Norges miljø- og biovitenskapelige universitet

Masteroppgave 2024 30 stp
Fakultet for realfag og teknologi

Deteksjon av grøftet myr ved bruk av maskinlæring på digital terrengmodell

Detection of Peatland Ditches using Machine Learning and Digital Terrain Model

Tobias Bjørnstad
Geomatikk

Forord

Denne masteroppgaven er avslutningen av en femåring mastergrad i Geomatikk ved Norges miljø- og biovitenskapelige universitet (NMBU). Jeg er veldig fornøyd med at jeg har fått en såpass konkret oppgave med spennende problemstilling og relevante samfunnsspørsmål å jobbe med de siste fem månedene. Oppgaven er gitt av NIBIO med Nicolai Munsterhjelm som ekstern veileder, og Misganu Debella-Gilo som hovedveileder fra NMBU.

Med dette ønsker jeg å rette en stor takk til Nicolai for stødig veiledning gjennom hele prosessen, samt raske og alltid gode svar på epost.

Jeg ønsker også å takke Misganu for alltid gode tips og triks, og for uvurderlig hjelp under selve skrivingen av oppgaven.

Takk til NIBIO for oppgaven og datasettet, og for at jeg har fått ta del i deres forskning.

En stor takk rettes til familie og venner for all støtte og motivasjon gjennom studietiden.

Jeg vil også takke alle medstudenter for fem år med lunsjer på Sørhellinga, lange dager på lesesalen og onsdagskvelder i Bodegaen. Særlig takk til Gents Academy.

Til sist rettes en stor takk til Aurora for all støtte og kjærlighet under arbeidet med oppgaven, og for god hjelp til korrekturlesing. Du har vært en uvurderlig ressurs for meg det siste halve året.

Tobias Bjørnstad

Ås, 14. mai, 2024

Sammendrag

Myr er et viktig økosystem, ikke bare for det biologiske mangfoldet den huser, men også på grunn av de klima- og miljøregulerende økosystemtjenestene den tilbyr. Grøfting av norske myrområder har ført til hydrologiske endringer i disse områdene, som igjen har hatt påvirkning på økosystemtjenestene som tilbys. Økt fokus på miljømessig risiko knyttet til denne tematikken har ført til et behov for kartlegging av grøftede myrer.

I denne oppgaven undersøkes muligheten for bruk av maskinlæring for deteksjon av grøftede myrer ved bruk av digital terrengmodell. To maskinlæringsmodeller er trent på et datasett med totalt 2656 bilder av myrområder fra hele Norge, hvor halvparten av bildene er av grøftet myr. Den ene modellen er et konvolusjonelt nevralt nettverk som tar i bruk overføringslære med ResNet50V2 som forhåndstrent modell for binær klassifisering av bildene. Den andre er en støttevektormaskin. De to modellene er utviklet for å se på forskjellene mellom klassisk maskinlæring og dyplæring innen binær klassifisering av digital terrengmodell.

Dyplæringsmodellen er optimalisert ved hyperparameter tuning og oppnådde en F1-score på 0,94 og en MCC på 0,87. Støttevektormaskinen ble også hyperparameteroptimalisert og landet på en F1-score på 0,83 og en MCC på 0,66. Oppgaven viser at kartlegging av myrgrøfting kan gjøres effektiv ved bruk av maskinlæring på digital terrengmodell, og at dyplæring er en svært god metode for dette. Modellen tar kun én topografisk indeks som input, hvilket gjør den mulig å benytte i stor skala, selv ved begrenset computerkraft tilgjengelig.

Abstract

Peatlands are important ecosystems, not only because of the biodiversity it hosts, but also for the climate and environmental regulatory ecosystem services it provides. Drain ditches in Norwegian peatlands has led to hydrological changes in these areas, which in turn have impacted the ecosystem services provided. Increased focus on environmental risks associated with this issue has created a need for mapping these ditches.

This study investigated the use of machine learning for detecting peatlands ditches using digital terrain models. Two machine learning models were trained on a dataset containing a total of 2656 images of Norwegian peatlands, half of which were ditched. The first model is a convolutional neural network utilizing transfer learning, with ResNet50V2 as the pre-trained backbone, for binary classification of the dataset. The other model is a support vector machine. The two models were developed to compare classical machine learning to deep learning in the binary classification of digital terrain models.

The deep learning model was optimized by hyperparameter tuning and achieved a F1-score of 0.94 and a MCC of 0.87. The support vector machine was also hyperparameter optimized and achieved a F1-score of 0.83 and an MCC of 0.66. The study demonstrates that mapping of peatland ditches can be done effectively by using machine learning on digital terrain models, and that deep learning is an effective way of doing this. The model only takes one topographic index as input, making it possible to use on a large scale, even when limited computational power is available.

Innhold

Figurer

Tabeller

1 Introduksjon

1.1 Bakgrunn og formål

Myr anses som den naturtypen med absolutt høyest karbontetthet både i Norge og globalt. Allikevel har vi sett hvordan et betydelig forfall av myrområder her til lands, grunnet drenering for etablering av jordbruksareal, skogbruk eller torvuttak, har gjort disse områdene til enorme kilder for menneskeskapt CO_2-utslipp (Leifeld et al., 2019). I rapporten «Klimagassutslipp fra torvproduksjon i Norge» fra 2017 (Søgaard et al., 2017), skriver NIBIO at «Det er store utfordringer knyttet til datagrunnlaget for rapportering av utslipp fra torvproduksjon i Norge, både med hensyn på arealer og volum, og i rapporten beskrives usikkerhet knyttet til dagens datagrunnlag og noen muligheter for å forbedre dette datagrunnlaget.».

På grunn av det økte søkelyset på våtmark som det økosystemet med størst innvirkning på klima og artsmangfold, er det et generelt ønske om økt kartlegging av disse områdene. I 2023 ga Forskningsrådet 7 millioner kroner til prosjektet «Landsdekkende våtmarksdatasett». Prosjektet vil øke kunnskapen vi har om myrer i Norge ved å utvikle et heldekkende kartlag for våtmark, for kommuner og andre myndigheter å ta i bruk, for å ta vare på disse områdene (Mekjan, 2023). Vi vet at spesielt torvutvinning slipper ut betydelige mengder klimagasser, og da er det essensielt å vite omfanget av torvuttak/grøftet myr når Norges nasjonale klimagassregnskap skal beregnes.

I denne masteroppgaven testes mulighetene for deteksjon og klassifisering av grøftet myr ved bruk av dyplæring på høydedata i form av digital terrengmodell, for å kunne bedre datagrunnlaget for fremtidige rapporter. Grøfting har et distinkt avtrykk i en digital terrengmodell, med rette linjer i et forgreinende nettverk. Det er da rimelig å anta at de kan oppdages ved hjelp av maskinlæring. Oppgaven vil dreie seg om å oppdage torvuttak og grøftet myr ved bruk av digital terrengmodell og maskinlæring.

1.2 Oppgavens fremgangsmåte og oppbygning

I forberedelsen til oppgaven har jeg gått frem ved å lese meg opp på relevant teori for å utvikle en dypere forståelse for oppgaven og dens formål. Modellen er utviklet etter prinsipper jeg lærte i faget DAT300 ved NMBU høsten 2023.

Oppgaven er delt inn i 6 kapitler. Kapittel 1 er introduksjonen til oppgaven, med bakgrunn for studien, fremgangsmåte og relatert arbeid. Kapittel 2 ser på teori knyttet til forskningsspørsmålet og presenterer konsepter og prinsipper for metodikken som legger bakgrunnen for arbeidet som er gjort. Kapittel 3 tar for seg metoder dataprosessering og optimalisering av modellene, samt visualisering av datasettet. Kapittel 4 presenterer resultater, mens kapittel 5 diskuterer datakvalitet, treningen av modellene og deres ytelse, i tillegg til relaterte studier og videre arbeid. Og til sist, kapittel 6 som oppsummerer funn i studien, foreslår videre arbeid innen feltet og konkluderer masteroppgaven. I vedlegg A finnes kildekoden, samt oversikt over programmer og Python-moduler som er tatt i bruk.

2 Teori

I dette kapittelet presenteres det teoretiske grunnlaget for metodene som er brukt i oppgaven. Kapittelet tar for seg ulike relaterte arbeider, før myr og ulike aspekter ved våtmark nevnes. Deretter dykkes det dypere inn i bildefiltrering og digitale terrengmodeller, før kapittelet til slutt avsluttes med en presentasjon av teori og anvendelse av maskinlæring.

2.1 Relatert arbeid

Lenge var den beste metoden for klassifisering og innsamling av myrdata feltbasert oppmåling og digitalisering av høyoppløselig data fra satellitter, flyfoto eller LiDAR (Robb et al., 2023). Rapinel et al. viser i «Ditch network extraction and hydrogeomorphological characterization using LiDAR-derived DTM in wetlands» (Rapinel et al., 2015), hvordan en kan bruke manuell bildeanalyse på DTM for å kartlegge grøfter i våtmarksområder i nord-østlige Frankrike. I Belgia benyttet Roelens et al. klassisk maskinlæring ved Random Forest klassifisering på en kombinasjon av topografiske og radiometriske egenskaper hentet fra LiDAR punktsky, for å kartlegge av grøftelinjer (Roelens et al., 2018). For å kartlegge grøftet myr i Skottland tok Robb et al. i bruk en U-net-basert dyplæringsmodell med flyfoto som treningsdata, og fikk segmentert grøfter i skotsk myr med 79 % nøyaktighet (Robb et al., 2023). Lidberg et al. brukte en liknende framgangsmåte som Robb et al. for å segmentere grøftelinjer i Sverige, Finland og Baltikum. Her benyttet de topografisk data fra flybåren laserfangst i form av en digital terrengmodell og oppnådde en nøyaktighet på 86 % (Lidberg et al., 2023). Utviklingen beveger seg mot at det er konvolusjonelle nevrale nettverk og topografisk laserdata som burde brukes for å oppnå best resultater når det kommer til klassifisering og identifisering av grøftet myr.

2.2 Myr

I sin rapport «Utfasing av torvuttak i Norge - effekter på naturmangfold og andre viktige økosystemtjenester» definerer Øien et. al. myr som «et landområde med fuktighetskrevende vegetasjon som danner torv» (Øien et al., 2017). Bárcena definerte i 2016 myr som «[…] et areal med myrvegetasjon og et minst 30 cm tykt torvlag som inneholder minst 40 prosent organisk materiale» (Bárcena et al., 2016). Felles for disse definisjonene er at de mener myr forekommer i områder der tilførselen av vann er høyere enn fordampningen. Det vil i områder

med høy fuktighet og lavt oksygennivå legges til rette for produksjon av torv. Torv dannes av døde planterester som på grunn av lavt oksygennivå, høy surhet og generell mangel på næringsstoffer ikke kan brytes ned, men heller delvis komposteres og pakkes sammen (Hofstad, (2005-2007)).

Norske myrer kan deles inn i to hovedgrupper: jordvassmyr, som får næringen sin fra grunnvannet, og nedbørsmyr, som tar næring fra nedbør. Dannelse av norske myrområder startet rett etter siste istid, og de eldste avsetningene er mellom 8000 og 10 000 år gamle [(Gya, 2023), (Vilde Fluge Lillesund et al., 2018)]. I 2010 ble karboninnholdet i norske myrer estimert til 950 millioner tonn, som betyr et CO_2-innhold på omtrent 3,5 milliarder tonn (Vilde Fluge Lillesund et al., 2018).

Myrområder utgjør i dag kun 3 % av verdens landareal, men lagrer allikevel omtrent 20 % av karbonet vi har i jordsmonnet. For å opprettholde myras egenskaper er det svært viktig å opprettholde dens krav om høy vannstand. Kyrkjeeide skrev at drenering er den største trusselen mot myr i Norge, tett etterfulgt av utbygging og torvuttak. Disse ødeleggelsene av myr ble estimert til å utgjøre 10 % av Norges totale utslipp i 2013. Estimatet er basert på et myrareal på 3618 km^2, men NINA mener at det faktiske arealet sannsynligvis er nærmere 7000 km^2 (Kyrkjeeide et al., 2020). Drenering av myrområder vil i tillegg til utslipp av karbon lagret i torv, hindre ny karbonlagring ved fremtidig torvdannelse.

Økosystemtjenester er fordelaktige funksjoner og tjenester som ulike økosystemer tilbyr mennesker og andre organismer, noen av dem kan også ha negative effekter på selve økosystemet de forkommer i ved overforbruk av tjenesten. Disse tjenestene deles inn i fire hovedkategorier: støttende, forsynende, regulerende og kulturelle tjenester (Øien et al., 2017). Tabell 2.1 viser ulike tjenester myrområder har å tilby. En av grunnene til det økte søkelyset på myrområder de siste årene, er deres rolle som klimaregulator. Myr og våtmark lar ikke organisk materiale bryte fullstendig ned, i stedet vil døde planter og andre organismer akkumuleres over tid og etter hvert synke ned i myra og omdannes til torv. Det er denne prosessen som fører til at karbon, i form av organisk materiale, blir lagret i myrområder (Gya, 2023).

Tabell 2.1: *Ulike økosystemtjenester som tilbydes av intakt myr. Tjenestene er også markert med hvorvidt utnyttelse kan foregå på en bærekraftig måte (Øien et al., 2017).*

Type økosystemtjenester		**Eksempel**	**Bærekraftig/destruktivt**
Støttende	Artsmangfold	Habitat for arter	Bærekraftig
	Jordoppbygging	Akkumulering av organisk materiale	Bærekraftig
	Næringsomsetning	Lagring, nedbryting og prosessering av næringsstoffer	Bærekraftig
Forsynende	Ville planter og bær	Plukking av molter og tranebær	Bærekraftig
	Vinterfôr	Slått av grasliknede vekster	Bærekraftig, men kan påvirke struktur og prosesser
	Beiteareal	Husdyr på beite på myrareal	Ofte bærekraftig, men vil påvirke negativt ved intensiv bruk (tråkkproblematikk)
	Torv til ulike formål	Uttak av torv til brensel eller vekstmateriale	Destruktivt
	Trevirke	Skogreising på drenert myr	Destruktivt
Regulerende	Klimaregulering	Produksjon og lagring av karbon i torvmasser	Bærekraftig
	Vasskvalitet	Filtrering av grunnvatn/nedbørsvatn gjennom torvmasser	Bærekraftig
	Flomdemping	Forsinket utflomming av nedbørsvatn gjennom torvmasser	Bærekraftig
	Branndemping?		
Kulturelle	Friluftsliv og rekreasjon	Bruk av myrareal som turområde	Stort sett bærekraftig, men vil påvirke negativt ved intensiv bruk (tråkkproblematikk)
	Vitenskapelig og historisk dokument	Klimahistorisk lager gjennom gamle torvmasser	Bærekraftig

2.2.1 Torvuttak

Torvuttak har forgått lenge i Norge, og kan spores over tusen år tilbake. Tidligere ble torv brukt som energiressurs ved at den ble tørket og brent. I dag høstes torv hovedsakelig til hagebruk og planteproduksjon på grunn av dens gode egenskaper for vannretensjon og drenering (Vilde Fluge Lillesund et al., 2018). Uttak av torv foregår ved at en først drenerer myra, enten ved dreneringsgrøfter eller vha. rørsystemer før torven skjæres opp. Her fjernes det øverste torvlaget, enten ved at en skjærer ut blokker, eller med store maskiner som harver opp torven. Etter torvskjæringen tørkes den, før den pakkes og transporteres ut på markedet.

I en kubikkmeter med torv er det lagret opp til 50 kg karbon, som vil si at dersom denne torven oksideres eller brennes vil den frigjøre opp til 180 kg med CO_2. Det er imidlertid ikke bare den

høstede torven som bidrar til utslipp: Ved grøfting vil all myr i område sakte tørke opp og frigjøre lagret karbon. Når torv tørkes og kommer i kontakt med luft, vil det meste av lagret karbon omgjøres til CO_2 og forsvinne ut i atmosfæren (Hofstad, (2005-2007)).

Etter at driften av torvuttak avsluttes vil en stå igjen med et karakteristisk dreneringssystem som brede grøfter systematisk plassert i terrenget. Det finnes en rekke metoder for miljøvennlig etterbehandling av myrområder, hvor målet er å restaurere torvmarka, men det å fylle igjen grøftene er tidskrevende og tungt arbeid, og har derfor sjeldent blitt gjort i Norge. Ved utfasing av et torvuttak uten noen form for etterbehandling vil områdene fortsette å tørke opp og lekke klimagasser i lang tid fremover (Vilde Fluge Lillesund et al., 2018).

2.2.2 Skogbruk på myr

En del myrer i Norge er også utsatt for drenering som ikke er relatert til torvuttak. Nedrivning av myrer til fordel for skogreising er den vanligste påvirkningen på norske myrer, med et anslag på rundt 4100 km^2 med myr grøftet for dette formålet fram til 1995 (Øien et al., 2017). Drenering av myr for etablering av skogdrift er i dag ulovlig etter «Forskrift om berekraftig skogbruk» [(Lovdata, 2006), (Øien et al., 2017)]. Etter etablering vil grøftesystemet fortsette å drenere og føre til fortsatt nedbrytning av torv. Slike grøfter vil ikke opptre like systematiske som ved torvuttak, og har ofte større mellomrom mellom hver grøft. Selv om skog dyrkes over torven vil grøftene synes i laserbasert datafangst, da laserpulsene trenger gjennom blader og ned til bakken.

2.2.3 Kartlegging av myr

Lenge ble myr sett på som unyttige arealer det var vanskelig å jobbe med. Sur og oksygenfattig jord, høy grunnvannstand og mye insekter gjorde myrlandskap til en lite attraktiv plass å arbeide med, og ble på grunn av dette ansett som arealer kun egnet for nydyrking og skogreising. Dette er grunnen til at vi i Norge har et nokså godt kartlag for myr som befinner seg under skoggrensen. Myr ble tidligere klassifisert etter om den bar preg av myr- eller torvmark, og om den var egnet som dyrkingsjord eller skogreisningsmark. Var den egnet for dette ble det klassifisert etter torvdybde, omdanningsgrad og vegetasjon (Einevoll, 1980). Myrkosten, sett i figur 2.1, var figuren som ble brukt i økonomisk kartverk for å kartlegge myra og dens lokale egenskaper.

Figur 2.1: *Eksempel på bruk av myrkosten. Figuren gir informasjon om myrens dybde, vegetasjon, torvlag, osv. (Einevoll, 1980).*

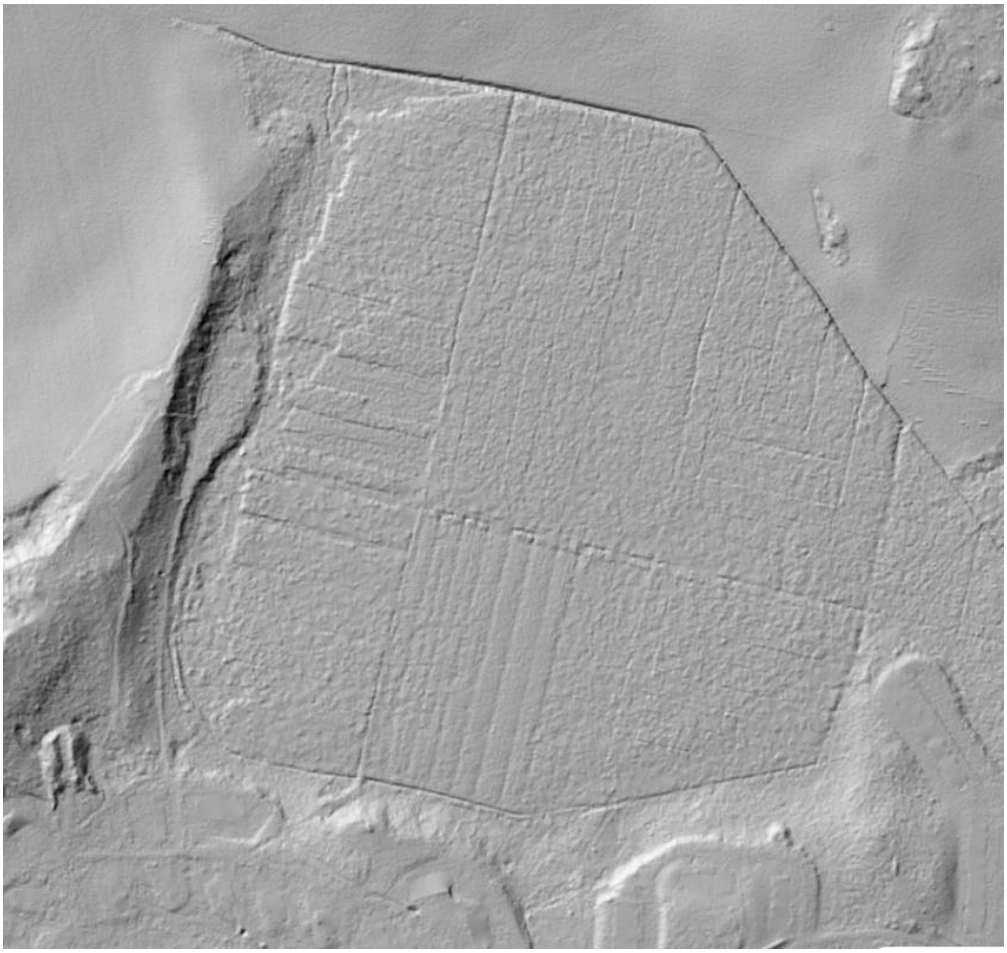
I dag er det andre interesser som driver kartleggingen av myr fremover. Myras rolle som tilbyder av økosystemtjenester er i vinden, og det er derfor et ønske om å nøyaktig klassifisere myrarealene i Norge. Det er særlig i kartleggingen av myr over tregrensa hvor behovet i dag er stort.

Det er en del utfordringer knyttet til kartlegging av myrområder. Lenge var flyfotografering og feltobservasjoner de eneste kildene en benyttet i kartleggingen, men etablering av skogbruk og dyrkamark på gamle torvuttak gjorde det vanskelig å oppdage gamle grøftede myrer. Over skoggrensa er terrenget vanskeligere å ta seg rundt i, og det ble benyttet flybilder uten omfattende feltobservasjoner (Hatlevik, 2021). Her har fremskritt innen fjernmålingsfaget bedret datafangsten, og i dag benyttes maskinlæring og fjernmålingsteknikker som innebærer bruk av multispektrale bilder og digitale terrengmodeller til kartlegging av myrområder. I 2018 fant NIBIO at myr utgjorde 8,9 % vegetasjonsarealet i Norge, ikke 5,8 % som en trodde tidligere. Denne endringen betydde at Norge hadde 10 000 km^2 mer myr enn tidligere antatt (Bryn et al., 2018). Arbeidet ble gjort ved å manuelt gjennomgå et representativt utvalg flater fra hele landet. En kan se for seg at økt bruk av kunstig intelligens (KI) i fjernmåling vil øke nøyaktighetsgraden av arealkartleggingen. KI vil gjøre det mulig å ikke bare se på utvalg slik det tidligere er gjort, men studere bilder av hele Norge for å få et mer nøyaktig estimat.

2.3 Digital terrengmodell i ressurskartlegging

En digital terrengmodell (DTM) tar i bruk en punktsky for å kartlegge terrengoverflaten strippet for bebyggelse, vegetasjon og andre objekter som måtte befinne seg i modellen. En punktsky konstrueres av rådata fra LIDAR-skanninger, dette kalles en digital overflatemodell (DOM). Objekter som ikke tilhører terrenget identifiseres og klassifiseres, før overflateobjekter fjernes.

Modellen er nå kantete og ujevn, og må derfor glattes ut for å gi en realistisk modell som også er lesbar. For å få en jevn modell estimeres flatene mellom punkter via interpolering (Dong & Chen, 2017). En vil så sitte igjen med en digital terrengmodell som ser ut som figur 2.2. Kartverket er hovedleverandør av terrengmodeller i Norge og produserer modeller med 1, 10 og 50 meters oppløsning for landområder (Kartverket, 2024).



Figur 2.2: *Digital terrengmodell av Åsmåsan i Ås kommune. Bildet viser karakteristiske grøftelinjer i terrenget selv om området er dekket av skog. Hentet fra høydedata.no.*

Digitale terrengmodeller har flere bruksområder. De brukes blant annet til å finne høydeverdier på objekter ved å sammenlikne DTM mot DOM, eller for å gjøre endringsanalyser innen skogbruk. En annen bruk er i anleggsbransjen, hvor DTM brukes for å gjøre masseberegninger. I en DTM vil det dessuten komme frem tydelige mønstre i terrenget som ofte ikke er synlige med det blotte øye, blant annet kommer stier og gamle bebyggelser til syne under det som på

et flybilde kanskje bare vil sett ut som skog. Grøftede myrer er også blant disse topografiske mønstrene som vil visualiseres godt i en DTM. Grøftingen etterlater seg lange parallelle grøfter, så dype at det er et mønster det er lett å kjenne igjen i en DTM. En ser i figur 2.2 at de over 100 år gamle grøftene i Åsmåsan i Ås kommune kommer til syne, gjennom skogen som står der i dag, i en DTM (French, 2016).

2.4 Filtrering

Laplace-filter

Laplace-filtrering er et konvolusjonelt filter som ofte brukes for deteksjon av lineære elementer i bilder (Haralick et al., 1987). Filteret tar i bruk gråtonebilders andrederiverte for å finne kanter, ved å se på endringer i gradienten til pikslene, og dermed finne områder hvor intensiteten endres plutselig. Det er viktig å merke seg at Laplace er et filter som kun finner kanter i bildet, og ikke tar hensyn til styrke eller retning. Filteret tar i bruk en konvolusjonskjerne som legges over hver eneste piksel i bildet. For et standard 2D Laplace-filter benyttes følgende kjerne:

$$D_{xy}^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (1)$$

Bruk av denne kjernen i filtreringen vil finne pikselens nettoverdi, som indikerer graden av endring i intensitet rundt pikselen.

Canny-kantdetektor

Canny-kantdetektor er en bildebehandlingsteknikk utviklet av John F. Canny i 1986 for identifisering og ekstraksjon av kanter i bilder (Canny, 1986). Filteret reduserer støy i bildet ved å ta i bruk Gaussisk uskarphet, for å unngå feilaktige kantdeteksjoner. Deretter brukes en Sobel-operator for å finne gradientene, ved førstederiverte i horisontal og vertikal retning, som gir retningen og styrken til kantene. Canny-filtreringen tar så i bruk en lav og en høy terskelverdi. Kanter med gradientverdier over den høye terskelen aksepteres som kanter, mens gradienter under den lave avvises. Gradientverdier mellom terskelverdiene aksepteres som kanter dersom de er koblet til de sterke kantene.

2.5 Maskinlæring

«The primary usefulness of all machine learning models is gained from their ability to generalize their learning from seen training data to unseen examples.» (Aggarwal, 2018).

Maskinlæring er et bredt felt med like mange algoritmer som det er oppgaver. Allikevel kan vi dele alle disse ulike metodene inn i tre grupper: veiledet læring (eng: supervised learning), ikke-veiledet læring (eng: unsupervised learning), og forsterket læring (eng: reinforcement learning). Disse kategoriene beskriver hovedmåten modellen lærer fra dataene på, og de har forskjellige bruksområder og egenskaper. Modeller som bruker veiledet læring trener på par av klassifisert og merket (eng: labeled) data. Dataen deles inn i trenings- og testdata hvor testdataens output behandles som sanne, og algoritmen kalkulerer og endrer vekter for å prøve å oppnå de samme svarene som i testdataen. Når modellen etter hvert produserer tilfredsstillende resultater, kan en forsøke å estimere resultater på ny, usett data (Tidemann & Elster). Veiledet læring er en populær metode som brukes i bla. konvolusjonelle nevrale nettverk. I ikke-veiledet læring får modellen tilført treningsdata uten tilknytning til sanne verdier. Dette krever at algoritmen selv identifiserer strukturer og mønstre i dataene.

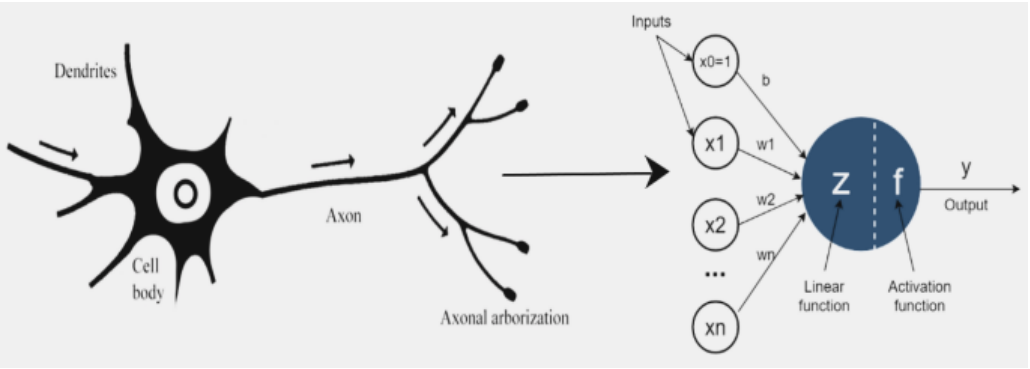
En viktig gren innen maskinlæring er dyplæring, hvor det utvikles algoritmer basert på strukturen til den menneskelige hjernen. Metoden tar i bruk nevrale nettverk med lag av noder, kalt nevroner, koblet sammen med informasjon som flyter frem og tilbake mellom lagene. En viktig del av dyplæring er algoritmenes evne til å selv oppdage når de gjør feil, og endre fremgangsmåten før neste forsøk. Noen eksempler på oppgaver som i dag jobbes på med dyplæring er bildegjenkjenning, språk- og chatmodeller, generative bildemodeller og talegjenkjenning.

Dyplæring krever store datamengder for å kunne trenes effektivt. Dette gjør at det ved enkelte oppgaver kan være krevende å finne tilstrekkelig med treningsdata til å utvikle gode generaliserte modeller. Det er også knyttet lang beregningstid og store ressurskrav til disse metodene, da modellene ofte er store og avhengig av kraftig maskinvare. Opplæring tar ofte betydelig tid, spesielt ved større modeller. Allikevel anses gevinsten av dyplæring for så stor at det er blitt synonymt med «kunstig intelligens», og det er snakk om at vi står ovenfor en «datarevolusjon». Denne revolusjonen skyldes metodens forbedring av maskinlæringens ytelse, allsidighet, autonomi og tolkningskapasitet. Dette er oppgaver som tidligere var vanskelige eller lite effektive å løse med klassisk maskinlæring.

Å håndtere overtilpasning (eng: overfitting) er en av de store utfordringene en vil møte på, dersom en tar i bruk maskinlæring. Det kan være fristende å tilpasse modellen perfekt til treningsdataene for å oppnå høy presisjon i estimeringene her. Problemet er at da vil modellen ofte prestere mye svakere på ny usett data. Målet med maskinlæring er å utvikle en modell som produserer estimater av med høy presisjon på ny data (Teien, 2022). Ny data vil ofte ikke følge det samme mønsteret som treningsdataen, og en god modell vil produsere gode resultater på generelle problemer, men heller med noen feil, enn å produsere perfekte resultater kun på treningsdataene (Dietterich, 1995).

2.5.1 Kunstige nevrale nettverk

Nodene i en dyplæringsalgoritme kalles nevroner, og et nettverk med lag av nevroner kalles et nevralt nettverk. I figur 2.3 ser vi at på samme måte som aksoner og dendritter kobler sammen nevronene i synaptiske koblinger i menneskets nervesystem, kobles neuronene i et kunstig nevralt nettverk sammen med vekter. Disse kunstige vektene endres på samme måte som at de synaptiske koblingene endres ut fra ulike sanseinntrykk. Det er slik både mennesker og kunstige nevrale nettverk (ANN) er i stand til å lære (Aggarwal, 2018).



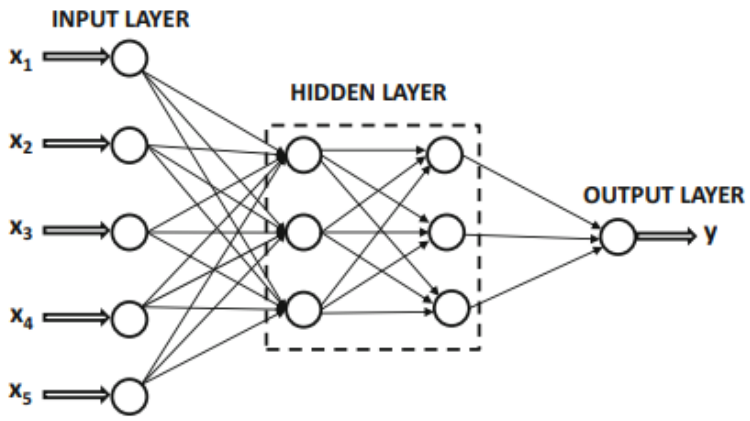
Figur 2.3: *Illustrasjon av hvordan nevroner kobles sammen av synaptiske koblinger både i hjernen og i kunstige nevrale nettverk (Thorat et al., 2022).*

Allerede på 1950-tallet ble ideen om at mennesker og datamaskiner kan lære på samme måte syntetisert da Frank Rosenblatts perceptron-maskin så dagens lys. Modellen bygde på W. McCulloch og W. Pitts teori og beskrivelse av hvordan nevroner kunne fungere (McCulloch & Pitts, 1943). Perceptron var en enkel algoritme for lineær binær klassifisering. Modellen tar

flere binære verdier som input, beregner en vektet sum av inputen før det aktiverer output basert på en bestemt terskelverdi. Selv om modellen hadde begrenset evne til å løse komplekse, ikke-lineære problemer, la den grunnlaget for senere utvikling av mer avanserte nevrale nettverk.

Flerlags perceptron (eng: multilayer perceptron) (MLP) kobler flere enkle nevroner sammen i et nevralt nettverk. Dersom alle nevroner i et lag er koblet til alle nevroner i et annet, sier en at laget er fullt tilkoblet (eng: fully connected). En kan legge til flere lag med nevroner i skjulte lag (eng: hidden layers) i modellen, og mellom lagene finner vi vekter som kobler dem sammen, se figur 2.4. Målet er å finne en funksjon av inputnevronene for å løse oppgaver av usett data (Raschka & Mirjalili, 2019). MLP propagerer fra inputnevron til outputnevronene, med vektene som trenbare parametere mellom lagene. Treningsdata med for eksempel bilder og sanne etiketter sendes gjennom modellen, og inputdata brukes for å predikere output. Målet med treningen er å endre modellens funksjon for å gjøre den predikerte outputdataen mer nøyaktig i fremtidige iterasjoner. Outputdata sammenliknes derfor opp mot de sanne etikettene, og vektenes korrekthet noteres Dersom disse ikke er tilfredsstillende vil de justeres for å nærme seg det ønskede resultatet (Aggarwal, 2018). Dersom denne prosessen gjentas mange nok ganger vil en til slutt sitte igjen med en modell som gir mer nøyaktige prediksjoner.

For å utvikle en robust modell er det avgjørende å ha tilgang til store mengder variert treningsdata. En modell blir bedre på å klassifisere en sykkel når den eksponeres for en bred samling av bilder som viser forskjellige sykler fra ulike vinkler, med varierende grad av støy. Modellens evne til å generalisere, det vil si å beregne en funksjon for usett data basert på treningsdata med kjent input og output, er avgjørende. Dette konseptet er kjent som modellgeneralisering, og det spiller en sentral rolle i utviklingen av nevrale nettverk og dyplæring (Aggarwal, 2018).



Figur 2.4: *Arkitekturen bak en flerlags Perceptron med fem nevroner som input, to skjulte lag med tre nevroner hver, og en output (Aggarwal, 2018).*

Kunstige nevrale nettverk består av en rekke ulike parametere som deles inn i hyperparametere og trenbare parametere, basert på hvilken rolle de spiller i konstruksjonene og treningen av nettverket. Trenbare parametere er nettverkets vekter og biaser som trenes gjennom tilbakepropagering. Hyperparametere er parametere som oppdateres utenfor treningen av nettverket, og settes før treningen starer. Viktige hyperparametere inkluderer: antall fullt tilkoblede lag, aktiveringsfunksjoner, tapsfunksjon, læringsrate og optimaliseringsmetode.

2.5.1a Tapsfunksjon

Etter en iterasjon gjennom nettverket av noder og vekter vil en sitte igjen med et estimat estimert av algoritmen. Dette estimatet måles opp mot de sanne verdiene i datasettet, den kvantifiserte avstanden mellom disse er tapsfunksjonen (eng: loss function). Tapsfunksjonen er et mål på hvor godt modellen presterer på datasettet. Målet ved å trene en dyplæringsmodell er å minimere tapsfunksjonen, slik at estimatet etter hvert vil nærme seg de sanne verdiene og en får en mer presis modell. Det er flere typer tapfunksjoner tilgjengelige, og det er viktig å velge den som passer best til det spesifikke problemet man ønsker å løse. For binære klassifiseringsoppgaver er den vanligste tapsfunksjonen «binary cross-entropy» (BCE), som er gitt ved formelen

$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[y_i\, log(p_i) + (1 - y_i)\, log(1 - p_i)] \quad (2)$$

hvor N = antall instanser i datasettet, y_i = den sanne binære verdien i den i-ende instansen, p_i = den predikerte verdien i i-ende instans..

Fordeler med BCE er at det er en intuitiv funksjon som er lett å både tolke og beregne. I tillegg er den differensierbar, som er nødvendig om en vil benytte gradientbaserte optimaliseringsalgoritmer som for eksempel «gradient descent». En ulempe er at den er følsom for ubalanserte datasett, dersom de inneholder betydelige skjevheter i antallet observasjoner i hver klasse (Godoy, 2018).

2.5.1b Tilbakepropagering

Etter at det nevrale nettverket har kjørt gjennom en epoke, produserer den et estimat for oppgaven en ønsker å løse. Tapsfunksjonen beregner så hvor gode dette estimatet er i forhold til de sanne verdiene, og gir en feilverdi for nettverket. Denne feilverdien er hva som bestemmer hvordan vektene i nettverket skal oppdateres, slik at modellen i fremtiden blir mer nøyaktig. Tilbakepropagering en algoritme som beveger seg bakover i nettverket. På veien beregner den gradienten av feilverdiene, med hensyn på vektene og biaser i nettverket. Dette gjøres rekursivt ved å ta i bruk kjerneregelen fra kalkulus (Aggarwal, 2018). Verdien vi vil beregne er den negative gradienten av feilverdien. For å minimere feilverdiene justeres nettverkets vekter og bias. De justeres i retning av den negative gradienten av feilverdiene med hensyn til vektene og biaser.

Optimering

Dette oppnås ved å ta små skritt innenfor rommet definert av verdiene til nettverks parametere. Størrelsen på hvert skritt styres av en parameter kjent som læringsraten. Dette er en optimeringsalgoritme kjent som «Gradient Descent (GD)». Oppdateringen av vektene blir da

$$w := w + \Delta w, hvor\ \Delta w = -\eta\, J(w) \quad (3)$$

her er J tapsfunksjonen og η er læringsraten. GD beregner gradienten basert på hele settet med treningsdata, hvilket kan gjøre algoritmen mindre effektiv på store datasett.

En videreutvikling av Gradient Descent er stokastisk gradient descent (SGD). I motsetning til GD som beregner gradienten av hele datasettet beregner SGD gradienten til tapsfunksjonen for små partier av treningseksempler om gangen, hvilket reduserer den nødvendige datamaskinkapasiteten betraktelig sammenliknet med vanlig GD (Rahul_Roy, 2023).

«Adaptive Moment Estimation (Adam)» er blitt en de-facto-metode for optimalisering av vektoppdateringer. Adam kombinerer fordelene ved både GD og SGD ved å tilpasse læringsraten til problemet. I tillegg bruker algoritmen momentum. Det hjelper algoritmen med å unnslippe lokale minimum, og med å konvergere raskere. Dette gjør at Adam er en meget sterk optimeringsalgoritme, spesielt ved store datasett (Ajagekar, 2021).

2.5.1c Aktiveringsfunksjon

Enhver node i et nevralt nettverk har en aktiveringsfunksjon (eng: activation function) som bestemmer output, gitt et sett av input-data. Aktiveringsfunksjoner kan tilføre ikke-linearitet til nettverket, slik at det kan løse komplekse mønstre og forhold i datasettet. Uten aktiveringsfunksjoner ville et nevralt nettverk oppføre seg som en svakt lineær regresjonsmodell, og for eksempel bildeklassifisering ville ikke vært en mulig (Sharma et al., 2017). Det er en rekke ulike aktiveringsfunksjoner tilgjengelige, og valg av riktig funksjon kan ha svært stor betydning for både treningsprosessen og ytelsen til nettverket. Derfor er det viktig å vite hva slags aktiveringsfunksjoner som passer ulike problemer og nettverksstrukturer.

I aktiveringsfunksjonen mottas den vektede summen av inputverdiene, og det produseres et resultat basert på formelen til valgt funksjon. Dette resultatet kan for eksempel være en terskelverdi (eks. 0 eller 1), en sigmoid (en verdi mellom 0 og 1), eller en verdi mellom 0 og maks (eks. ReLu).

Sigmoid er en aktiveringsfunksjon som fungerer spesielt godt på binær klassifisering. Den konverterer inputverdier til en verdi mellom 0 og 1 som representerer sannsynligheten for at den tilhører en av de to klassene. Sigmoidfunksjonen er gitt ved

$$\phi(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

hvor x er inputverdien til funksjonen. Sigmoid har form som en s-kurve og skjærer y-aksen i $x = 0$.

Et problem som kan oppstå under treningen av nevrale nettverk, er at gradientene gradvis forsvinner. Dette skjer når de partiellderiverte er nær null. Gradientene blir da gradvis mindre etter hvert som de tilbakepropageres gjennom nettverket. Dette problemet kalles forsvinnende gradienter (eng: vanishing gradients). Situasjonen vil forverres dersom aktiveringsfunksjonen også har små gradienter, hvilket er tilfelle i for eksempel sigmoid. En måte å håndtere dette problemet på er ved å ta i bruk aktiveringsfunksjonen «Rectified Linear Unit (ReLu)». Den tar x som input, og returnerer x dersom den er positiv og 0 om den skulle være negativ. Dette betyr at den deriverte av funksjonen vil være konstant for positive input-verdier ved gradient = 1 (Raschka & Mirjalili, 2019). ReLu er gitt ved

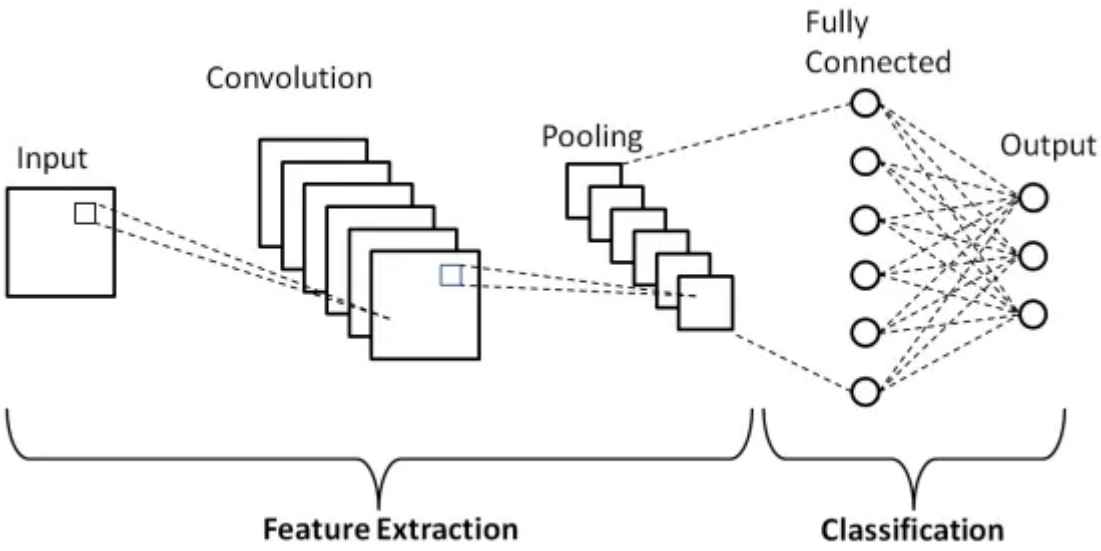
$$\phi(x) = \max(0, x) \tag{5}$$

hvor x er inputverdien til funksjonen.

2.5.1d Konvolusjonelle nevrale nettverk (CNN)

Konvolusjonelle nevrale nettverk (CNN) kan sies å være en videreutvikling av ANN hvor fokuset ligger på kunstig syn, noe som gjør CNN spesielt godt egnet til bildegjenkjenning og klassifisering. Inspirasjonen til CNN er hvordan menneskets visuelle cortex fungerer, med hierarkisk inndelte lag av nevroner som prosesserer informasjon for å gjenkjenne objekter (Raschka & Mirjalili, 2019). CNN består også av input-verdier, flere skjulte lag og output-verdier. Det som er spesielt med CNN, er at deres skjulte lag også består av såkalte konvolusjonslag som trekker ut egenskaper som kanter, teksturer og former. I dag kjennetegnes CNN av sin tilpasningsdyktighet og evne til å automatisk lære egenskaper fra input. Allikevel var det først i 2012 at konvolusjonelle nevrale nettverk ble de første store suksessene innen

dyplæring, da modellen AlexNet (Krizhevsky et al., 2012) vant ImageNets årlige bildeklassifiseringskonkurranse. Siden den gang er konvolusjonelle nevrale nettverk ansett som så dominerende at bortimot alle deltakere i konkurransen har benyttet CNN siden (Aggarwal, 2018).



Figur 2.5: *CNN med konvolusjonslag som henter egenskaper fra bildet som input. Dette sendes så gjennom et kunstig nevralt nettverk (Balaji, 2020).*

CNN består av to deler. Den første delen trekker ut egenskaper i bilde, og består av konvolusjonelle lag. Den andre delen er den som står for selve klassifiseringen. Denne består av fullt tilkoblede lag som lærer hvilke egenskaper i bilde som tilhører de ulike klassene.

Konvolusjonslagene i et CNN benytter filtre til å hente ut egenskapene fra bilde. Dette kan være for eksempel kanter, hjørner og farger. Selve konvolusjonen er operasjonen hvor et sett med vekter prikkes med ulike deler av bilde. Output vil være en matrise kalt egenskapsmatrisen (eng: feature map). Etter hver konvolusjon følger en pooling-operasjon, som drastisk reduserer den romlige størrelsen på egenskapsmatrisen (Aggarwal, 2018). Som sett i figur 2.5, gjennomføres deretter delen av nettverket som står for selve klassifiseringen, nemlig fullt tilkoblede lag med aktiveringsfunksjoner, samt ulike regulariseringsmetoder for å unngå overtilpasning. Output-laget er det siste som skjer i nettverket og består av like mange nevroner som det er klasser i datasettet.

2.5.1e Overføringslære

For å trene et nevralt nettverk for å oppnå høy ytelse er en avhengig av store treningsdatasett med markerte data. Innsamling av slike data kan av ulike årsaker være vanskelig å få til, og kan være utfordringer som at innsamlingen og markeringen av dataen er dyr, forekomsten av ønsket data sjelden eller at dataen er vanskelig å samle inn. I slike tilfeller kan en ta i bruk overføringslæring (eng: transfer learning). Det er en tilnærming hvor egenskaper og representasjoner lært fra én oppgave, overføres til en annen. Overføringslæring lar nettverket ta avgjørelser basert på erfaringer fra tidligere relaterte oppgaver (Weiss et al., 2016). Dersom en person med lang erfaring som baseballspiller, og en uten særlig erfaring med idrett, begge ønsker å bli gode i golf. Da kan en med rimelig sikkerhet kunne anta at baseballspilleren raskere vil mestre golf, ved å overføre kunnskapen fra en liknende idrett til den nye. Fremgangsmåten og logikken ved overføringslæring vil være den samme (Pan & Yang, 2009). I dag er det en rekke store datalagringsområder tilgjengelig for nedlastning og bruk til overføringslæring. Oppgaver som ofte benytter overføringslæring er bildegjenkjenning, naturlig språkbehandling og lydbehandling.

Overføringslæremodeller til konvolusjonelle nevrale nettverk trenes ofte på store datasett med bilder. Et eksempel på et slikt datasett er Imagenet (Deng et al., 2009), en database med over 14 millioner bilder i mer enn 20,000 kategorier, alle notert med innholdet i bildene. I treningen av overføringslæremodeller er det derimot vanlig å benytte kun en tilfeldig delmengde av disse, avhengig av prosjektets behov.

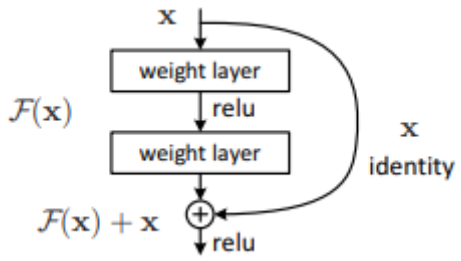
Vanlig bruk av overføringslæremodeller innebærer som regel (Chollet, 2020):

1. Velg ut noen lag fra modellen og frys dem for å unngå å ødelegge informasjonen de holder på under treningen av nettverket.
2. Legg til nye lag som skal trenes over de fryste lagene. Disse vil lære å bruke de overførte egenskapene til å lage nye estimater.
3. Tren de nye lagene på eget datasettet.

For å skvise ut den maksimale ytelsen til modellen ovenfor, kan en finjustere modellen. Da avfryses hele eller deler av modellen, før den trenes på ny data med veldig lav læringsrate. Dette kan forbedre modellen ved at de overførte egenskapene gradvis vil tilpasses de nye dataene (Chollet, 2020). Finjustering er en fin måte å kompensere for manglende data, dersom en benytter mindre datasett i treningen.

ResNetV2

På midten av 2010-tallet var CNN på fremmarsj innen bilde- og talegjenkjenning, grunnet sine effektive løsninger på komplekse problemer. Modellene ble stadig dypere, med flere lag og trenbare vekter. Disse dype nevrale nettverkene ble etter hvert vanskelige å trene grunnet problemer som forsvinnende og eksploderende gradienter. Derfor introduserte K. He et al. residuale nettverk (ResNet) i 2016 (He et al., 2016). Metoden tar i bruk residualblokker med skip-koblinger som tillater bedre flyt av informasjon gjennom nettverket, som motvirker problemene med forsvinnende og eksploderende gradienter. Skip-koblingen i figur 2.6 hjelper input-info å nå senere lag i nettverket uten å passere gjennom alle transformasjonene. I treningen av nettverket vil vektene tilpasses også skip-koblingene, i tillegg til de vanlige lagene. Dette vil gjøre nettverket best mulig rustet for å utnytte informasjonen hvert lag mates fra skip-koblingene, for å igjen forbedre ytelsen.



Figur 2.6: *Residualblokk med skip-kobling fra ResNet. F(x) er transformasjonene som skjer i de to konvolusjonelle lagene, x er inngangen som har hoppet over disse lagene (He et al., 2016)*

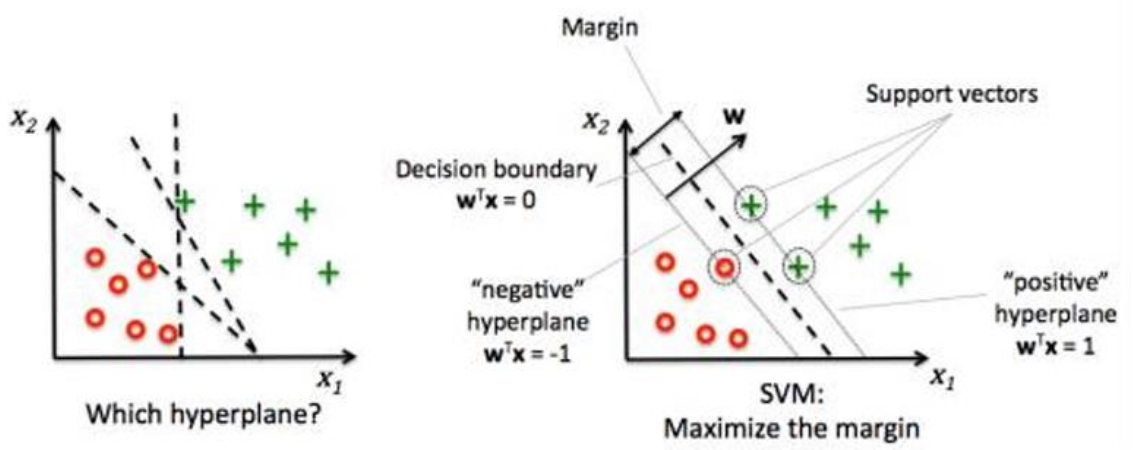
ResNetV2 er en videreutvikling av den originale ResNet. Her er strukturen i residualblokkene endret for å øke ytelsen til modellen. I ResNetV2 er batchnormalisering og ReLu-aktivering flyttet så de kommer før konvolusjonslagene, i motsetning til originale ResNet hvor batchnormalisering og ReLu-aktivering kom etter konvolusjonen. Det gjør at gradientene lettere kommer seg gjennom nettverket under trening i ResNetV2. Dette gjør at ResNetV2 tilbyr bedre treningshastighet og konvergens, og gir generelt bedre ytelse enn originale ResNet (Sampath et al., 2023).

Både ResNet og ResNetV2 kan trenes på forhånd og benyttes i overføringslæremodeller, men også her gir ResNetV2 hakket bedre ytelse enn originalen. Begge ligger tilgjengelige i Keras, forhånstrente på Imagenets store bildedatasett (*Keras Applications*).

2.5.2 Support vector machines

Support vector machines (SVM) er en maskinlæringsalgoritme av veiledet læring (eng: supervised learning) for dataklassifisering. I klassisk maskinlæring er SVM ansett som en av de bedre klassifiseringsalgoritmene, ettersom den er relativt enkel å forstå, implementere, bruke og tolke, i tillegg til at den fungerer svært godt på datasett med færre treningseksempler.

Målet ved SVM er å definere det hyperplanet i datasettet som best skiller dataene fra ulike klasser basert på klassenes egenskaper. Datapunktene som ligger ytterst i hver klasse kalles støttevektorer (eng: support vectors). Hyperplanet ønsker å maksimere marginen mellom disse støttevektorene (Raschka, 2015), slik en kan se i figur 2.7. Dette er med på å generalisere modellen og redusere overtilpasning. Avstanden mellom klassene og dette hyperplanet kalles for marginer, og er vinkelrett mellom det separerende hyperplanet og støttevektorene.



Figur 2.7: *SVM finner støttevektorer og maksimerer marginen mellom dem for så å skille dem med er hyperplan (Raschka & Mirjalili, 2017).*

Kjerner

I sin opprinnelige form er SVM en lineær klassifisør som søker etter lineært separable hyperplan. I virkelige oppgaver er det sjeldent en ser slik linearitet. For å kunne klassifisere mer komplekse klassifiseringsoppgaver benyttes kjernefunksjoner for å introdusere ikke-linearitet til støttevektormaskiner. Kjernefunksjoner projiserer dataen til et høyere dimensjonalt rom, hvor det er mulig å separere klasser med et lineært hyperplan. Denne projeksjonen hever dataen til et hypotetisk rom av høyere dimensjon, men trengs ikke eksplisitt å gjennomføres takket være teknikken «kjernetrikset» (eng: the kernel trick) (Raschka, 2015).

Blant de mest bruke kjernefunksjonene finner vi i dag radiell basis funksjon (eng: Radial basis function) (RBF). RBF-funksjonen tar formen til en Gaussisk fordeling, beregner likheten mellom punkter, x_i og x_j, basert på avstanden mellom dem, og projiserer dem i uendelig dimensjonalt rom. Denne Gaussiske funksjonen er gitt ved

$$K(x_i, x_j) = e^{(-\gamma \|x_i - x_j\|^2)} \tag{6}$$

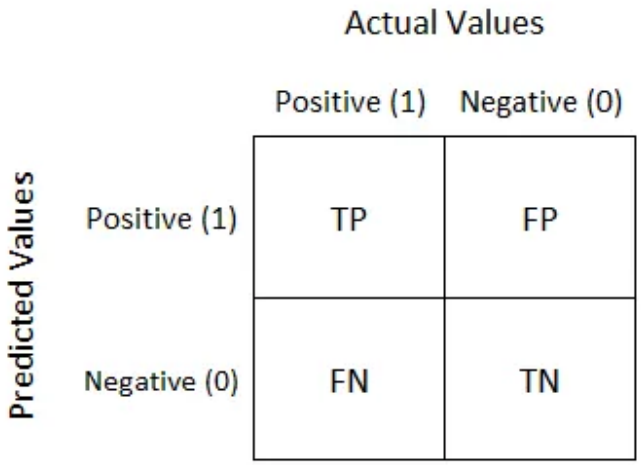
hvor γ er en valgt parameter for styrer bredden på fordelingen, og $\|x_i - x_j\|^2$ er den kvadrerte euklidske avstanden mellom punktene (Raschka, 2015). Funksjonen beregner altså en likhetsscore mellom to punkter basert på avstanden mellom dem. Slik får punkter som ligger nærmere hverandre i det opprinnelige datarommet ha en høyere vekt i valg av beslutningsgrense. Dette gjør at SVM-modellen kan ta hensyn til komplekse datamønstre når beslutningsgrensen skal bestemmes. Parameteren γ bestemmer bredden på den Gaussiske fordelingen, og må justeres riktig for å oppnå en god ytelse med RBF-funksjonen (Raschka, 2015).

C-parameteren i en SVM viser til hva slags grad av feilklassifisering som tillates. En lav C-verdi tillater større grad av feilklassifisering, hvilket vil gi en mindre kompleks modell med bredere marginer. Høy C-verdi vil være strenger på feilklassifiseringer og øke modellens kompleksitet med smalere marginer mellom klassene. Valg av en god C-verdi er derfor viktig for å oppnå en kompleks modell som allikevel er generell nok til å anvendes på ekte data.

2.5.3 Vurderingsmetoder

Vurderingsmetoder (eng: scoring metrics) er en viktig del av å trene et nevralt nettverk er å måle modellens ytelse og nøyaktighet. Det finnes ulike måter å beregne ytelsen til et nettverk på. Noen metoder fokuserer på det absolutte forholdet mellom riktige og ikke riktige klassifiseringer, mens andre ser nærmere på hva slags feil som er gjort i de gale klassifiseringene. Felles for alle er at de sammenlikner modellens estimater mot de samme verdiene. De vanligste metodene for å evaluere en modells ytelse tar i bruk en forvirringsmatrise (eng: confusion matrix) som viser hvordan modellens estimater er fordelt basert på hva slags

feil den har gjort. I et binært klassifiseringsproblem vil forvirringsmatrisen ha fire kategorier, basert på estimatene som produseres. Disse kategoriene er: sann negativ (NP), med estimater som korrekt klassifiseres som negativ; falsk positiv (FP) som feilaktig klassifiseres som positiv; falsk negativ (FN), dersom den feilaktig klassifiseres som negativ; sann positiv (TP), som er korrekte klassifiserte positive estimater (Raschka & Mirjalili, 2019).



Figur 2.8: *Forvirringsmatrise for binær klassifisering med de fire kategoriene dataen blir sortert inn i (Narkhede, 2018).*

Figur 2.8 viser en forvirringsmatrise en kan bruke til å beregne modellens feil (ERR) og nøyaktighet (ACC), som begge gir oss generell informasjon om hvor mange feilklassifiseringer modellen har produsert. Feil er gitt ved

$$ERR = \frac{FP + FN}{FP + FN + TP + TN} \tag{7}$$

Nøyaktigheten kan beregnes direkte fra feilen ved

$$ACC = \frac{TP + TN}{FP + FN + TP + TN} = 1 - ERR \tag{8}$$

Disse sier dog ikke noe om hva slags feil som er begått. Derfor tas det i bruk mål for presisjon (PRE) som måler hvor mange av de positive tilfellene som korrekt klassifisert, og sensitivitet (REC) som måler hvor mange av de sanne positive verdiene som ble korrekt identifisert av modellen.

$$PRE = \frac{TP}{TP + FP}, \qquad REC = \frac{TP}{TP + FN} \tag{9, 10}$$

F1-score

For å kunne gi et helhetlig bilde av hvordan modellen presterer benyttes ofte F1-score. Her benyttes et harmonisk gjennomsnitt av presisjon og sensitivitet for å gi en balansert vurdering av estimatene (Raschka & Mirjalili, 2019). F1-scoren beregnes ved

$$F1 = 2 \cdot \frac{PRE \cdot REC}{PRE + REC} \tag{11}$$

og gis som en skalar fra 0 til 1, hvor en modell med god ytelse vil nærme seg 1. Da vil modellen oppnå god nøyaktighet med god dekning for identifisering av positive tilfeller, samtidig som den unngår falske klassifiseringer. Det som derimot kan oppstå ved bruk av F1-score er at den ved ubalanserte datasett kan gi inntrykk av at resultatene er bedre enn de burde være. Metoden tar ikke i bruk sanne negative (TN), noe vil gi ubalanserte resultater hvor resultatet avhenger av hva man klassifiserer som positive og negative.

Matthews korrelasjonskoeffisient

I forskningsartikkelen «The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation» skriver Chicco og Jurman at Matthews korrelasjonskoeffisient (MCC) er å foretrekke over F1-score i alle binære klassifiseringsoppgaver (Chicco & Jurman, 2020). MCC tar nemlig i bruk alle elementer i forvirringsmatrisen, og er derfor en mer egnet måte å evaluere modeller i klassifiseringsproblemer hvor datasett kan være ubalanserte. Verdiene fra MCC ligger mellom -1 og 1, hvor 1 indikerer perfekt prediksjon, 0 indikerer tilfeldige prediksjoner og -1 indikerer fullstendig feil prediksjon. MCC gis ved formelen

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \quad (12)$$

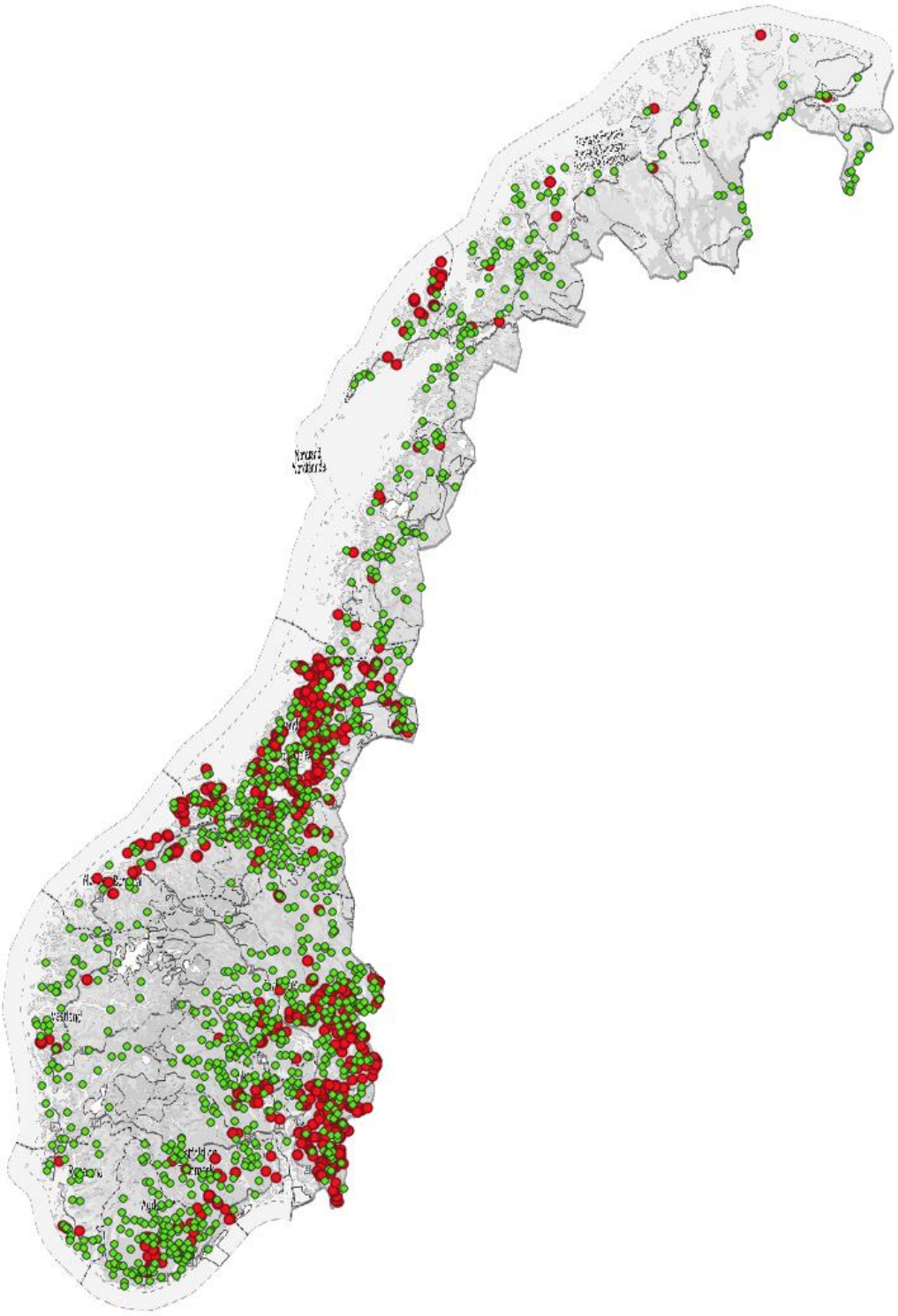
I tilfeller hvor hele rader eller kolonner i forvirringsmatrisen er null, vil telleren også bli null og MCC vil være udefinert. For å allikevel kunne produsere en meningsfull MCC-verdi kan det benyttes ulike teknikker fra kalkulus. En kan for eksempel bytte ut null-verdiene med tilfeldige små verdier (Chicco & Jurman, 2020).

3 Metode

Dette kapittelet går inn i metodikken som er brukt i denne studien. Datasettet som ble brukt i treningen av modellene vil presenteres. Deretter følger en gjennomgang av de to modellene som er utviklet med tilhørende forprosessering og filtrering, i tillegg til markering av dataene. Videre beskrives hva slags programvare og maskinvare som er benyttet i studie, før kapittelet avsluttes med en forklaring av hvordan kunstig intelligens er brukt i oppgaven.

3.1 Område og datagrunnlag

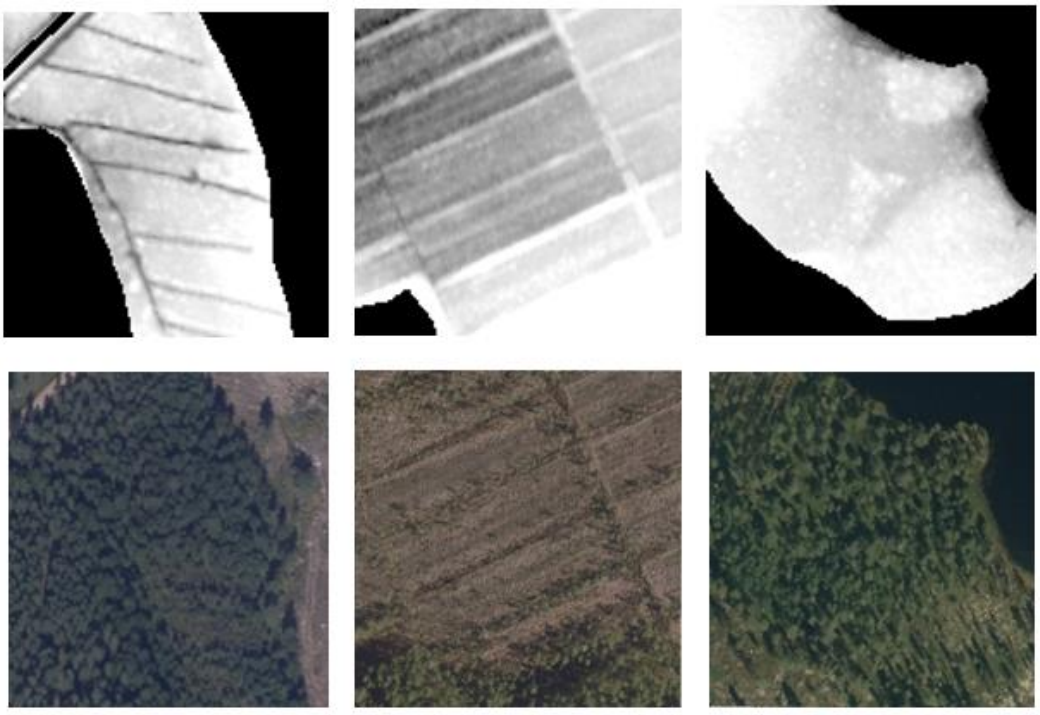
Dataen som benyttes er levert av NIBIO og består av til sammen 2656 bilder av norsk myr, hvor 1328 er grøftet og 1328 ikke er grøftet. Bildene i datasettet er hentet fra myrer i hele Norge for å kunne lage en generell klassifiseringsmodell som skal kunne brukes i hele landet. Figur 3.1 viser hvor datasettet er hentet fra. Hvert eksempel er et bilde på 128x128 piksler, og er utdrag av kartverkets digitale terrengmodell med 1x1m oppløsning.



Figur 3.1: *Viser områdene hvor datasettet som brukt i denne oppgaven er hentet fra. Rød er bilder av grøftet myr, grønn er ikke grøftet.*

Pikselverdiene i bildene er relative, hvor laveste punkt i bildet er 0, og resterende pikselverdier er høydene over dette punktet. Siden ønsket er at modellen kun skal klassifisere områder vi vet er myr, klippes alt som ikke er myr ut av bildene. De gjenværende områdene med NoData-verdier er fylt med -99 for å indikere at modellen ikke skal lære uønskede sammenhenger og mønstre fra disse områdene. Bilene er på GeoTIFF-format, som er en forlengelse av TIFF-formatet. Formatet tillater lagring av romlig metadata, og er derfor et populært filformat for lagring av geodata.

For å få en dypere forståelse av oppgaven og datasettet som benyttes, var første skritt å utforske den tilgjengelige dataen. Bildene kommer som 8-bit gråskala rastere hvor pikselverdier varierer fra 0 (sort) til 255 (hvit). Verdiene imellom presenteres som ulike nyanser av grå. 0 gis til laveste punkt i bildet og piksler blir lysere etter hvert som høydeverdien øker. Høyeste punkt er hvitt. Siden bildene er gråskala, kommer de også med kun én fargekanal (intensitet) i motsetning til RGB som har tre (red, grønn og blå).



Figur 3.2: *Tre typer områder som utgjør datasettet. En dreneringsgrøft, et torvuttak og en intakt myr. Bildene under viser flyfoto av de samme områdene.*

Det er to typer grøftet myr som er avbildet datasettet, som sett i figur 3.2. Til venstre ser en grøfting i forbindelse med drenering av myrområder, gjort for å kunne etablere skog, beitemark, dyrkamark eller områder av annen interesse. Denne typen grøfting kjennetegnes av omfattende systemer med relativt dype, tynne grøfter som forgrener seg i myra. Den andre typen grøfting kan ses midt i figur 3.2 og består av grunnere, tykke bånd. Her kan vi se tydelige spor etter grøfting og harving av myra, hvilket tyder på at det har foregått, eller fortsatt foregår, torvuttak i området. Til høyre i figuren er terrengmodellen av en ikke grøftet myr.

3.2 CNN-modell

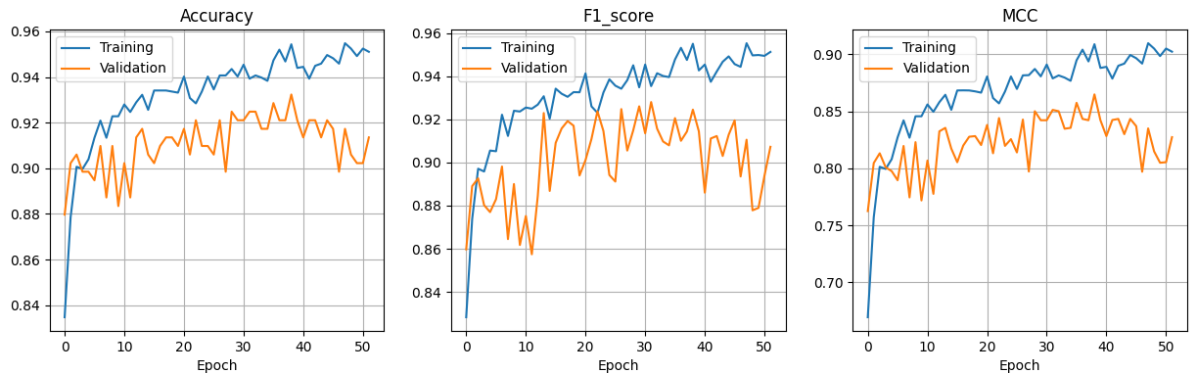
3.2.1 Forprosessering

Datasettet som ble levert av NIBIO var allerede godt forprosessert. Bildene kom i TIFF-format, var klippet til samme størrelse på 128x128 piksler, og uviktige områder var fjernet. For å unngå store skjevheter i bildene ble områdene med verdien -99 endret til 0. Fremgangsmåten i forprosesseringen kan ses i figur 3.3.

Figur 3.3: *Steg i forprosesseringen for CNN-modellen.*

3.2.2 Laplace-filtrering

For å framheve grøftemønsteret som skal brukes for å klassifisere bildene, ble Laplace-filtrering benyttet. Tidligere trening av nettverket uten Laplace-filtrering, som kan ses i figur 3.4, viste tydelige tegn til overtilpassing tidlig i treningsprosessen. Det ble derfor testet med Laplace-filter, nettopp for å unngå at modellen skulle lære mønstre og sammenhenger fra områder utenfor de aktuelle grøftene, samtidig som grøftene fremheves og gjøres mer markante i bildet. Datasettet ble evaluert med ulike kjernevarianter ved å kjøre samme kode mange ganger med ulike kjerner. Resultatet ble kjernen fra likning (1). Dette filteret ble brukt på både trening-, validering- og testdatasettene.



Figur 3.4: *Treningshistorikken til CNN trent uten Laplace-filtrering.*

3.2.3 Merking av data

For å kunne trene og evaluere modellen var det viktig å merke bildene med riktig klasse. Siden oppgaven er en binær klassifisering ble bildene for grøftet myr merket som «1», og ikke grøftet myr merket som «0». Bildene som ble benyttet ble importert fra mappene «grofta» og «ikkegrofta», og markeringene ble følgende gjort ut ifra hvilken mappe de ble importert fra. For å unngå skjevheter og at modellen lærte uønskede mønstre baser på rekkefølgen til treningsdataen, ble dataen tilfeldig stokket om for å tillate læring av mer generelle mønstre. Parene av bilder og markeringer ble stokket som en enhet for å sikre at de fortsatt stemte overens.

3.2.4 Oppdeling av datasettet

For å utvikle en god generalisert modell, er en viktig del av prosessen å holde tilbake noe av dataen fra treningen for å unngå overtilpasning. For eksempel vil en modell kunne lære spesifikk mønstre og gi unøyaktige klassifiseringer på usett data, dersom en bruker kun treningsdata. Ved oppdeling av datasettet vil den usette dataen brukes til validering og testing av modellen.

Datasettet ble delt inn i tre hoveddeler: trening, validering og test. I oppdelingen ble scikit-learns (Pedregosa et al., 2011) funksjon «train_test_split» benyttet med fordelingen 80 % til trening, 10 % til validering og 10 % til testdatasett. Valget av 80 % av dataen til trening kom av at ettersom hele datasettet består av kun 2656 bilder, ble det ansett som viktig å få brukt så mye om mulig av datasettet til trening. Oppdelingen betyr at 2124 bilder brukt i treningen, mens det ble brukt 266 i valideringen og 266 til testing av modellen.

3.2.5 Normalisering

For å sikre at all data er skalert likt, samt sørge for at variasjonen i intensiteten mellom bildene er så lav som mulig, normalisertes alle bildene i datasettet. Dette ble gjort for transformasjonen

$$x_{skalert} = \frac{x - \mu}{\sigma} \tag{13}$$

hvor μ er bildets gjennomsnittlige intensitet og σ er standardavviket. Etter normaliseringen er målet at alle bildene vil ha $\mu = 0$ og $\sigma = 1$. Målet med normaliseringen var å gjøre det lettere som modellen å finne og generalisere mønstre, som igjen vil kunne gi en modell som produserer pålitelige resultater.

3.2.6 Omformatering av bildene

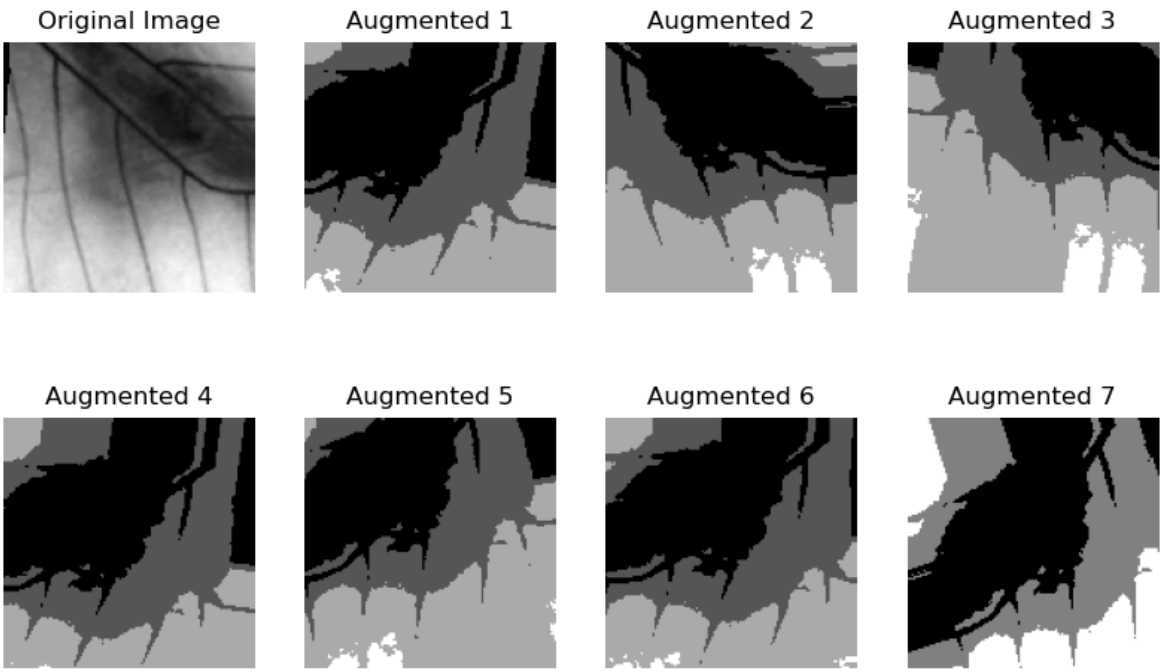
Den forhåndstrente modellen fra Keras, ResNet50V2, prosesserer kun bilder med nøyaktig tre inngangskanaler. Ettersom modellen i denne oppgaven benyttet gråskala bilder (en kanal), var det nødvendig å omformatere bildene slik at de tilsynelatende bestod av tre kanaler. Dette ble gjort ved å duplisere gråskalakanalen tre ganger langs kanalaksen, resulterende i bilder med formen (128, 128, 3). De ferdigprosesserte bildenes dimensjoner finnes i tabell 3.1.

Tabell 3.1: *Dimensjoner på trenings-, validerings- og testdata.*

Datasett	x data	y data
Treningssett	**(2124, 128, 128, 3)**	**(2124,)**
Valideringssett	**(266, 128, 128, 3)**	**(266,)**
Testsett	**(266, 128, 128, 3)**	**(266,)**

3.2.7 Dataforsterkning

Som en konsekvens av størrelsen på datasettet ble det besluttet å ta i bruk dataforsterkning (eng: data augmentation). Det er en metode som øker antallet variasjoner av treningsbildene ved å strekke, flippe, rotere og/eller skifte treningsbildene. Klassen vil sørge for at treningsbildene er litt forskjellige for hver epoke i treningsprosessen. I denne oppgaven ble Keras' «ImageDataGenerator» benyttet til dataforsterkning. Treningsbildene ble transformert innen følgende rammer: rotasjonsområde $\pm 40°$, vertikal skiftningsgrad på opptil 20 % av total høyde, horisontal skiftningsgrad på inntil 20 % av total bredde og tillatt tilfeldig horisontal bildeflipping. Disse transformasjonene ble gjort tilfeldig og sørget for at modellen aldri så et helt likt bilde to ganger. Effekten av forsterkningen vises i figur 3.5.



Figur 3.5: *Effekten av bildeforsterkning. Originalbilde er oppe til venstre, resten er transformerte versjoner av dette.*

3.2.8 Nettverksstruktur

I startfasen av prosjektet ble flere nettverksstrukturer og metoder vurdert som interessante for oppgaven. Dype residuale nettverk og inception-nettverk ble begge vurdert ettersom begge er effektive varianter av konvolusjonelle nevrale nettverk, men etter å ha vurdert størrelsen på treningsdataen som for lite for fullstendig opplæring av disse, havnet valget på overføringslæring. Overføringslæring krevde også betydelig mindre beregningskraft enn andre alternativer.

Etter testing med flere overføringslæringsmodeller landet valget på ResNet50V2 (Chollet, 2020), hvor tallet 50 refererer til antall konvolusjonslag i modellarkitekturen. Modellen kunne tilby god ytelse, uten å bruke alt for lagt tid i treningen. Oppbygningen av overføringslæringsmodellen tok utgangspunkt i eksempelkoden fra hjemmesiden til Keras (Keras), og ble utviklet ved å ta i bruk TensorFlows rammeverk. TensorFlow er et kraftig rammeverk av verktøy, biblioteker og andre ressurser, utviklet av Google AI (Google AI. (n.d.)) for å bygge opp og trene dyplæringsmodeller.

For å skvise så høy ytelse ut av ResNet50V2 som mulig ble nettverket finjustert. Prosessen krever en lav læringsrate for å ikke overdøve modellens tidligere lærte vekter. Her ble læringsraten satt til 0,0001. Dette gjorde at den forhåndslærte modellens egenskaper trinnvis ble tilpasset til datasettet brukt i denne oppgaven. Dette er en effektiv måte å øke ytelsen til nettverket.

3.2.9 Hyperparameter tuning

Under treningen av nettverket oppsto det problemer knyttet til valg av hyperparametere. Valg av gode hyperparametere er viktig for å oppnå best mulig ytelse av nettverket. God kunnskap om nettverket og dets metode kan gi en viss intuisjon når hyperparametere skal velges, men det er likevel en god idé å teste ulike verdier for å finne de optimale parameterne. For å slippe å kjøre koden med ulike parametere og notere de ulike ytelsene manuelt, ble programbiblioteket Keras Tuner (O'Malley et al., 2019) benyttet. Under tuningen var det verdien for F1-scoren på valideringssettet som avgjorde om verdiene for hyperparameterne var gode nok. Det var modellen med høyest F1-score som ble valgt ut som den beste. Oversikt over testede og valgte verdier kan ses i tabell 3.2.

Tabell 3.2: *Oversikt over hyperparameteroptimaliseringen med testverdi og valgte parametere.*

Hyperparameter	**Testverdier**	**Valgte verdier**
# nevroner i dense-lag	32 → 512	288
Dropout-rate	0,1 → 0,5	0,5
Læringsrate	0,001 → 0,01	0,0018

3.2.10 Visualisering av resultater

For visualisering av treningsresultater ble det tatt utgangspunkt i filen «Visualization», brukt i de obligatoriske oppgavene i DAT300 ved NMBU. Funksjonen som ble benyttet heter «plot_training_history» og visualiserer treningshistorikken ved å plotte utviklingen av nøyaktighet, F1-score og MCC gjennom alle epokene, for både trenings- og valideringsdataen. Funksjonen tar treningshistorikkobjektet fra tf.keras.model.fit() og en liste over ønskede beregninger som input. Output er plottet over treningshistorikken.

Da treningen var gjennomført, ble det usette testdatasettet kjørt gjennom modellen og ukorrekte prediksjoner ble plottet for å vise hva slags bilder modellen ikke klarte å klassifisere korrekt. Målet her var å se om det var noen sammenheng mellom de ukorrekte prediksjonene.

3.3 SVM

Datasettet ble merket på samme måte som i CNN-modellen, med ikke grøftet merket som 0 og grøftet som 1.

For å fremheve grøftelinjene i datasettet ble Canny-kantdeteksjon benyttet. Canny ble ansett som en nyttig funksjon å bruke for å fremheve mer informativ informasjon, som former, kanter og konturer. Funksjonen reduserer også bakgrunnsstøy, hvilket kan være lurt når det er linjene i bildet en ønsker å se på.

SVM forventer å få input-data på vektorform, ikke i matriser slik som CNN. Derfor ble bildene flatet ut i vektorer hvor hvert element representerte intensiteten i en piksel.

Datasettet ble også her delt opp, men kun i trening og test. Valideringsdata ble utelatt da det ble foretatt en kryssvalidering istedenfor. 80 % av dataen, altså 2124 bilder, ble satt av til trening, og 20 %, dvs. 532 bilder, til test.

Alle bildene ble normalisert for å sikre av at de var skalert likt. Her ble ScikitLearns StandardScaler (Pedregosa et al., 2011) benyttet. SVM er sensitiv for dataens målestokk. Derfor hjelper normalisering med å sikre at modellen ikke forstyrres av funksjoner med høyere målestokk.

I SVM er det også knyttet problemer til valg av hyperparametere. Det er derfor, også i denne modellen, foretatt en hyperparameter tuning for å oppnå god ytelse av modellen. RandomSearchCV fra SciKitLearn ble benyttet for å finjustere C-parameteren og γ. I tillegg er RandomSearchCV relativt kostnadseffektiv da den undersøker et valgt antall med tilfeldig valgte subsett av parametere. Her er det testet 16 ulike subsett. Parameterrommet kan ses i tabell 3.3.

Tabell 3.3: *Hyperparameter tuning SVM med testverdi og valgte parametere.*

Hyperparameter	Testverdier
C-parameter	0.1, 1, 10, 100
γ	scale, auto, 0.001, 0.0001
Kjerne	RBF, poly, linear
Klassevekt	Balansert, ingen

Når γ er satt til «auto», blir verdien av γ beregnet basert på antall egenskaper i datasettet. Dette gjør at γ ignorerer datasettets skala eller varians og gir en enklere, men mindre fleksibel γ. Når γ er satt til «scale», blir verdien av γ beregnet av egenskapenes skala (varians), noe som ofte er foretrukket da det automatisk tilpasser γ til datasettets variabilitet. For å finne modellen med de beste hyperparameterne ble det foretatt en 5-fold-kryssvalidering med 16 forskjellige sammensetninger av hyperparametere.

3.4 Programvare og maskinvare

All programmering i denne oppgaven er gjort i Python (Van Rossum & Drake, 1995) versjon 3.10.13, i Spyder IDE 5.4.5 (Raybaut, 2009). For visualisering av datasettet ble Matplotlib (Hunter, 2007) og Pandas (McKinney, 2010) benyttet, mens NumPy (Harris et al., 2020) ble brukt til utregninger og matriseoperasjoner. For å implementere dyplæringsmodellen ble Tensorflows 2.15.1 rammeverk (Abadi et al., 2015) tatt i bruk. En komplett tabell over programvare og python-moduler finnes i tabell 0.2 i vedlegg A.

I denne studien er alle utregninger utført på en stasjonær pc med GeForce RTX 3060 12 GB grafikkort, 16 GB RAM og Intel Core i7-9700F prosessor. Maskinen kjører på operativsystemet 64-bit Windows 10. Med et dedikert grafikkort som er optimalisert for parallell databehandling fungerer maskinvaren generelt effektivt på oppgaver med store data og mange beregninger. Datamaskinen fullfører alle nødvendige beregninger innenfor en akseptabel tidsramme, og leverer gode resultater. Selv om en enda kraftigere datamaskin kunne ha fremskyndet prosessen, spesielt under hyperparameteroptimaliseringen, som krever mange treningsløp, ble alle nødvendige oppgaver fullført ved å sette av tilstrekkelig tid.

3.5 Bruk av kunstig intelligens

Det er i denne oppgaven benyttet kunstig intelligens til litteratursøk, idémyldring og feilsøking i kodedelen av oppgaven. Til litteratursøket er Scholar GPT (OpenAI & SiderAI, 2024), en forlengelse av ChatGPT (OpenAI, 2024), benyttet.

Scholar GPT tok input som for eksempel «Find lastest research about CNN» eller «Terrain analysis with CNN research» og linket, med direkte tilgang til blant annet Google Scholar, til relevante artikler. Scholar GPT ble også brukt til rådføring til kildehenvisning når det ikke var åpenbar hvordan det skulle gjøres, her ble svarene vurdert og dobbeltsjekket før det ble tatt med videre. Før det ble satt i gang store delprosjekter i oppgaven, ble ChatGPT spurt om idéen var verdt å teste ut eller ikke. Det ble for eksempel spurt om filtrering som Laplace og Canny kunne ha noen positiv effekt på CNN-modellen. Avhengig av svar, gjorde slike spørsmål det lettere å avgjøre om større delprosjekter burde testes eller ikke. I feilsøkingen i kodedelen ble feilmeldinger som var vanskelige å tolke limt inn i ChatGPT, gjerne med spørsmål som «hva kan være grunner til at dette ikke virker?» eller «forklar dette». Da dette var koderelaterte spørsmål, var det lett å sjekke om svarene som ble gitt var gode eller ikke.

ChatGPT (OpenAI, 2024) er også bruk i utviklingen av funksjonen «find_points_in_tiff», en funksjon som finner senterpunktene utvalgte tiff-filer. Spørsmål som «set a point in the centre several layers of tiff files» og «calculate centroids from tiff» ble benyttet. Svarene ble inkorporert i egen kode, og utviklet til å kunne iterere gjennom hele mapper med bilder. Resultatene ble manuelt dobbeltsjekket og gav riktige punkter.

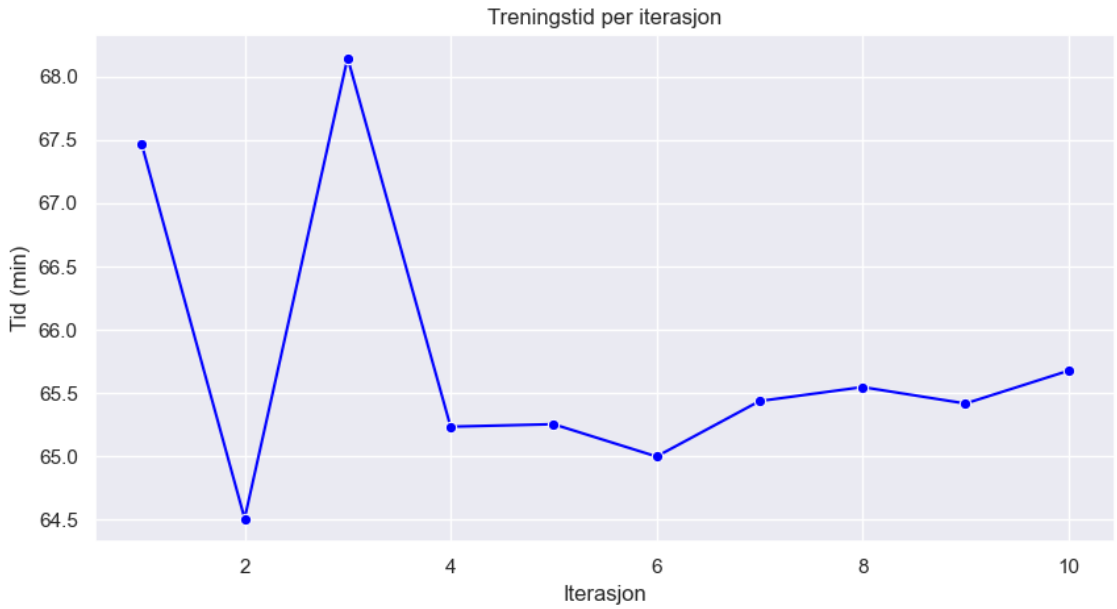
4 Resultater

I dette kapitelet presenteres og sammenlignes resultater og prediksjoner fra de to modellene. Kapittelet er delt inn i tre deler, to for hver av modellene og en for sammenlikning av resultatene. Modellenes deler er begge delt i to, først en del som presenterer resultater av mange gjennomkjøringer, og deretter en del som beskriver resultatene fra det som ble funnet å være den beste modellen.

4.1 CNN

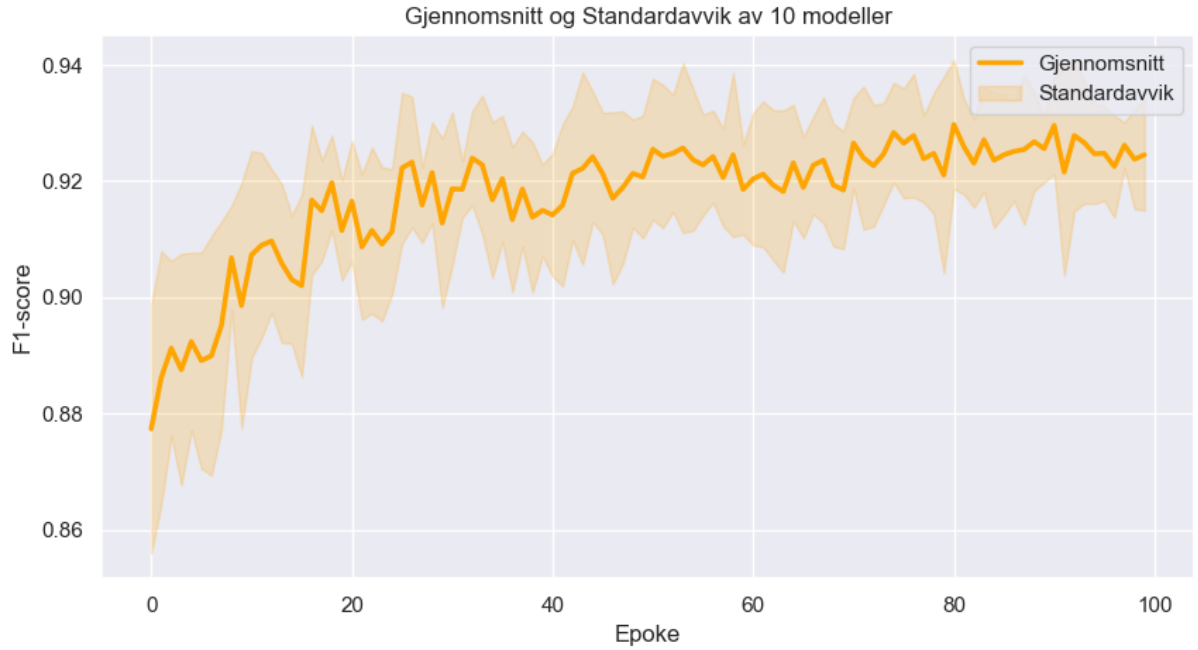
4.1.1 Treningsresultater

For å unngå tilfeldige svake klassifiseringer grunnet den tilfeldige fordelingen av startvekter, ble hoveddelen av nettverket trent totalt 10 modeller av samme oppsett. Hver treningsperiode ble gjort over 100 epoker. Underveis i treningen ble F1-scoren på både trenings- og valideringssettet overvåket. Det var den epoken med høyest validering i form av F1-score, fra den beste modellen, som ble valgt med videre for å finjusteres. Også her var den beste F1-score på valideringssettet som ble tatt med videre, og selve klassifiseringsestimatene ble beregnet på denne modellen. Treningen tok omtrent 39,5 sekunder per epoke, og 66 minutter for treningen av hele nettverket per iterasjon. Totalt brukte nettverket 10 timer og 58 minutter på treningen. Treningstidene for alle de 10 iterasjonene kan ses under i figur 4.1.

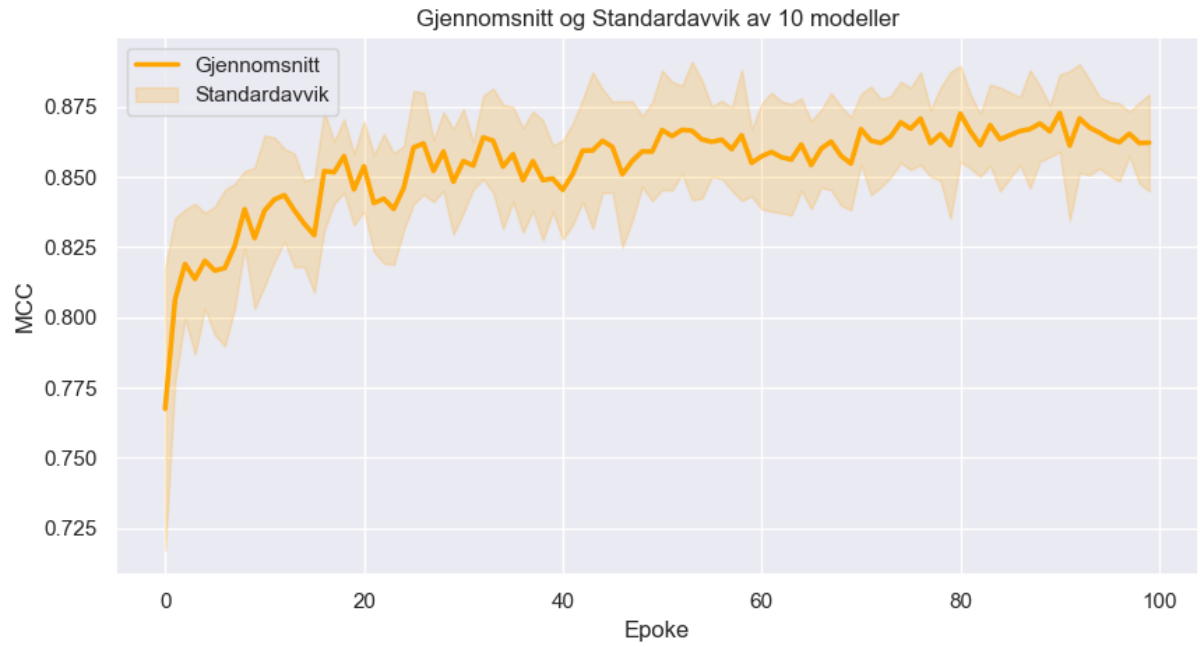


Figur 4.1: *Treningstider per iterasjon CNN.*

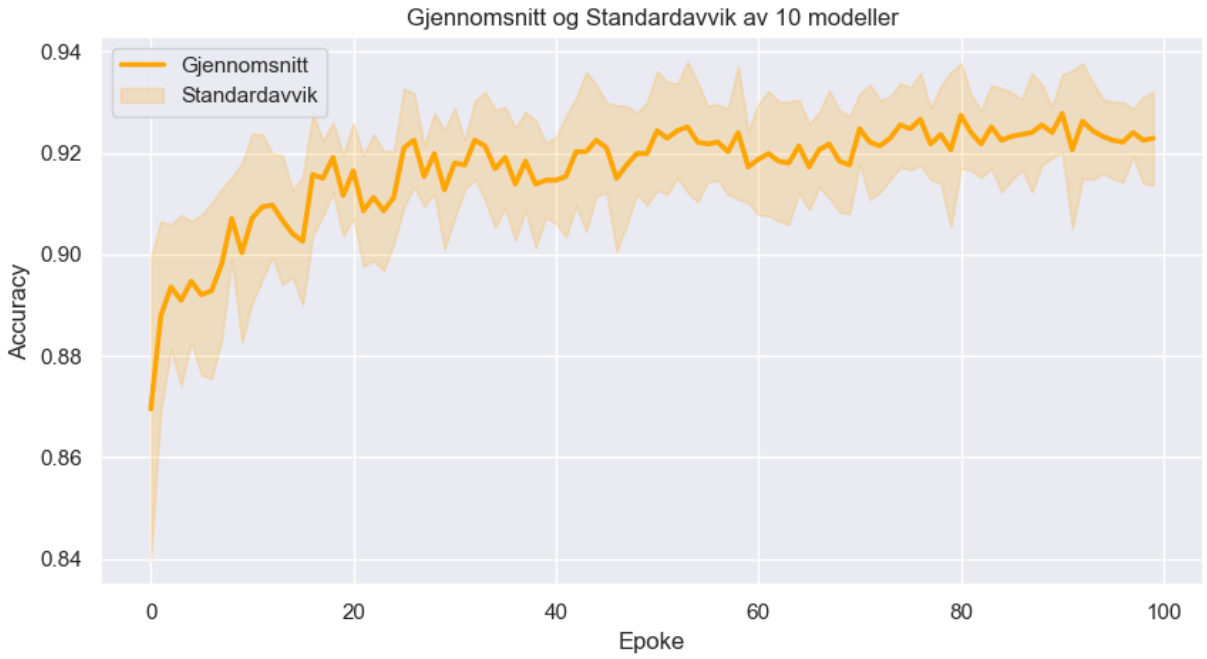
Over de 10 iterasjonene med trening av nettverket, ble vurderingsmetodene F1-score, MCC og nøyaktighet for både trening- og valideringssettet lagret for å evaluere på nettverkets ytelse. Alle de tre metodene er altså hensiktsmessige å maksimere. 10 iterasjoner med 100 epoker gav 1000 datapunkter å analysere per vurderingsmetode. Figurene 4.2, 4.3 og 4.4 viser en beregnet gjennomsnittsverdi, med standardavvik, for de tre vurderingsmetodene, på de 266 bildene som utgjorde valideringssettet. En kan se at i alle iterasjonene startet verdiene for vurderingsmetodene som relativt høye. Dette skyldes integrasjonen av overføringslæringsmodellen ResNet50V2 som gav modellen høy ytelse fra og med første epoke.



Figur 4.2: *Utviklingen av F1-score over 100 epoker gjennom ti ulike gjennomkjøringer.*



Figur 4.3: *Utviklingen av MCC over 100 epoker gjennom ti ulike gjennomkjøringer.*



Figur 4.4: *Utviklingen av nøyaktigheten over 100 epoker gjennom ti ulike gjennomkjøringer.*

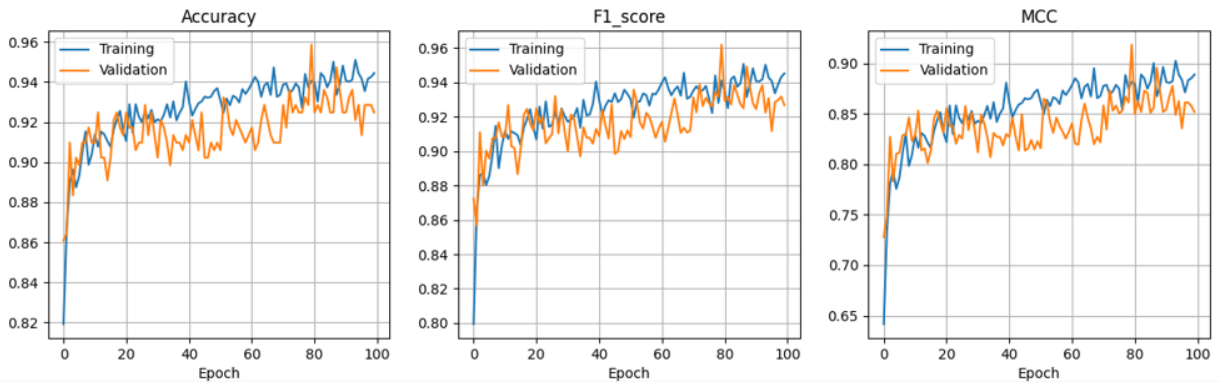
Tabell 4.1 viser de gjennomsnittlig høyeste verdien for de ulike vurderingsmetodene fra de 10 iterasjonene med tilhørende standardavvik. F1-score og nøyaktighet følger hverandre ganske tett, mens MCC-verdien er noe lavere.

Tabell 4.1: *Vurderingsmetoder. Gjennomsnittlig høyeste verdi og standardavvik gjennom 10 gjennomkjøringer.*

Metode	gjennomsnitt høyeste verdi ± standardavvik
F1-score	0,948 ± 0,006
MCC	0,903 ± 0,009
Nøyaktighet	0,945 ± 0,005

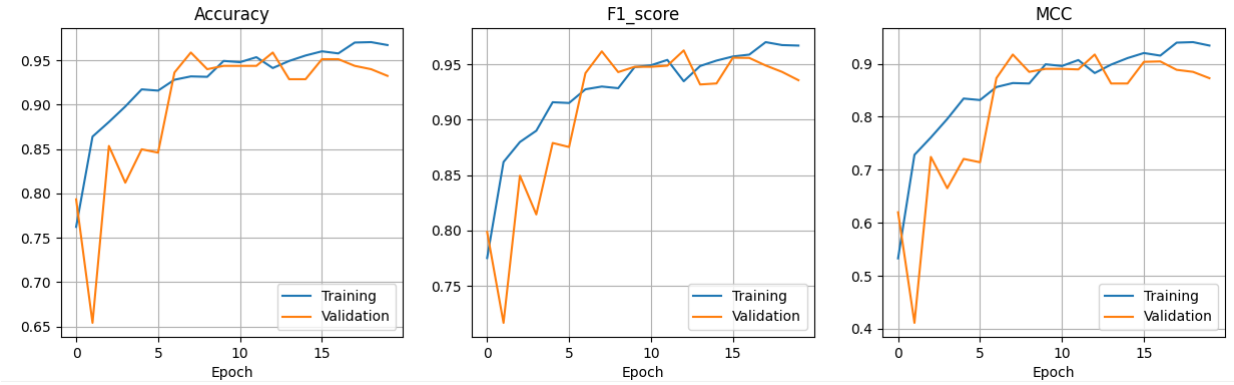
4.1.2 Beste modell

Av de 10 iterasjonene var det modell nummer 8 som gav høyest F1-score, og dermed ble valgt som beste modell. Figur 4.5 viser hvordan nøyaktigheten (Accuracy), F1-scoren og MCC-verdien utviklet seg i løpet av treningen av nettverket, for både trenings- og valideringssettet. Høyeste verdier for vurderingsmetodene på valideringssettet kom alle sammen i epoke 79 hvor F1-scoren var på 0,962, MCC på 0,927 og nøyaktigheten var på 0,959. Epoken kan ses i figur 4.5, hvor valideringkurven plutselig gjør et markant hopp i alle vurderingsmetodene. Grunnet disse gode resultatene i alle vurderingsmetodene, fallt valget av modell for klassifisering på epoke nummer 79 fra iterasjon 8.



Figur 4.5: *Treningshistorikken til gjennomkjøringen med høyest F1-score. Plottet viser utvalgte verdier for både test- og valideringssettet.*

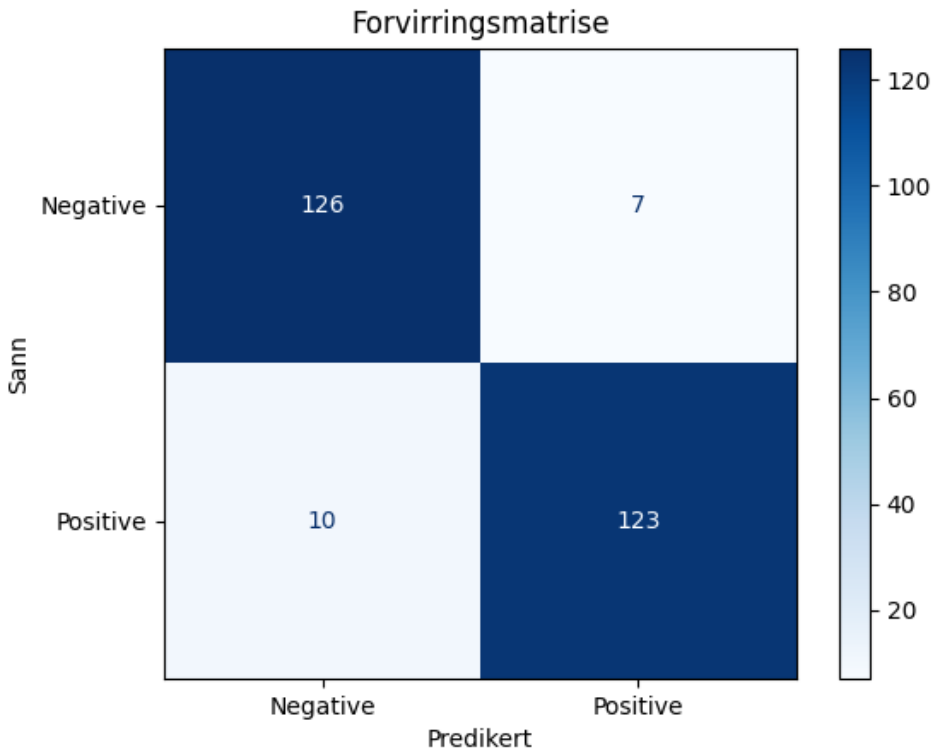
Finjustering av nettverket ble gjort for å kompensere for det relativt lille datasettet som er brukt. Justeringen ble utført over 20 epoker, og gav økte verdier for både nøyaktighet, F1-score og MCC. Treningshistorien kan ses i figur 4.6.



Figur 4.6: *Treningshistorikken til finjusteringen av det høyest scorende nettverket. Plottet viser utvalgte verdier for både test- og valideringssettet.*

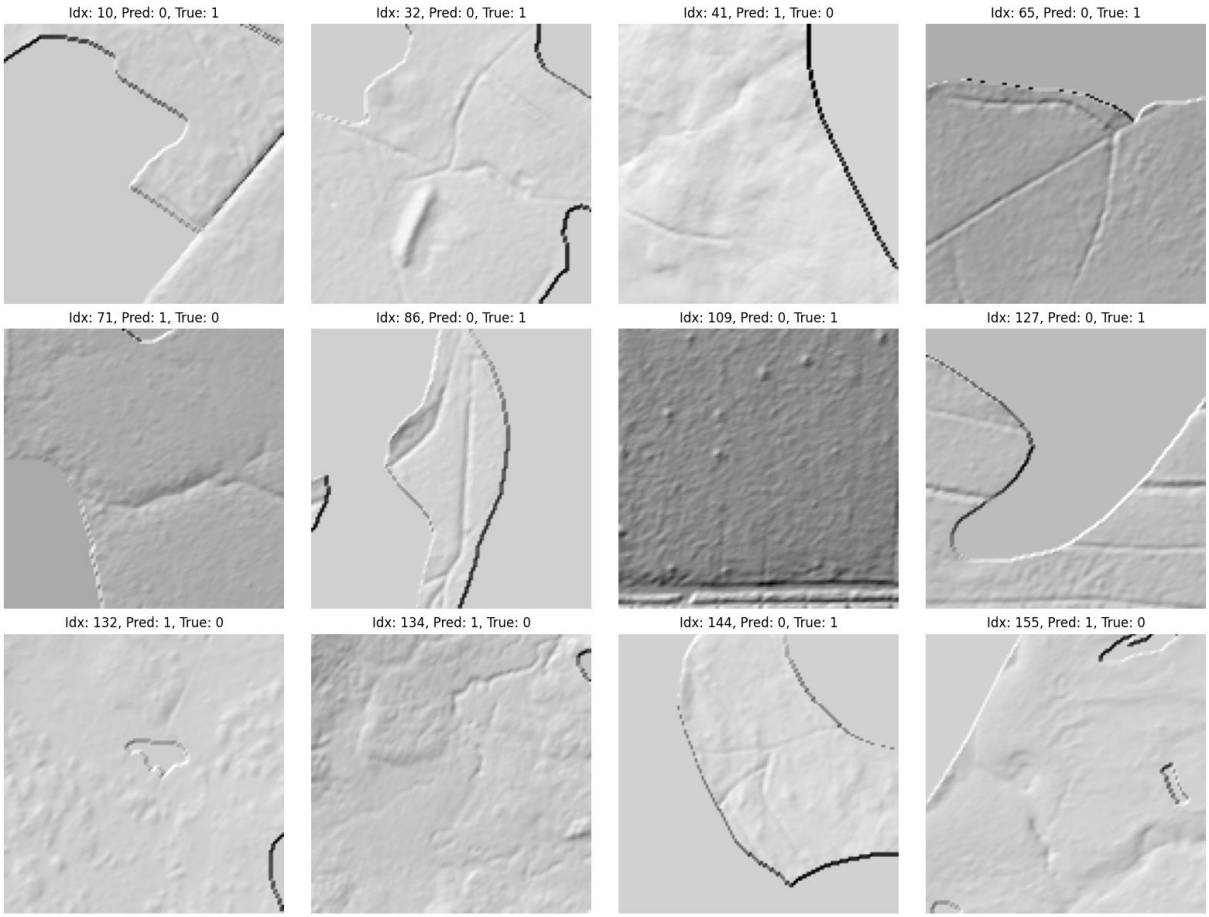
Resultater på testsett

Bildene fra testsettet ble sendt gjennom den valgte modellen, og det ble produsert estimater for de to klassene. Testsettet besto, i likhet med valideringssettet, av 10 % av datasettet, altså 266 bilder hvor begge klassene var representert. I figur 4.7 kan klassifiseringens forvirringsmatrise ses. Figuren viser hva slags type feil som er gjort i modellens estimater. Feil av typen falsk negativ (FN) forekommer hyppigere enn falsk positiv (FP). Det kan skyldes både en systematisk skjelvfordeling i type feil som gjøres, eller en tilfeldig fordeling grunnet et relativt lite testsett.



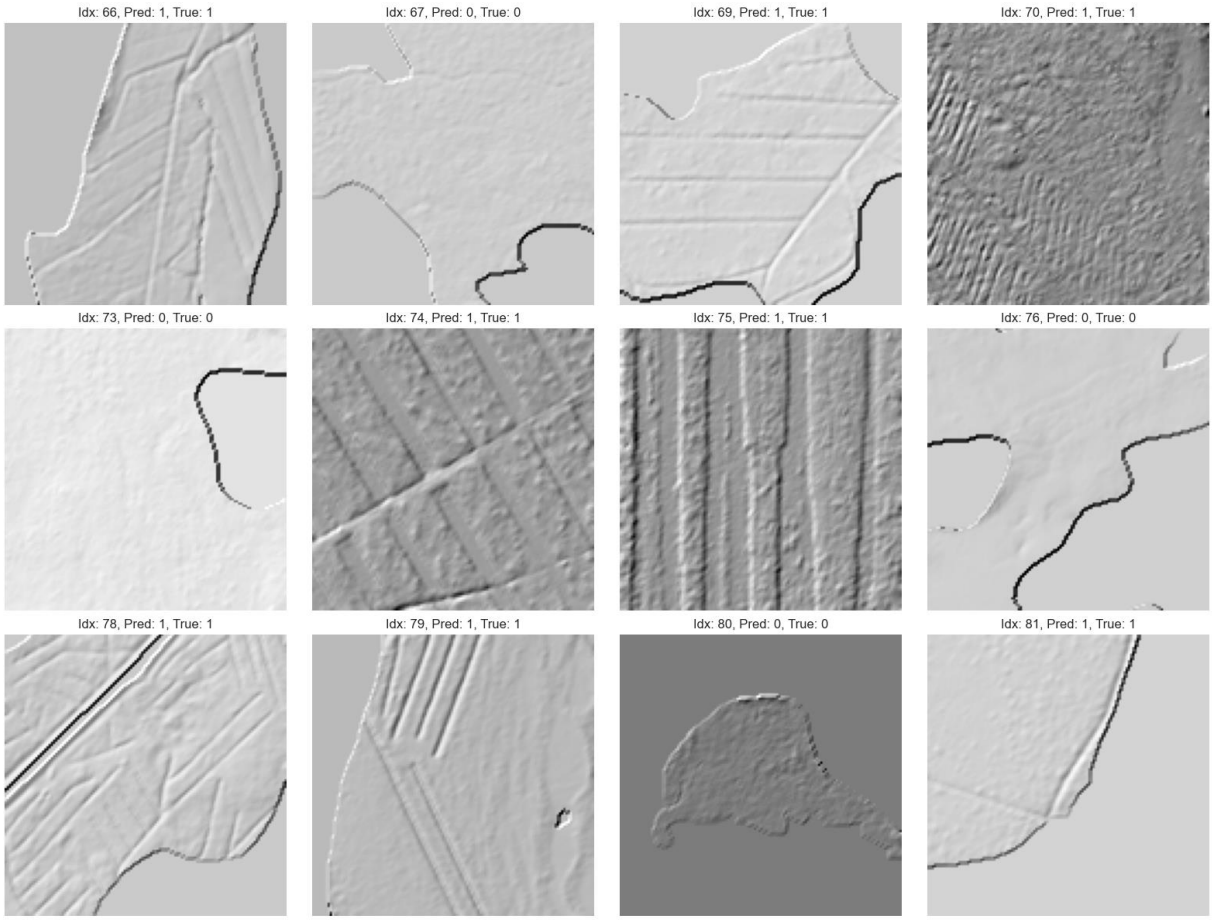
Figur 4.7: *Forvirringsmatrise som viser fordeling av feilklassifiseringene gjort av CNN-modellen.*

Av bildene i testdatasettet var det totalt 17 stk. som ble feilklassifisert. 12 av disse er plottet i figur 4.8. Her kan en se hvordan begge typer feil er gjort i klassifiseringen. En gjennomgang av de feilklassifiserte bildene viste at det kun var 2 av dem som var grøftet for torvuttak, mens 8 av bildene var dreneringsgrøfter. Bildene av feilklassifiseringene viser også at forvirringsmatrisen fra figur 4.7 stemmer: uoppdagede grøfter forekommer oftere enn feilaktig oppdagede grøfter.



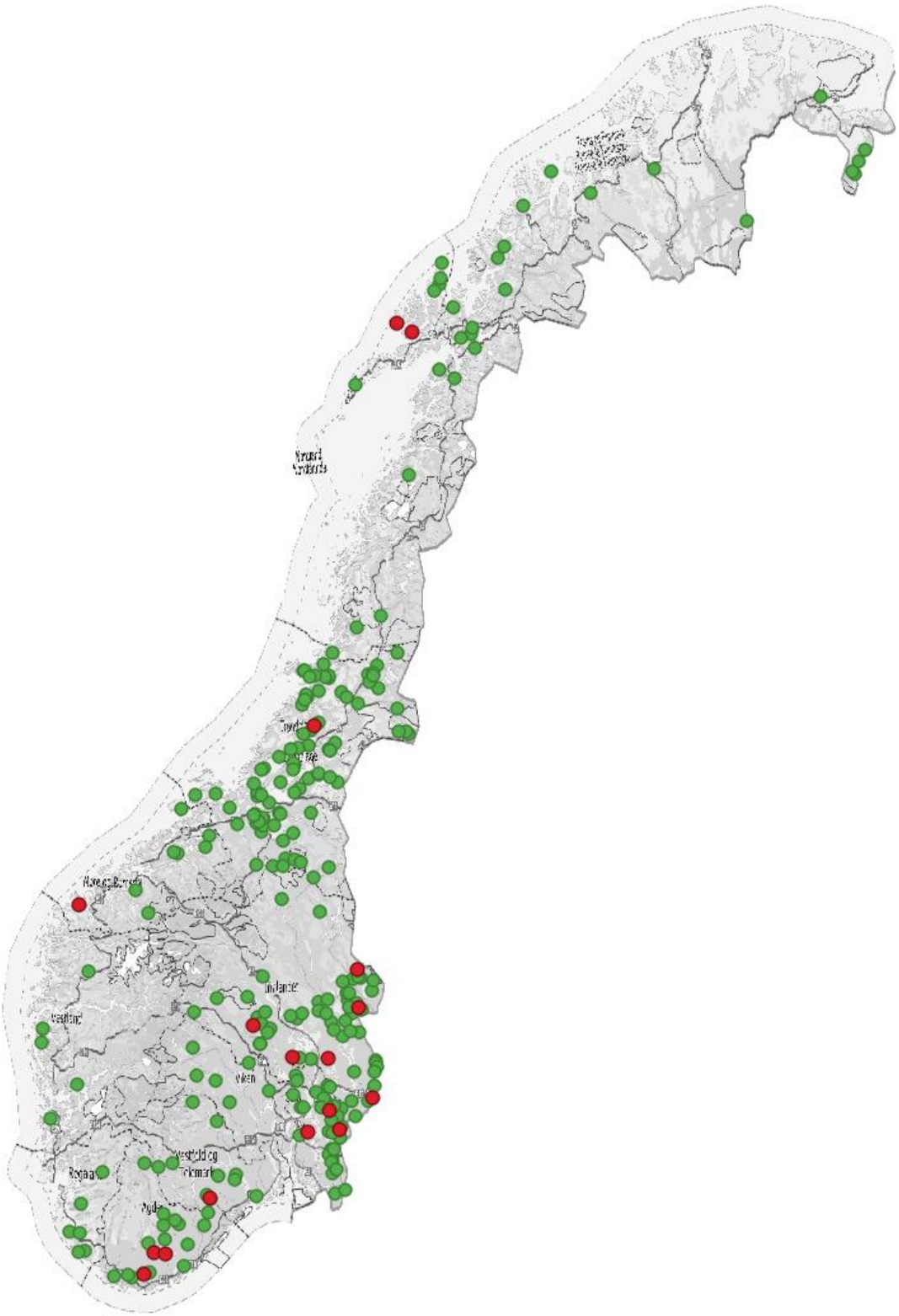
Figur 4.8: *Noen av de bildene CNN-modellen feilklassifiserte, med tilhørende indeks i testsettet, samt predikerte og sanne verdi.*

Et utvalg riktige klassifiseringer forekommer i figur 4.9. Her kan en på de to bildene i midten se hvordan torvuttak etterlater større bånd av grøfter i terrengmodellen, som lettere oppdages av modellen. I tillegg oppdager modellen også dreneringsgrøfter som forgreiner seg ut i store grøftenettverk, slik en kan se iblant annet utklippene med idx. 66, 69, 78 og 79 i figuren.



Figur 4.9: *12 utvalgte korrekt klassifiserte bilder. Utvalget består av både grøftede og ikke grøftede myrer, og de grøftede myrene er både torvuttak og grøfter for annen drenering.*

Med de klassifiseringene som er gjort av testdatasettet, endte modellen med scorene nøyaktighet = 0,94, F1-score = 0.94 og MCC = 0,87. Presisjon og sensitivitet ender på henholdsvis 0,95 og 0,92. I figur 4.10 kan en se hvordan de korrekt og ukorrekt klassifiserte myrene er fordelt i landet. Det er totalt 249 korrekte og 17 ukorrekte klassifiseringer i figuren.

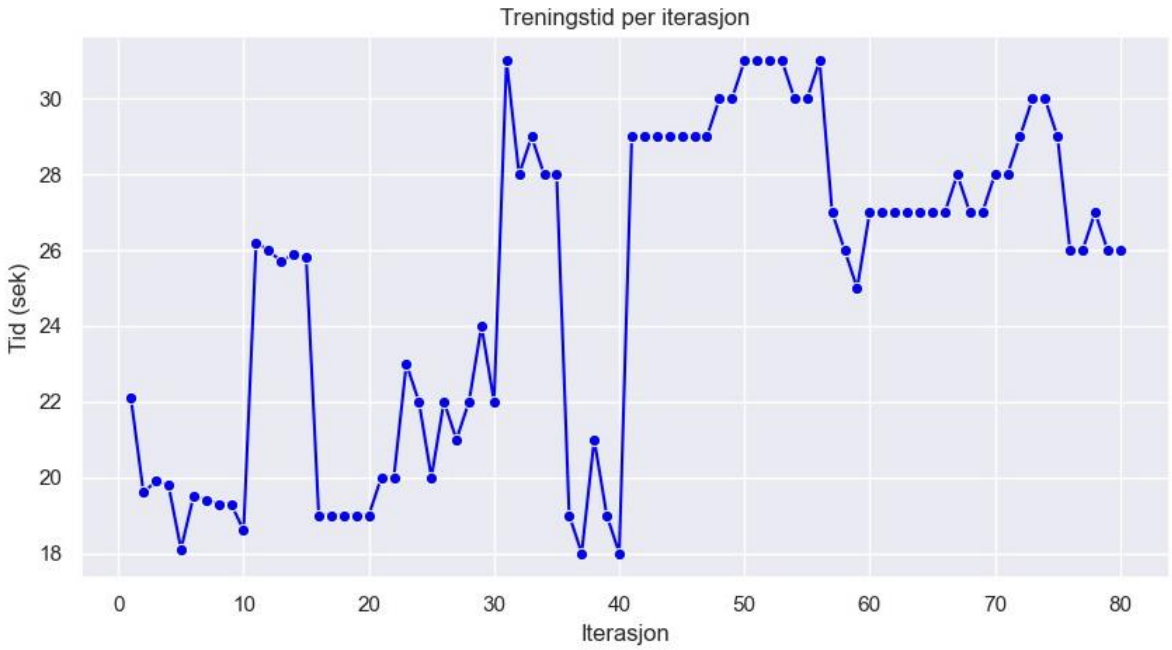


Figur 4.10: *Visualisering av bildene i testsettets beliggenhet. Klassifiseringene er gjort av CNN-modellen. Grønne er korrekt klassifiserte bilder, røde er ukorrekte.*

4.2 SVM

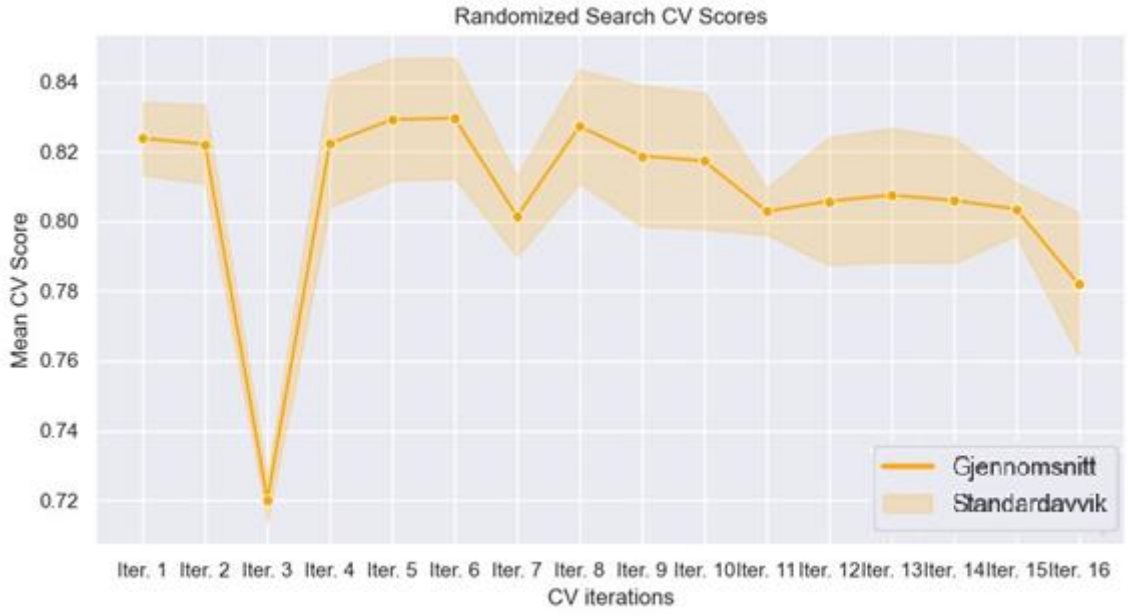
4.2.1 Treningsresultater

For å finne modellen med de beste hyperparameterne ble det gjennomført en 5-fold-kryssvalidering med 16 forskjellige sammensetninger av hyperparametere, hvilket betyr at nettverket ble trent totalt 80 ganger. Treningstider for de ulike treningsepokene kan ses i figur 4.11. Totalt tok treningsprosessen 33 minutter. Modellen er trent med RandomSearchCV som bruker tilfeldig sampling av hyperparameter-rommet for å finne den beste modellen.



Figur 4.11: Treningstid for SVM-modellen. Treningen består av 16 varianter * 5-fold-kryssvalidering = 80 iterasjoner

For å kalkulere ytelsen til modellene er F1-score benyttet. Resultatene for de ulike iterasjonene kan ses i figur 4.12. Den markerte streken gjennom plottet er gjennomsnittlig F1-score for hver 5-fold-kryssvalidering, mens det skraverte område er tilhørende standardavvik.



Figur 4.12: *Kryssvalideringsscore for de 16 ulike variantene som er testet. Grafen markerer gjennomsnittsverdi gjennom kryssvalideringen.*

4.2.2 Beste modell resultater

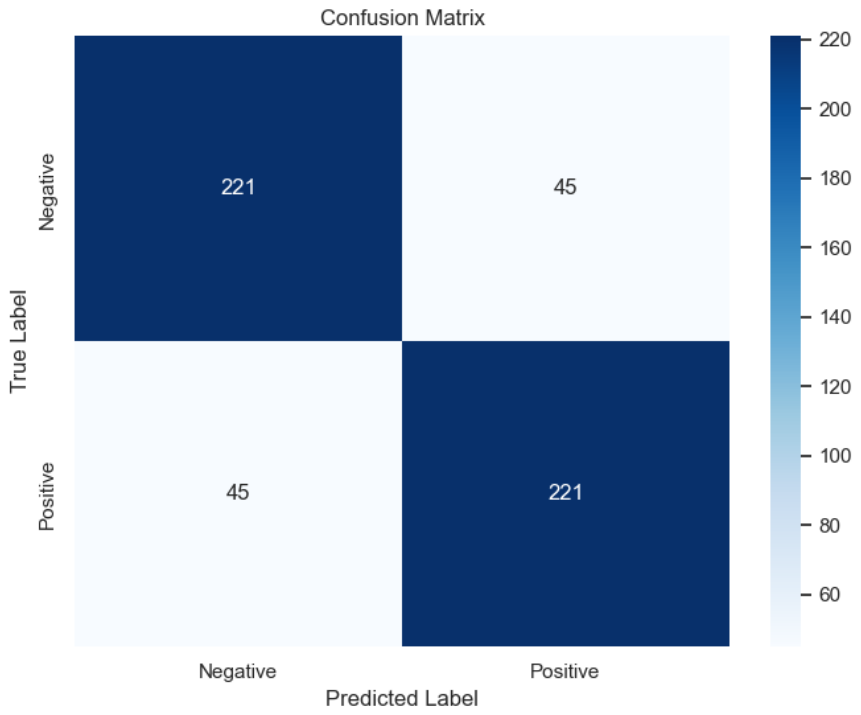
Kryssvalideringen som ble foretatt viste at modellen som gav de beste resultatene hadde hyperparameterne fra tabell 4.2, og gav en kryssvalideringsscore, regnet av F1-score, på 0,83. I en SVM vil verdien γ bety $\frac{1}{antall\ egenskaper}$, som tidligere var standardinnstillingen for SVM i Scikit-Learn.

Tabell 4.2: *Oversikt over valgte parametere etter hyperparameteroptimalisering.*

C-parameter	1
γ	Auto
Kjerne	RBF
Klassevekt	Balansert

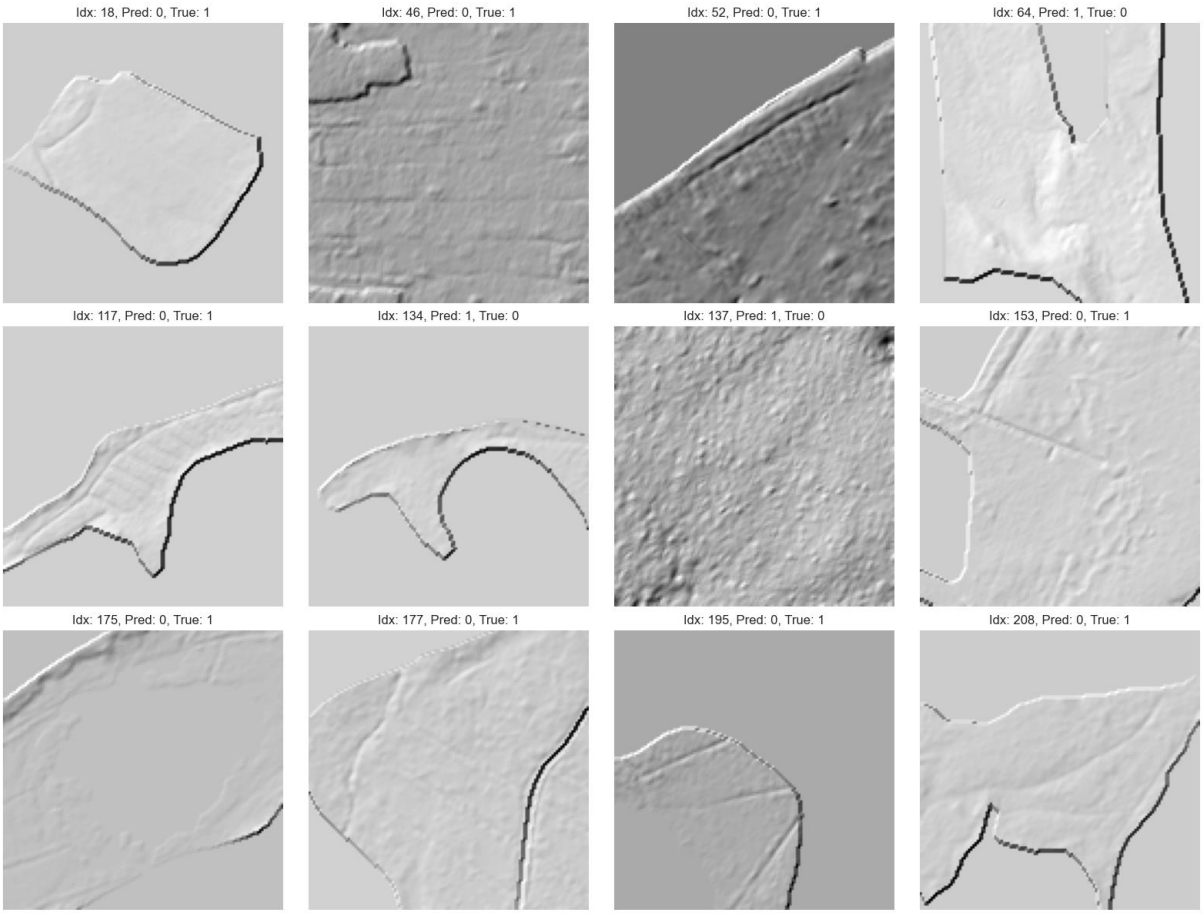
Resultater på testsettet

Testsettet bestående av 532 bilder ble sendt gjennom modellen for binær klassifisering Resulterende forvirringsmatrise kan ses i figur 4.13. Matrisen er symmetrisk fordelt, hvor sann positiv og sann negativ er like, og falsk positiv og falsk negativ har like mange tilfeller.



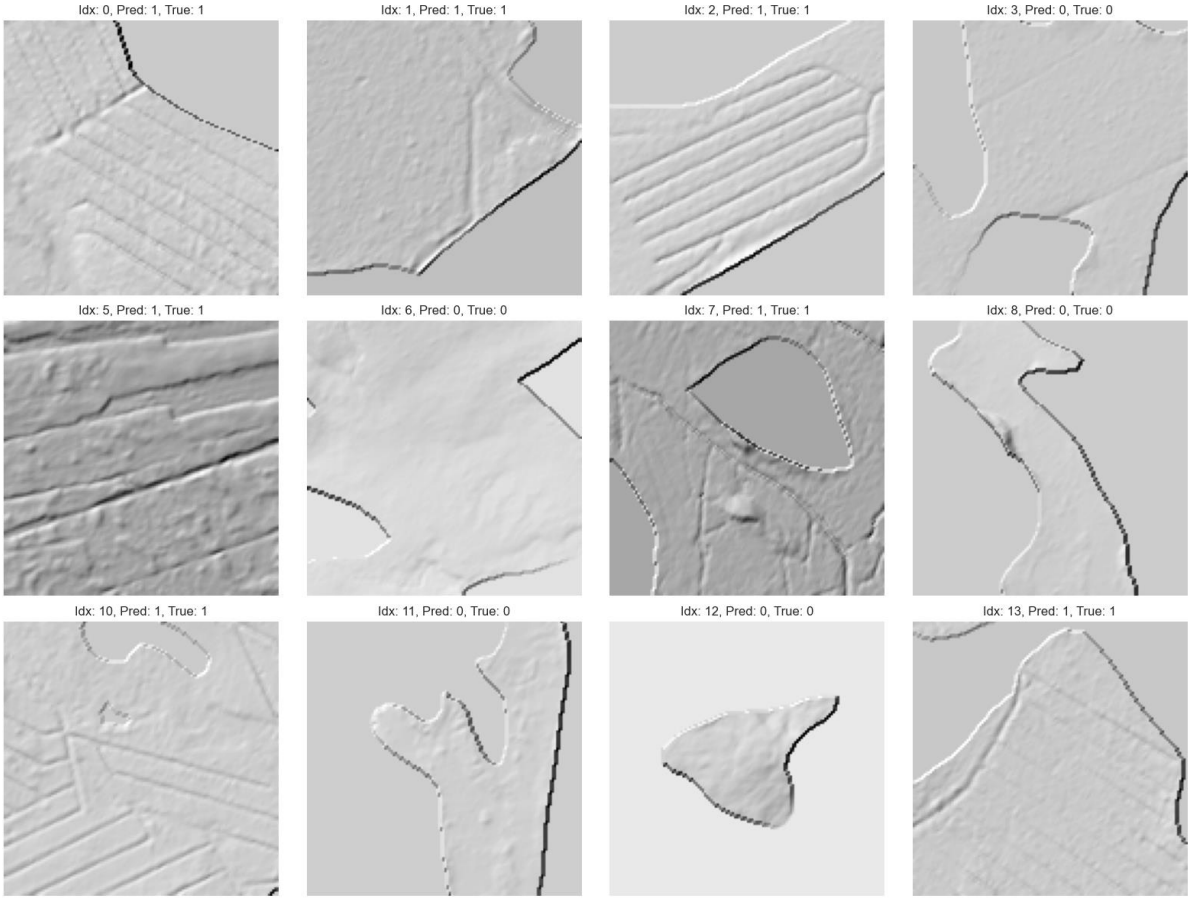
Figur 4.13: *Forvirringsmatrise som viser fordelingen av feilklassifiseringene gjort av SVM-modellen.*

Figur 4.14 viser et tilfeldig utvalg av 12 av bildene modellen ikke klassifiserte riktig. Slik som forvirringsmatrisen tilsier, er det jevn fordeling mellom de typene feil modellen har gjort.



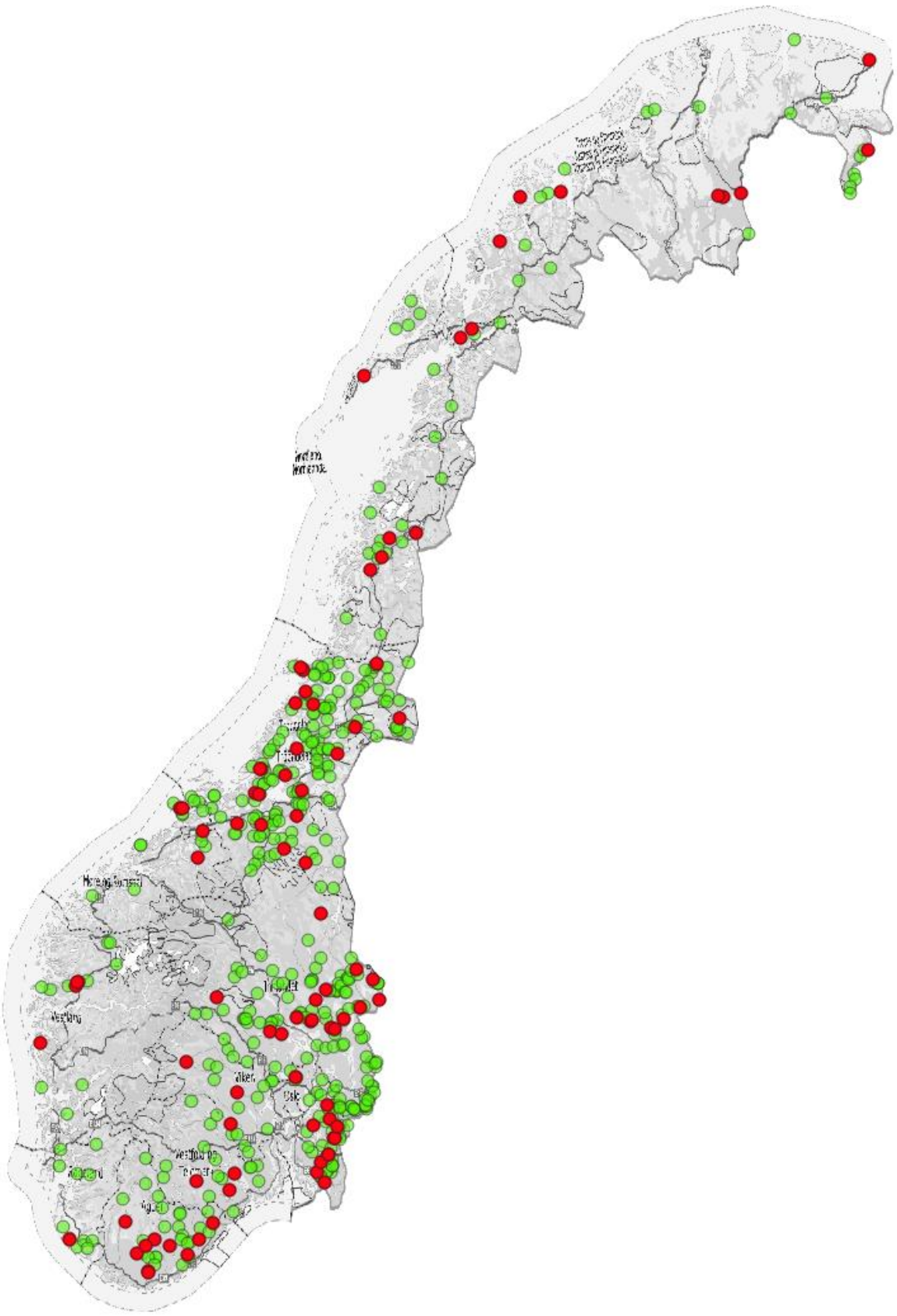
Figur 4.14: *Noen av de bildene SVM-modellen feilklassifiserte, med tilhørende indeks i testsettet, samt predikerte og sanne verdi*

De korrekte klassifiseringene er også jevnt fordelt mellom TP og TN. I figur 4.15 viser bildet med idx: 5 et korrekt klassifisert torvuttak, mens inx: 10 viser et system av dreneringsgrøfter. SVM-modellen er, i likhet med CNN-modellen, også god på å oppdage de tykke båndene med grøfter en finner i torvuttak.



Figur 4.15: *12 utvalgte korrekt klassifiserte bilder. Utvalget består av både grøftede og ikke grøftede myrer, og de grøftede myrene er både torvuttak og grøfter for annen drenering*

Etter alle gjennomkjøringene og valg av beste modell, endte klassifisering med en F1-score på 0, 83, mens nøyaktighet var på 0,83 og MCC på 0,66. Presisjon og sensitivitet endte også begge på 0,83. Det er gjort 532 klassifiseringer vha. SVM-modellen, hvor 90 ble klassifisert feil og 442 riktig. Deres beliggenhet finnes i figur 4.16.



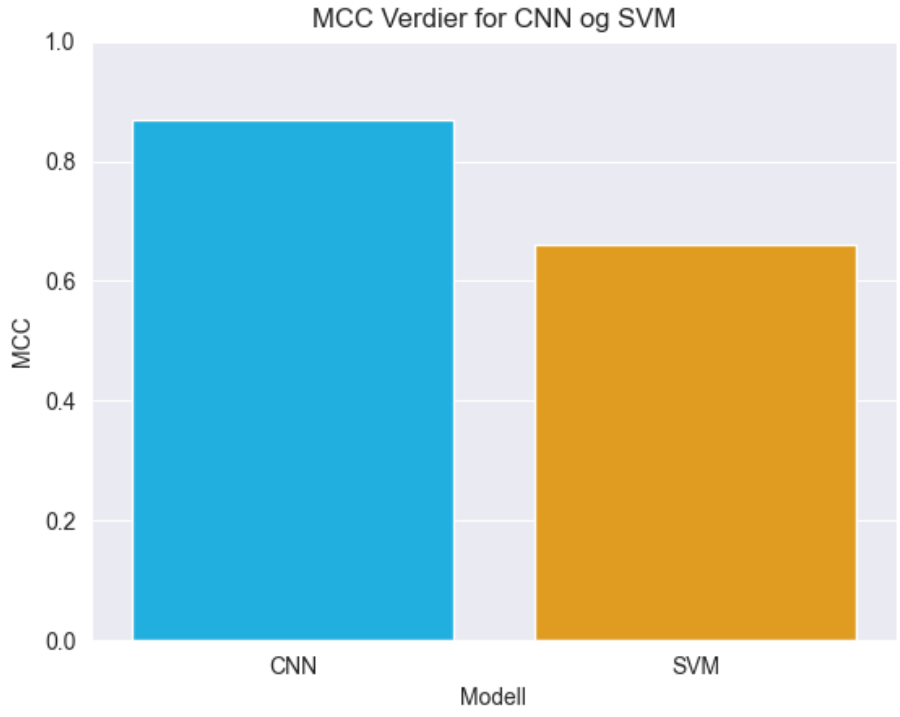
Figur 4.16: *Visualisering av bildene i testsettets beliggenhet. Klassifiseringen er gjort av SVM-modellen. Grønne er korrekt klassifiserte bilder, røde er ukorrekte.*

4.3 Sammenlikning av CNN og SVM

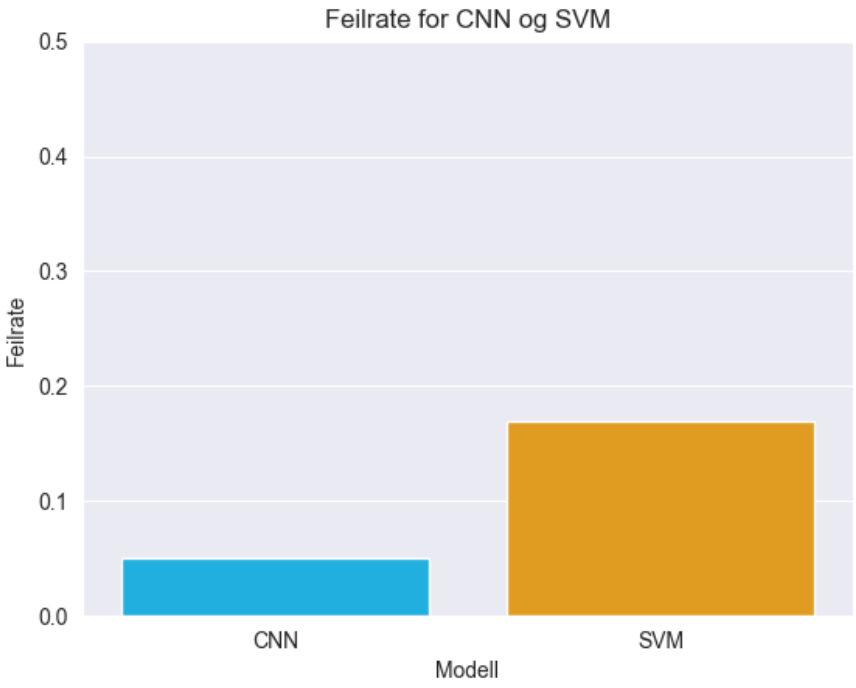
For å enklere kunne sammenlikne de to modellenes resultater, noteres de her ved siden av hverandre i tabell 4.3. Tabellen viser de beregnede gjennomsnittverdiene for F1, MCC og nøyaktighet, med tilhørende standardavvik, fra treningsprosessene av begge modellene.

Tabell 4.3: *Oppsummering av de to modellenes resultater, målt i de utvalgte enhetene.*

	CNN	SVM
F1	0,94	0,83
Nøyaktighet	0,94	0,83
MCC	0,87	0,66
Presisjon (pre)	0,95	0,83
Sensitivitet (rec)	0,92	0,83

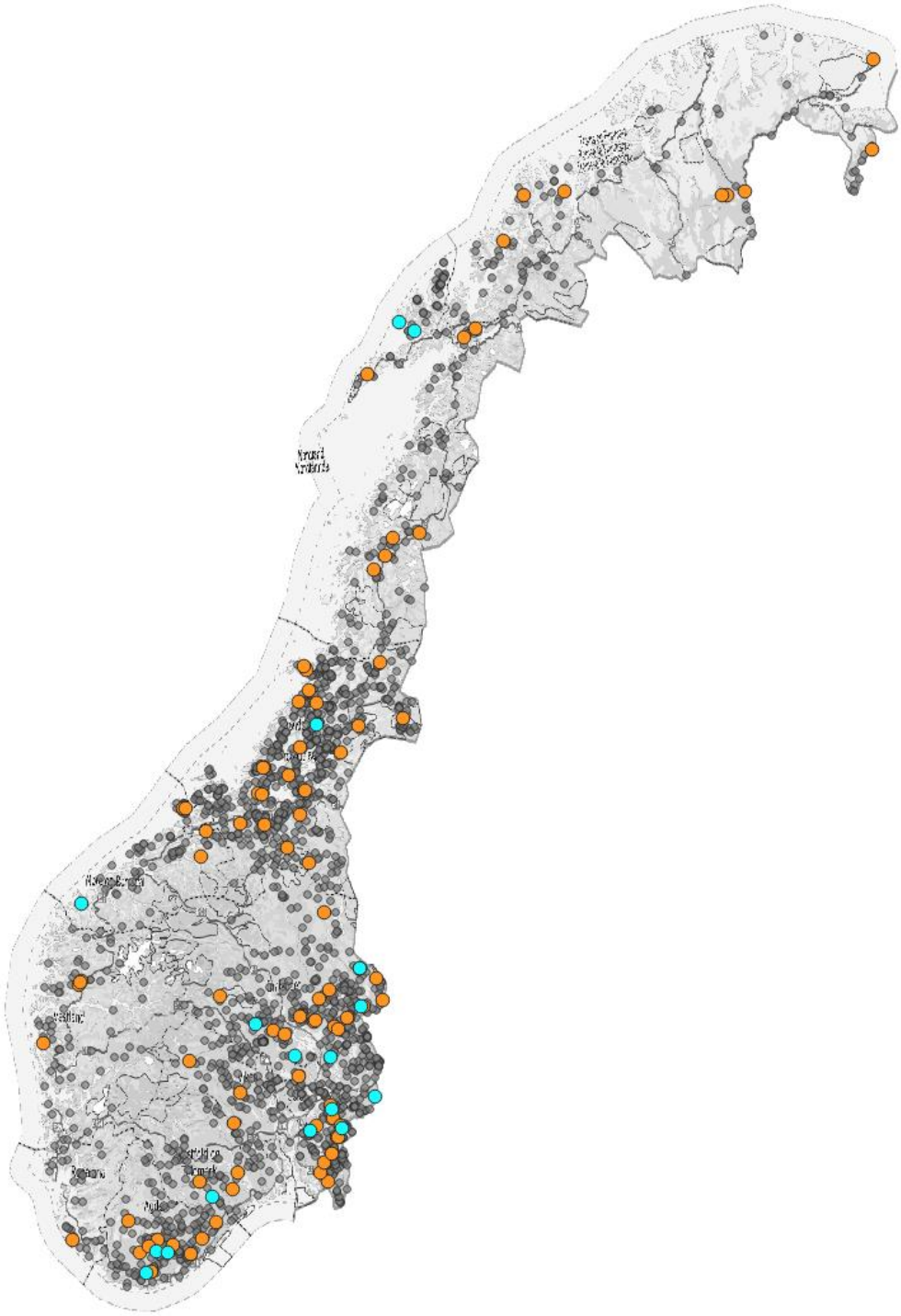


Figur 4.17: *De to modellenes MCC-verdi plottet ved siden av hverandre.*



Figur 4.18: *De to modellenes feilrater plottet ved siden av hverandre. CNN-modellen har en feilrate på 0,06 og 0,17.*

Figurene 4.17 og 4.18 viser MCC og feilrate for de to modellene. En kan se hvordan disse henger sammen, hvor høyere MCC-verdi gir lavere feilrate. Testsettet til SVM-modellen var på 532 bilder, mens CNN-settet var kun 266. Forholdet mellom fordelingene er altså uproporsjonalt. Allikevel gir figuren et bilde på hvor myrene som ble feilklassifisert befinner seg. I bakgrunnen vises det fulle datasettet som 2656 grå punkter.



Figur 4.19: *Feilklassifiseringer for de to modellene. CNN er blå og SVM er oransje. De korrekt klassifiserte kan ses som de grå punktene i bakgrunnen.*

5 Diskusjon

I dette kapittelet diskuteres datasettet som er brukt i oppgaven, treningen av modellene og resultatene som ble presentert i kapittel 4. Deretter sammenliknes resultatene med relaterte arbeider, før kapittelet avsluttes med noen forslag til videre arbeid.

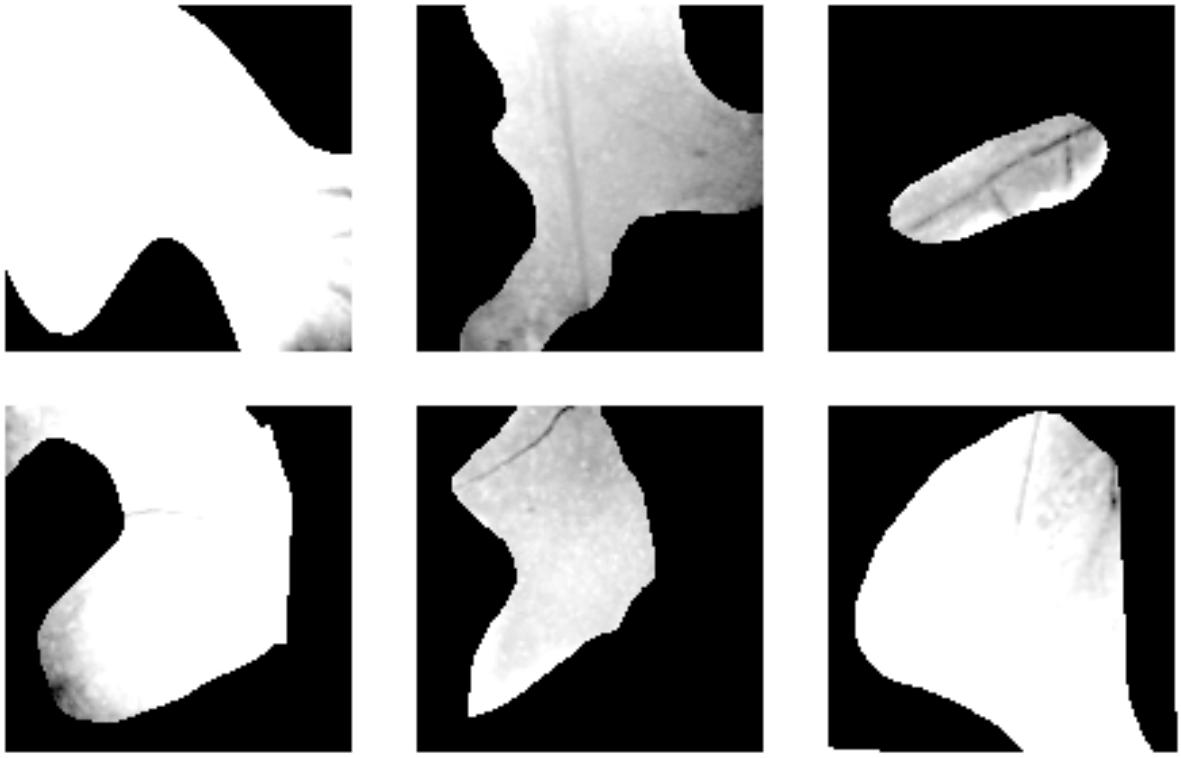
5.1 Datasettet

Dataene som ble benyttet er 128 x 128 piksel store bilder av norske myrområder. Bildene er hentet og levert av NIBIO.

Datasettet som er benyttet kommer fra laserskanninger gjort med LiDAR. Utklippene kommer fra DTM1-prosjektet som generelt har høy kvalitet, grunnet kartverkets strenge krav til datainnsamling og kvalitetskontroll. Dataen kom som gråskalabilder, hvor hver piksel har fått verdi ut ifra høyden i punktet. I tillegg er bildene lagret som GeoTIFF, hvilket betyr at bildenes koordinater også er lagret i filene. Det gjorde det mulig å plotte bildenes beliggenhet i terrenget. I datasettet med grøftede myrer fantes bilder av både torvuttak og andre grøfter for drenering av myr. Bildene var plukket ut fra hele Norge og gav dermed god variasjon i bildene, med varierte terrengmodeller fra nord til sør.

Det er verdt å merke seg at det til tross for generelt gode data, har vært noen utfordringer knyttet til datasettet. I datasettet med bilder av ikke grøftede myrer forekommer det en del stier, traktorveier og små bekker som går gjennom myra. Dette kan likne veldig på enkelte tilfeller av grøftede myrer, hvor det i bildet ikke er noe grøftesystem, men kun en eller to små grøfter. Visualisering av dette finnes i figur 5.1. Det burde også nevnes at det nok forekommer feil når det gjelder markering av bildene i datasettet, en vanlig utfordring som oppstår i dyplæring, og skyldes i dette tilfelle menneskelig feil da bildene er markert manuelt. En modell kan aldri bli bedre enn dataene en putter inn, så dersom for eksempel 5 % av testsettet er markert feil kan modellen aldri bli mer enn 95 % nøyaktig. På samme måte kan en risikere enda større feilmargin dersom en har 5 % feilmarkeringer i treningsdata, da modellen kan overtilpasses feil sannhet eller utvikle manglende evne til å korrekt generalisere de gale dataene.

Begge klassene var representert likt i trenings- og testsettet i begge modellene, så vel som i valideringssettet i CNN-modellen. Fordelingen av torvuttak og andre grøfter er ukjent, da disse ikke er spesifikt markert i datasettet, men ligger under grøftet myr.



Figur 5.1: *Visualisering av mulige feilkilder. Figuren viser ikke grøftede myrer øverst, og grøftede myrer under.*

5.2 Trening av modellene

Valg av overføringslæring som modellgrunnlag og ResNet som forhåndslærte vekter, var basert på tidligere litteratur [(Teien, 2022), (Yang et al., 2023), (Li et al., 2021)]. Etter testing med ulike ResNet-modeller havnet valget til slutt på ResNet50V2. Selve treningen av nettverket tok litt over en time på den tilgjengelige datamaskinen.
SVM ble valg som klassisk maskinlæringsmodell etter å ha lest litteratur med liknende problemstilling [(Hagen, 2023), (Blum et al., 2018)]. Denne var vesentlig raskere å trene enn CNN-modellen og brukte rundt 30 sekunder per iterasjon.

CNN-modellen krevde at ResNet50V2 ble lastet inn, og med en dybde på 50 lag utgjorde den en størrelse på 98 MB. Totalt endte den trente CNN-modellen på 203 MB, mens SVM-modellen tok opp 45 MB. En kan dermed si at CNN-modellen er mer komplisert og krever mer computerkraft enn SVM-modellen.

5.3 Resultater og ytelse

Tabell 4.3 viser, med utvalgte vurderingsmetoder, hvordan de to modellene presterte på testsettet. Resultatene viser at CNN-modellen gir høyere verdier enn SVM-modellen i alle enhetene. Det ble også, etter mye testing, avdekket at CNN-modellen gav bedre resultater når bildene var filtrert med et Laplace-filter. Det antas at bruken av Laplace-filteret hjalp til med å fremheve kantegenskapene i bildene, noe som gjorde at modellene lettere kunne gjenkjenne visuelle trekk, som kan ha vært viktige i klassifiseringen. I tillegg kan filtreringen også ha bidratt med å redusere støyen i bildene, som igjen hjalp modellen med å fokusere på de viktige trekkene i bildene. CNN-modellen ble også testet uten Laplace-filter og gav omtrent like gode valideringstall som med filteret. Forskjellen var at den da overtilpasset seg ganske kraftig til treningsdataen, slik en så i figur 3.4.

SVM-modellen presterte best med et Canny-filter lagt over. Dette gjorde det lettere for støttevektormaskinen å fokusere på relevante egenskaper i bildet, ved å redusere mengden irrelevant data. Canny-filtret hentet ut kantene som hovedtrekket i bildet, hvilket lot SVM-modellen finne beslutningsgrensene mellom klassene på en mer effektiv måte.

Den vurderingsmetoden de to modellene ligger nærmest hverandre, er i sensitivitet, hvor CNN ligger på 0,92 og SVM på 0,83. Dette betyr at den klassifiseringsegenskapen SVM-modellen er nærmest CNN-modellen, er i deteksjon av sanne positive tilfeller. Dette er også en av de lavere scorene til dyplæringsmodellen, noe som kommer frem av forvirringsmatrisen i figur 4.7. Selv om det ikke like kritisk med falske negative i denne oppgaven som i andre klassifiseringstilfeller, som for eksempel i medisinske diagnoser eller sikkhetssystemer, er det likevel ikke optimalt. Målet med kartlegging av grøfting i norske myrer kommer av et ønske om å beregne klimagassutslipp knyttet til nettopp dette, vil falske positive utgjøre en mindre trussel enn falske negative, da disse potensielt kan hindre at grøftede myrer tas med i klimaregnskaper.

Det er et stort sprang mellom modellene når det gjelder MCC-verdi, hvor CNN endte på 0,87 og SVM på 0,66. MCC tar i bruk alle verdier i forvirringsmatrisen, som betyr at en modell som produserer høy MCC-score vil være en svært robust modell med høy treffprosent i klassifiseringene den foretar. MCC på 0,87 indikerer veldig god prediktiv ytelse, med sterk korrelasjon mellom observerte og predikerte klassifiseringer. 0,66 er også rimelig sterkt, men med rom for forbedringer. Den utregnede MCC-verdien på testsettet er også litt lavere enn gjennomsnittet ± standardavviken fra de 10 gjennomkjøringene av nettverket. Dette kan tyde

på at modellen har en tendens til å overtilpasse seg mot dataen som benyttes i trenings- og valideringssettet.

Begge modellene har tilfeller av feilklassifiseringer over hele landet. Allikevel kan en si at CNN-modellen har flest på Øst- og Sørlandet, og færre i Midt-Norge enn hva en kunne forventet. Dette kan for eksempel skyldes regionale forskjeller i grøftemønster og/eller dybde, ulik forekomst av feilklassifiserbare stier i myrer, eller at det lille datasettet tilfeldigvis er enklere å klassifisere i Midt-Norge. SVM-modellen har en forutsigbar fordeling av feilklassifiserte, hvor de områdene med mye myr også inneholder flere feil.

5.4 Sammenlikning med relatert arbeid

Tidligere studier for kartlegging av myrgrøfting har ofte krevd betydelige mengder med manuell bildeanalyse og kartlegging, i tillegg til at de ofte er tilpasset relativt små områder på opptil 100 km^2. Det er brukt ulike kombinasjoner av klassiske maskinlæringsalgoritmer, sammen med satellittbilder, flyfoto eller flybåren laserskanning. De siste årene har dyplæring, ved blant annet (Robb et al., 2023) og (Habib et al., 2024), tatt over som den foretrukne maskinlæringsmetoden for arealklassifisering, da ved bruk av flyfoto for å segmentere grøftesystemer.

Både (Robb et al., 2023) og (Habib et al., 2024) skriver at et neste steg i deteksjon av grøfter er implementering av LiDAR-avledet høyoppløselig høydedata. (Lidberg et al., 2023) bruker dette for semantisk segmentering av grøftet myr i områder rundt Østersjøen og oppnår en nøyaktighet på 99 % og MCC på 0,78. Artikkelen ble publisert i mars 2023 og var da den høyest ytende modellen for semantisk segmentering av myrgrøfter. I likhet med vår modell, tar (Lidberg et al., 2023) i bruk filter for å hente ut kantegenskaper i bildene. De bruker et high-pass median filter, der det i denne oppgaven er brukt Laplace. Modellen er derimot ikke laget for segmenteringer og tar derfor i bruk ResNet50v2 i et CNN for binær klassifisering, mens (Lidberg et al., 2023) bruker et koder-dekoder-nettverk.

5.5 Videre arbeid

Resultatene fra denne oppgaven er lovende med tanke på videre arbeid med kartlegging av grøftet myr. CNN modellen kan benyttes for å detektere grøfter i kjente myrområder, der digital terrengmodell er tilgjengelig. Dette kan kombineres med en segmenteringsmodell for å lage en helautomatisk prosess for kartlegging av grøftet myr. (Lidberg et al., 2023) skiver for eksempel at deres modell ved hjelp av overføringslære kan tilpasses lokale forhold, og kan nok derfor også benyttes i Norge.

Et umiddelbar neste steg for å forbedre modellen vil være å øke antallet bilder i datasettet. Et større datasett vil øke dekningen av variasjon i inputdataene, som igjen vil hjelpe modellen med å generalisere bedre til usett data. Modellen vil dessuten også kunne plukke opp flere funksjoner og forhold i bildene, hvilket muliggjør en dypere og mer kompleks modellutvikling. Store datasett er også mer motstandsdyktige mot støy og feil i dataene, da tilfeldige feil og unormaliteter blir mindre signifikante.

En annen forbedring av modellen vil kunne skje ved hyperparameteroptimalisering. Dette er en prosess som krever stor computerkraft, og bruk av en sterkere datamaskin vil gjøre det mulig å teste flere ulike kombinasjoner av hyperparametere. I et hav av forhåndstrente modeller kan det også være lurt å lete etter en modell som er trent på høydedata, og da spesielt på terrengmodeller. ResNet-modellen som benyttes er trent på ImageNets datasett med bilder i over 20 000 kategorier gjør en god jobb, men det kan tenkes at en mer tilspisset modell ville gitt enda bedre resultater.

6 Konklusjon

Målet med denne masteroppgaven var å utvikle en maskinlæringsmodell for automatisk deteksjon og klassifisering av grøftet myr fra digital terrengmodell. Modellen ble utviklet ved bruk av overføringslære med ResNet50V2 som forhåndtrent modell. Nettverket består også av et fullt tilkoblet lag, med 288 nevroner og ReLu aktiveringsfunksjon, samt et outputlag med to nevroner og sigmoid aktiveringsfunksjon. Bildene som er brukt i treningen av modellen er hentet fra en digital terrengmodell, hvor områder som ikke var myr ble klippet ut. Bildene i input ble filtrer med et Laplace-filter for å forsterke kantene og redusere mengden informasjon som ikke var knyttet til grøftene. CNN-modellen er trent på data fra hele Norge, og ligger delvis i svært ulent terreng og/eller dekket av skog.

For å teste CNN-modellens ytelsesmål ble det også utviklet en klassifiseringsmodell som tok i bruk støttevektormaskiner for å klassifisere myrene. Ved sammenlikning mellom modellene kom det tydelig fram at CNN presterte vesentlig bedre enn SVM-modellen, da den landet på en F1-score på 0,94 og MCC på 0,87 mens SVM-modellen endte på F1-score på 0,83 og MCC på 0,66. Begge de trente modellene vil være mulige å benytte selv ved begrenset computerkraft, da de kun tar én topografisk indeks som inputdata.

Som konklusjon kan en si at oppgaven viser at det er mulig, og en god idé, å benytte maskinlæring for å detektere grøftet myr ved bruk av digital terrengmodell. Metoden viser svært lovende resultater, og det antas at neste steg i utviklingen vil være å ta i bruk DTM og maskinlæring for å semantisk segmentere grøftelinjer.

Referanser

(n.d.), G. A. https://ai.google/discover/research/

Abadi, M., Agarwal, A., & Barham, P. (2015). TensorFlow: Large-scale Machine Learning on Heterogeneous Systems, v1. 14. In.

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning: A Textbook*. Springer Publishing Company, Incorporated.

Ajagekar, A. (2021). *Adam*. Cornell University. Retrieved 13.02 from https://optimization.cbe.cornell.edu/index.php?title=Adam

Balaji, S. (2020). *Binary Image classifier CNN using TensorFlow*. Medium. Retrieved 06.05 from https://jokerpt.medium.com/binary-image-classifier-cnn-using-tensorflow-a3f5d6746697

Bárcena, T. G., Grønlund, A., Hoveid, Ø., Søgaard, G., & Lågbu, R. (2016). Kunnskapsgrunnlag om nydyrkning av myr. *NIBIO Rapport*, *2*, 43.

Blum, E. S., Porras, A. R., Biggs, E., Tabrizi, P. R., Sussman, R. D., Sprague, B. M., Shalaby-Rana, E., Majd, M., Pohl, H. G., & Linguraru, M. G. (2018). Early detection of ureteropelvic junction obstruction using signal analysis and machine learning: a dynamic solution to a dynamic problem. *The Journal of Urology*, *199*(3), 847-852.

Bryn, A., Strand, G.-H., Angeloff, M., & Rekdal, Y. (2018). Land cover in Norway based on an area frame survey of vegetation types. *Norsk Geografisk Tidsskrift-Norwegian Journal of Geography*, *72*(3), 131-145.

Canny, J. (1986). A computational approach to edge detection. *IEEE transactions on pattern analysis and machine intelligence*(6), 679-698.

Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics*, *21*(1), 1-13.

Chollet, F. (2020, 25.06.2023). *Transfer learning & fine-tuning*. Keras. Retrieved 22.02 from https://keras.io/guides/transfer_learning/

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. 2009 IEEE conference on computer vision and pattern recognition,

Dietterich, T. (1995). Overfitting and undercomputing in machine learning. *ACM computing surveys (CSUR)*, *27*(3), 326-327.

Dong, P., & Chen, Q. (2017). *LiDAR remote sensing and applications*. CRC Press.

Einevoll, O. (1980). Markslagsklassifikasjon i økonomisk kartverk.

French, H. K. (2016). Sammendragsrapport om hydrologien i Åsmyra. *Åsmåsan*

https://img3.custompublish.com/getfile.php/4926269.2703.swltaqksmt7snm/Hydrologien+i+%C3%85smyra.pdf?return=www.as.kommune.no

GDAL/OGR contributors, C. (2020). GDAL/OGR geospatial data abstraction software library. *Open Source Geospatial Foundation*.

Godoy, D. (2018). *Understanding binary cross-entropy / log loss: a visual explanation*. Medium. Retrieved 09.02 from https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a

Gya, R. (2023). *Myr*. Retrieved 30.04 from https://snl.no/myr

Habib, W., Cresson, R., McGuinness, K., & Connolly, J. (2024). Mapping artificial drains in peatlands—A national-scale assessment of Irish raised bogs using sub-meter aerial imagery and deep learning methods. *Remote Sensing in Ecology and Conservation*.

Hagen, H. U. (2023). *Automatisk deteksjon av myr med bruk av hyperspektrale bilder og maskinlæring* Norwegian University of Life Sciences].

Haralick, R. M., Sternberg, S. R., & Zhuang, X. (1987). Image analysis using mathematical morphology. *IEEE transactions on pattern analysis and machine intelligence*(4), 532-550.

Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., & Smith, N. J. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357-362.

Hatlevik, S. E. (2021). *Store mengder ikke-kartlagt myr i fjellet en planleggingsutfordring*. Plan. Retrieved 26.01 from https://plantidsskrift.no/artikkel/store-mengder-ikke-kartlagt-myr-i-fjellet-er-en-planleggingsutfordring/

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition,

Hofstad, K. ((2005-2007)). *Torv*. Store Norske Leksikon. Retrieved 12.01 from https://snl.no/torv

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in science & engineering*, *9*(03), 90-95.

Kartverket. (2024). *Høgdedata og djupnedata*. Retrieved 11.05 from https://www.kartverket.no/api-og-data/terrengdata

Keras. Retrieved 16.03 from https://keras.io/api/applications/#usage-examples-for-image-classification-models

Keras Applications. Keras. Retrieved 27.02 from https://keras.io/api/applications/

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*.

Kyrkjeeide, M. O., Bartlett, J., Rusch, G. M., Sandvik, H., & Nordén, J. (2020). Karbonlagring i norske økosystemer (revidert utgave). In: Norsk Institutt for Naturforskning (NINA).

Leifeld, J., Wüst-Galley, C., & Page, S. (2019). Intact and managed peatland soils as a source and sink of GHGs from 1850 to 2100. *Nature Climate Change*, *9*(12), 945-947.

Li, H., Ye, W., Liu, J., Tan, W., Pirasteh, S., Fatholahi, S. N., & Li, J. (2021). High-resolution terrain modeling using airborne lidar data with transfer learning. *Remote Sensing*, *13*(17), 3448.

Lidberg, W., Paul, S. S., Westphal, F., Richter, K. F., Lavesson, N., Melniks, R., Ivanovs, J., Ciesielski, M., Leinonen, A., & Ågren, A. M. (2023). Mapping drainage ditches in forested landscapes using deep learning and aerial laser scanning. *Journal of irrigation and drainage engineering*, *149*(3), 04022051.

Lovdata. (2006). *Forskrift om berekraftig skogbruk*. Retrieved 06.05 from https://lovdata.no/dokument/SF/forskrift/2006-06-07-593

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*, 115-133.

McKinney, W. (2010). Data structures for statistical computing in python. Proceedings of the 9th Python in Science Conference,

Mekjan, S. (2023). *19 millioner til KI-forskning på kartlegging av myrer, arealregnskap og vasking av reguleringsplaner*. Forskningsrådet. Retrieved 15.03 from https://www.forskningsradet.no/nyheter/2023/19-millioner-til-ki-forskning-pa-kartlegging-av-myrer-arealregnskap-og-vasking-av-reguleringsplaner/

Narkhede, S. (2018). *Understanding Confusion Matrix*. Retrieved 06.05 from https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

O'Malley, Bursztein, T. a., Long, E. a., Chollet, J. a., Jin, F. a., Invernizzi, H. a., & others, L. a. (2019). *KerasTuner*. Retrieved 02.04 from https://github.com/keras-team/keras-tuner

OpenAI. (2024). ChatGPT In.

OpenAI, & SiderAI. (2024). *Scholar GPT*. https://chatgpt.com/g/g-kZ0eYXlJe-scholar-gpt

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345-1359.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., & Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, 2825-2830.

QGIS Development Team, A. (2018). QGIS geographic information system. *Open source geospatial foundation project*, *2018*.

Rahul_Roy. (2023). *ML | Stochastic Gradient Descent (SGD)*. GeeksForGeeks.org. Retrieved 13.02 from https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/

Rapinel, S., Hubert-Moy, L., Clément, B., Nabucet, J., & Cudennec, C. (2015). Ditch network extraction and hydrogeomorphological characterization using LiDAR-derived DTM in wetlands. *Hydrology research*, *46*(2), 276-290.

Raschka, S. (2015). *Python machine learning*. Packt publishing ltd.

Raschka, S., & Mirjalili, V. (2017). Python machine learning second edition. *Birmingham, England: Packt Publishing*.

Raschka, S., & Mirjalili, V. (2019). *Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing Ltd.

Raybaut, P. (2009). Spyder-documentation. *Available online at: pythonhosted. org*.

Robb, C., Pickard, A., Williamson, J. L., Fitch, A., & Evans, C. (2023). Peat Drainage Ditch Mapping from Aerial Imagery Using a Convolutional Neural Network. *Remote Sensing*, *15*(2), 499. https://www.mdpi.com/2072-4292/15/2/499

Roelens, J., Höfle, B., Dondeyne, S., Van Orshoven, J., & Diels, J. (2018). Drainage ditch extraction from airborne LiDAR point clouds. *ISPRS journal of photogrammetry and remote sensing*, *146*, 409-420.

Sampath, R., Watson, M., Rasskin, G., Prasad, S., & Jin, H. (2023). *ResNetV2 backbones*. Keras. Retrieved 27.02 from https://keras.io/api/keras_cv/models/backbones/resnet_v2/

Sharma, S., Sharma, S., & Athaiya, A. (2017). Activation functions in neural networks. *Towards Data Sci*, *6*(12), 310-316.

Søgaard, G., Økseter, R., & Borgen, S. K. (2017). Klimagassutslipp fra torvproduksjon i Norge-Metode, datagrunnlag og utslippfaktorer benyttet i klimagassregnskapet under FNs klimakonvensjon (UNFCCC). *NIBIO Rapport*.

Teien, S. (2022). Semantic segmentation of roof materials in urban environment by utilizing hyperspectral and LiDAR data.

Thorat, M., Pandit, S., & Balote, S. (2022). Artificial Neural Network: A brief study. *Asian Journal For Convergence In Technology (AJCT) ISSN-2350-1146*, *8*(3), 12-16.

Tidemann, A., & Elster, A. C. *Maskinlæring*. Store Norske Leksikon. Retrieved 16.01 from https://snl.no/maskinl%C3%A6ring

Van Rossum, G., & Drake, F. L. (1995). *Python reference manual* (Vol. 111). Centrum voor Wiskunde en Informatica Amsterdam.

Vilde Fluge Lillesund, R. V. H., Kvalevåg, M. M., Else Marte Vold, V. H., Bråten, K. G., Opsahl, J., & Økstad, E. (2018). Utfasing av uttak og bruk av torv – Kunnskapsutredning om konsekvenser for naturmangfold, klima,

næring og forbrukere. *Rapport Miløddirektoratet*, 53.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big data*, *3*(1), 1-40.

Yang, J., Xu, J., Lv, Y., Zhou, C., Zhu, Y., & Cheng, W. (2023). Deep learning-based automated terrain classification using high-resolution DEM data. *International Journal of Applied Earth Observation and Geoinformation*, *118*, 103249.

Øien, D.-I., Fandrem, M., Lyngstad, A., & Moen, A. (2017). Utfasing av torvuttak i Norge-effekter på naturmangfold og andre viktige økosystemtjenester. *NTNU Vitenskapsmuseet naturhistorisk rapport*.

A Vedlegg I

Kildekode

Tabell A.0.1: *Kildekode med lenker til GitLab.*

Filnavn	Link	Commit Hash
CNN.py	Gitlab-lenke	7ef32489
SVM.py	Gitlab-lenke	c012736f
dtm_operations.py	Gitlab-lenke	b2c73f13
utilities.py	Gitlab-lenke	c1368405
lagre_punkt_som_tiff.py	Gitlab-lenke	1d2a2e02
visualization.py	Gitlab-lenke	c012736f

GitLab mappe: https://gitlab.com/tobbjo/mesterprojektet

Programvare

Tabell A.0.2: *Programvare og Python-moduler brukt i oppgaven.*

Program/ modul	Bruk	Versjon	Referanse
Python	Programmeringsspråket benyttet	3.10.13	(Van Rossum & Drake, 1995)
Spyder IDE	Utviklingsmiljø for vitenskapelig koding	5.4.5	(Raybaut, 2009)
Tensorflow	Rammeverk for dyplæring	2.15.1	(Abadi et al., 2015)
NumPy	Vitenskapelige utregninger i Python	1.26.3	(Harris et al., 2020)
Pandas	Databehandling og analyse	1.5.2	(McKinney, 2010)
Matplotlib	Visualisering av data	3.8.0	(Hunter, 2007)
Scikit-learn	Maskinlæringsbibliotek	1.4.0	(Pedregosa et al., 2011)
Osgeo	Behandling av romlige data	3.6.2	(GDAL/OGR contributors, 2020)
QGIS	Geografisk informasjonssystem	3.28.15	(QGIS Development Team, 2018)

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

Postboks 5003
NO-1432 Ås
Norway