl. (2013) On rectified linear units for speech processing. *Vancouver: IEEE International Conference on Acoustics, Speech and Signal Processing*, s. 3517-3521. doi: 10.1109/ICASSP.2013.6638312. Tilgjengelig fra: https://arxiv.org/abs/1611.01491 (Hentet: 19. mai 2023)

Figurkilder

Amini, A., Soleimany, A., Karaman, S. og Rus, D. (2018) Spatial Uncertainty Sampling for End-to-End Control. *Neural Information Processing Systems (NIPS).*[Figur] Tilgjengelig fra: https://arxiv.org/abs/1805.04829 (Hentet: 24. april)

Amini, A. (2023) *Introduction to Deep Learning*. [Forelesningsnotat] Introduction to Deep Learning 6.S191. Sted: Massachusetts Institute of Technology: MIT. Tilgjengelig fra: http://introtodeeplearning.com/slides/6S191_MIT_DeepLearning_L1.pdf (Hentet: 30. april 2023)

Brice, M. (2020) *Installing the ATLAS calorimeter*. CC-BY-4.0. [Bilde]. Tilgjengelig fra: https://cds.cern.ch/images/CERN-EX-0511013-01 (Hentet: 9. mars 2023)

CERN Document Server (2008) *Event Cross Section in a computer generated image of the ATLAS detector.* CERN-GE-0803022. [Figur]. Tilgjengelig fra: https://cds.cern.ch/record/1096081 (Hentet: 23. mars 2023)

CERN Document Server (2015) *ATLAS event at 900 GeV – 5 May 2015 – Run 263962 Evt 20805.* ATLAS-PHO-Event-2015-005. [Figur]. Tilgjengelig fra: https://cds.cern.ch/record/2014009 (Hentet: 7. mars 2023)

CERN Document Server (2018) *Simulated production of a black hole in ATLAS.* CERN-EX-0705032. [Figur]. Tilgjengelig fra: https://cdsweb.cern.ch/record/1073733 (Hentet: 21. mars 2023)

CERN Document Server (2022) *The CERN accelerator complex, layout in 2022.* CERN-GRAPHICS-2022-001. [Figur]. Tilgjengelig fra: https://cds.cern.ch/record/2800984 (Hentet: 24. februar 2023)

Convolutional Neural Networks Explained (CNN Visualized) (2020) [Video]. Tilgjengelig fra: https://www.youtube.com/watch?v=pj9-rr1wDhM&t=267s (Hentet: 11. april 2023)

IBM (2023) *What is a neural network?* [Figur] Tilgjengelig fra: https://www.ibm.com/topics/neural-networks (Hentet: 19. februar 2023)

Saha, S. (2018) *A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way.* [Figur]. Tilgjengelig fra: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (Hentet: 31. mars 2023)

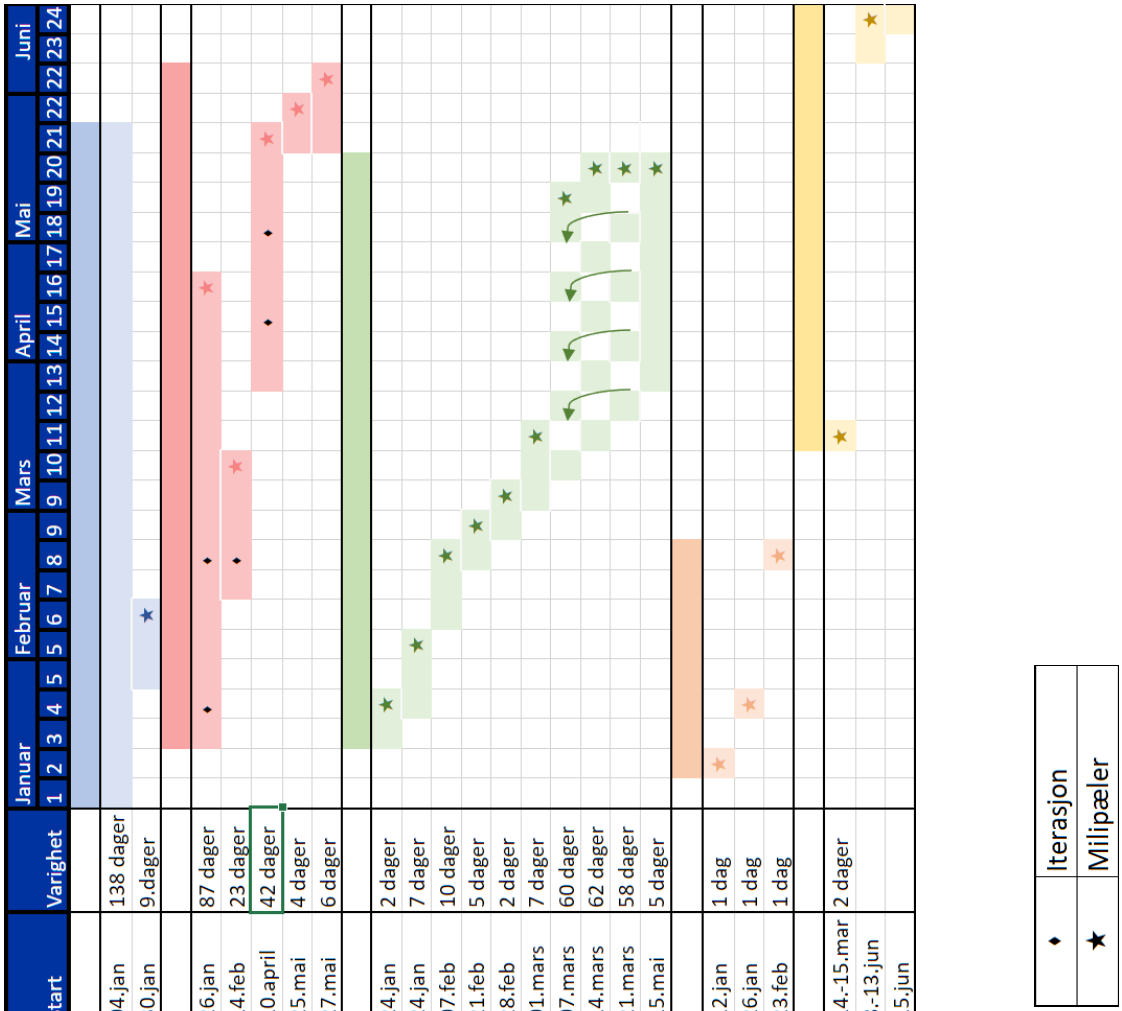
Shaffi, A. (2021). *Building a Confusion Matrix from Scratch* [Figur]. Tilgjengelig fra: https://medium.datadriveninvestor.com/building-a-confusion-matrix-from-scratch-85a8bfb97626 (Hentet: 1. mai 2023)

Wikimedia (2020) *File: Standard Model of Elementary Particles.svg* [Figur]. Tilgjengelig fra: https://commons.wikimedia.org/w/index.php?title=File%3AStandard_Model_of_Elementary_Particles.svg&oldid=401643300 (Hentet: 8. mars 2023)

Zanini, M (2022) *The organizational secrets of the Large Hadron Collider* [Figur]. Tilgjengelig fra: https://www.michelezanini.com/atlas/#fn1 (Hentet: 7. mars 2023)

10 Vedlegg

10.1 GANNT- diagram



ID	Fullført	Oppgave navn	Start	Varighet
		Forarbeid		
1	☑	Kompetansehevning og forskningsfasen	04.jan	138 dager
2	☑	Analyse av tidlegare bacheloroppgave	30.jan	9.dager
		Dokumentasjon		
3	☑	Prosjekthåndbok	26.jan	87 dager
4	☑	Midtveisrapport	14.feb	23 dager
5	☐	Endelig rapport	10.april	42 dager
6	☐	Refleksjonnotat	25.mai	4 dager
7	☐	EXPO plakat	27.mai	6 dager
		Modell utvikling		
8	☑	Oppsett av utviklingsmiljø	24.jan	2 dager
9	☑	Undersøke data	24.jan	7 dager
10	☑	Dataforståelse	07.feb	10 dager
11	☑	Dataforberedelse	21.feb	5 dager
12	☑	Sette opp et nevral nettverk	28.feb	2 dager
13	☑	Lage et grunnlag for evaluering	01.mars	7 dager
14	☑	Modellere	07.mars	60 dager
15	☑	Teste	14.mars	62 dager
16	☑	Evaluere	21.mars	58 dager
17	☑	Resultat	15.mai	5 dager
		Møter		
18	☑	Statusmøte 1 OA5	12.jan	1 dag
19	☑	Statusmøte 2 OA7	26.jan	1 dag
20	☑	Statusmøte 3 OA9	23.feb	1 dag
		Presentasjon		
21	☑	Midtveispresentasjon	14.-15.mar	2 dager
22	☐	Endelig presentasjon	8.-13.jun	
23	☐	EXPO Bergen	15.jun	

Januar 1 2 3 4 5 | Februar 5 6 7 8 9 | Mars 9 10 11 12 13 | April 14 15 16 17 18 | Mai 19 20 21 22 22 | Juni 23 24

◆ Iterasjon

★ Milipæler

Figur 10-1 – GANNT- diagram

10.2 Risikoanalysen

Tabell 10-1 – Risikoanalysen for prosjektet.

	Hendelse / risiko	Årsak	S	K	RP	Tiltak
1	Feiltolkning av oppgaven og tilhørende datasett	Grunnet prosjektets teoretiske og abstrakte natur er det lett å misforstå den. Datasettet kan misforståes uten riktig analyse	**4**	**5**	**20**	**Proaktive:** Jevn dialog med intern og eksterne veiledere. Se på relevant litteratur om problemstillingen og bli enige om en felles forståelse for oppgaven. **Reaktive:** Møte med veiledere for å finne ut av bakgrunnen for feiltolkningen, og sammen lage en plan for å rette opp i problemet.
2	Manglende kompetanse	Manglende kodeforståelse, kunnskaper om maskin-læring og/eller fysikk.	**4**	**5**	**20**	**Proaktive:** Lese seg opp på aktuelle teorier og begreper fra troverdige kilder. Be om hjelp fra eksterne veiledere og/eller HVL sin ATLAS gruppe ved eventuelle spørsmål / knutepunkter.
3	Fravær i gruppen	Plutselig fravær slik som sykdom og dårlig søvn, eller planlagt fravær slik som ferieturer.	**3**	**5**	**15**	**Proaktive:** Arbeide jevnt og trutt med prosjektet slik at fravær ikke får like store innvirkninger i progresjonen av arbeidet. **Reaktive:** Forsøk jobbe digitalt hvis sykdommen tillater
4	Interne konflikter	Dårlig samarbeid eller skjevfordeling av arbeid.	**3**	**4**	**12**	**Proaktive:** Jevn arbeidsfordeling og tydelig fremdriftsplan. **Reaktive:** Få kontakt med den andre parten og planlegge fremdriftsplanen bedre, muligens sammen med veileder.
5	Eksterne konflikter	Fraværende, utilgjengelig veileder Kompetanse mangel hos veileder	**2**	**4**	**8**	**Proaktive:** God og klar dialog med intern og ekstern veileder (oppdragsgiver) gjennom prosjektløpet. **Reaktive:** Digitale møter og kommunikasjon over nett. Ta kontakt med andre fagfolk som tilbyr hjelp.
6	Tekniske problemer	Menneskelig svikt Programvarefeil Strømbrudd Brann	**2**	**2**	**4**	**Proaktive:** Jevnlig lagre arbeidet og bruke sky-/ nettbaserte tjenester. **Reaktive:** Ta kontakt med veiledere eller IT-hjelpen.